

University Degree in Computer Science and Engineering  
2018-2019

*Bachelor Thesis*

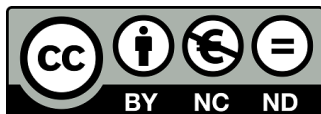
“System for cybersecurity career  
orientation based on self-assessment of  
competences”

---

Francisco Manuel Colmenar Lamas

Ana Isabel Gonzalez-Tablas Ferreres

Leganés, 2019



[Include this code in case you want your Bachelor Thesis published in Open Access University Repository]

This work is licensed under Creative Commons **Attribution – Non Commercial – Non Derivatives**



## SUMMARY

In the last years, cybersecurity industry has tried to mature by defining a set of different work roles and their needed skills, knowledge and abilities. As a consequence of this situation, several frameworks were created so as to facilitate the classification of the different cybersecurity profiles.

The aim of this report is to describe the results of the analysis performed to Sorting Hat as well as discussing the different proposals in order to improve it. Consequently, this project is based on Sorting Hat, whose main functionality is specifying the cybersecurity profile of the user once a self-assessment of competences has been completed. The scope of Sorting Hat is very extensive. This statement is based on the fact that Sorting Hat is focused on providing a service which could be useful for all the students and professionals regardless of where they live or where they are from.

The main objective of "System for cybersecurity career orientation based on self-assessment of competences" Bachelor Thesis is the analysis and the proposal of improvements of the current version of Sorting Hat. This version is the result of several Bachelor Thesis completed by former students of the University Carlos III of Madrid, Spain. The last Bachelor Thesis which was delivered according to Sorting Hat had the name of "Cybersecurity Sorting Hat: Ampliación de funcionalidades de análisis y desarrollo de interfaz de usuario avanzada 2". Its author is Javier Sanz López who was a student at the Bachelor's degree in Telematics Engineering and it was delivered for the 2017/2018 course.

The development of this Bachelor Thesis has been done taking into consideration the functionalities already designed and implemented for the last version of Sorting Hat. Furthermore, in order to be as accurate as possible the report and the code delivered by Javier Sanz López for his Bachelor Thesis is taken as the starting point for the development of this project.

The following document describes and analyzes the mandatory procedures in order to achieve a successful outcome from the development of "System for cybersecurity career orientation based on self-assessment of competences". In order to accomplish this goal, its functionalities as well as the documentation, planning and legal framework among others are taking into account.

## **Keywords**

- API: Application Programming Interface.
- EER: Enhanced Entity Relationship.
- ENISA: European Union Agency for Network and Information Security.
- GDPR: General Data Protection Regulation.
- IISP: Institute of Information Security Professionals.
- JS: JavaScript.
- NCWF: NICE Cybersecurity Workforce Framework.
- NICCS: National Initiative for Cybersecurity Careers and Studies.
- NICE: National Initiative for Cybersecurity Education.
- NIST: National Institute of Standards and Technology.
- PHP: Hypertext Preprocessor.
- SQL: Standardized Query Language.
- TKSAs: Tasks, Knowledges, Skills and Abilities .
- UI: User Interface.
- UML: Unified Modeling Language.



# CONTENTS

1. INTRODUCTION. . . . .	1
1.1. Motivation of Work . . . . .	1
1.2. Goals . . . . .	3
1.3. Document's structure . . . . .	4
1.4. Summary of the results . . . . .	6
1.4.1. Database Results Summary . . . . .	6
1.4.2. JavaScript Results Summary . . . . .	7
1.4.3. PHP Results Summary. . . . .	8
1.4.4. User Interface Results Summary . . . . .	8
1.4.4.1 CSS Results Summary . . . . .	8
1.4.4.2 HTML Results Summary . . . . .	9
1.4.4.3 Responsiveness Results Summary . . . . .	10
1.4.4.4 User Interface Design Results Summary . . . . .	10
2. STATE OF THE ART . . . . .	12
2.1. Current situation . . . . .	12
2.1.1. Cybersecurity Initiatives. . . . .	12
2.1.1.1 Cybersecurity Challenge UK . . . . .	12
2.1.1.2 National Initiative for Cybersecurity Careers and Studies	14
2.1.1.3 ENISA . . . . .	17
2.1.2. Cybersecurity Frameworks . . . . .	19
2.1.2.1 NICE Cybersecurity Workforce Framework (NCWF) .	19
2.1.2.2 IISP Skills Framework . . . . .	22
2.2. Cybersecurity Sorting Hat. . . . .	22
3. ANALYSIS OF THE SYSTEM . . . . .	26
3.1. Architecture of Sorting Hat . . . . .	26

3.2. Analysis of Sorting Hat . . . . .	28
3.2.1. Database analysis . . . . .	29
3.2.1.1 Database Project Structure . . . . .	29
3.2.1.2 Code Style . . . . .	31
3.2.1.3 Relational Data Model . . . . .	33
3.2.1.4 Database Security . . . . .	48
3.2.2. JavaScript analysis . . . . .	51
3.2.2.1 JavaScript Library . . . . .	51
3.2.2.2 JavaScript Project Structure . . . . .	53
3.2.2.3 Code Style . . . . .	56
3.2.3. PHP analysis . . . . .	57
3.2.3.1 PHP Framework . . . . .	57
3.2.3.2 PHP Project Structure . . . . .	59
3.2.3.3 PHP Code Style . . . . .	63
3.2.4. User Interface analysis. . . . .	65
3.2.4.1 User Interface Framework . . . . .	65
3.2.4.2 CSS . . . . .	66
3.2.4.3 HTML . . . . .	71
3.2.4.4 Responsiveness . . . . .	76
3.2.4.5 User Interface Design . . . . .	85
4. IMPROVEMENTS PROPOSED FOR SORTING HAT* . . . . .	97
4.1. Proposed Architecture for Sorting Hat* . . . . .	97
4.2. Description of the improvements for Sorting Hat* . . . . .	97
4.2.1. Proposed Database improvements . . . . .	98
4.2.1.1 Database Project Structure . . . . .	98
4.2.1.2 Code style . . . . .	100
4.2.1.3 Relational Data Model . . . . .	103
4.2.1.4 Security . . . . .	109

4.2.2. Proposed JavaScript improvements . . . . .	111
4.2.2.1 JavaScript Library . . . . .	111
4.2.2.2 JavaScript Project Structure . . . . .	112
4.2.2.3 JavaScript Code Style . . . . .	114
4.2.3. Proposed PHP improvements . . . . .	115
4.2.3.1 PHP Framework . . . . .	115
4.2.3.2 PHP Project Structure . . . . .	116
4.2.3.3 PHP Code Style . . . . .	119
4.2.4. Proposed User Interface improvements . . . . .	121
4.2.4.1 User Interface Framework . . . . .	121
4.2.4.2 CSS . . . . .	122
4.2.4.3 HTML . . . . .	127
4.2.4.4 Responsiveness . . . . .	131
4.2.4.5 User Interface Design . . . . .	134
5. PLANNING AND BUDGET . . . . .	139
5.1. Planning. . . . .	139
5.2. Budget of the project. . . . .	142
5.2.1. Material Cost . . . . .	142
5.2.2. Human Resource Cost . . . . .	143
5.2.3. Total Cost . . . . .	143
6. REGULATORY FRAMEWORK . . . . .	144
6.1. Network and Information Security Directive . . . . .	144
6.2. Ley orgánica de Protección de Datos. . . . .	145
6.3. Ley de Conservación de Datos . . . . .	146
7. CONCLUSION . . . . .	147
7.1. Full-filling of the initial objectives . . . . .	147
7.2. Summary of the results . . . . .	148
7.2.1. Database Results . . . . .	148
7.2.2. JavaScript Results . . . . .	150



7.2.3. PHP Results . . . . .	151
7.2.4. User Interface Results . . . . .	152
7.2.4.1 CSS Results . . . . .	152
7.2.4.2 HTML Results . . . . .	153
7.2.4.3 Responsiveness Results . . . . .	154
7.2.4.4 User Interface Design Results . . . . .	155
7.3. Future work. . . . .	156
BIBLIOGRAPHY. . . . .	157
APPENDIX A: USER MANUAL FOR DEPLOYING SORTING HAT. . . . .	



## LIST OF FIGURES

1.1	Graph representing the growth of Global ICT developments from 2001 to 2018 [2] . . . . .	1
2.1	Education’s tab from cybersecuritychallenge.com.uk [10] . . . . .	13
2.2	Career’s tab from cybersecuritychallenge.com.uk [11] . . . . .	13
2.3	Typical roles’ tab from cybersecuritychallenge.com.uk [12] . . . . .	14
2.4	Training’s page from NICCS [14] . . . . .	15
2.5	Workforce’s page from NICCS [15] . . . . .	16
2.6	Framework’s page from NICCS [16] . . . . .	17
2.7	ENISA’s main page [18] . . . . .	18
2.8	ENISA’s education page [18] . . . . .	18
2.9	ENISA’s cybersecurity exercises page [18] . . . . .	19
2.10	Example of a category from NCWF [19]. . . . .	20
2.11	Example of a specialty from NCWF [20]. . . . .	20
2.12	Example of a workrole from NCWF [21]. . . . .	20
2.13	Example of task from NCWF [22]. . . . .	20
2.14	Example of skill from NCWF [23]. . . . .	21
2.15	Example of knowledge from NCWF [24]. . . . .	21
2.16	Example of ability from NCWF [25]. . . . .	21
2.17	Example of workrole from NCWF [26]. . . . .	21
2.18	Example of ability from NCWF [28]. . . . .	22
2.19	Sorting Hat’s sign in page [18] . . . . .	23
2.20	Sorting Hat’s home page [18] . . . . .	24
2.21	Sorting Hat’s test page [18] . . . . .	24
2.22	Results of Sorting Hat’s test page . . . . .	25
2.23	Graphs representing the results of Sorting Hat . . . . .	25

3.1	Typical three layers application[31]	27
3.2	MCV architecture[32]	28
3.3	EER's entity [43]	35
3.4	EER's Zero or one relationship [43]	36
3.5	EER's one relationship [43]	36
3.6	EER's one and only one relationship [43]	36
3.7	EER's one or many relationship [43]	36
3.8	EER's many relationship [43]	36
3.9	EER's mandatory relationship [43]	37
3.10	EER's optional relationship [43]	37
3.11	EER of cyberhatdb Database	38
3.12	Output of user's remember_token query	42
3.13	Output of user's email query	43
3.14	Log in page in a small computer screen	78
3.15	Log in page in a large computer screen	78
3.16	Home page in a small computer screen	79
3.17	Home page in a large computer screen	79
3.18	Account page in a small computer screen	80
3.19	Account page in a large computer screen	80
3.20	Log in page in a mobile phone screen	82
3.21	Home page in a mobile phone screen	83
3.22	Account page in a mobile phone screen	84
3.23	The figures displayed in the Test page are not clear	87
3.24	No figures are displayed next to "%" at the DashBoard page	87
3.25	The figures displayed over the graphs are not clear	88
3.26	Diagram at the Home page with no consistency with the color palette used in Sorting Hat	89
3.27	Inconsistency of the text paragraphs at the Home page	90
3.28	Inconsistency in the font used for the Navigation Bar	91

3.29	The text in the Home picture cannot be correctly read . . . . .	91
3.30	The Navigation Toolbar always shows the same tab as the one selected . .	92
3.31	Meaningless error message at the Log in page . . . . .	93
3.32	Meaningless error message at the Certificate Creation page . . . . .	93
3.33	Asymmetry in the Log in and Register form . . . . .	94
3.34	Asymmetry of the diagram at the Home page . . . . .	94
3.35	Asymmetry of the text paragraphs at the Home page . . . . .	95
5.1	Planning of the Bachelor Thesis . . . . .	141
6.1	Summary of Ley orgánica de Protección de Datos . . . . .	146



## LIST OF TABLES

1.1	Scope of the analysis accomplished to Sorting Hat . . . . .	5
1.2	Summary of the analysis' result and of the proposed improvements for the Database . . . . .	7
1.3	Summary of the analysis' result and of the proposed improvements for the JavaScript . . . . .	7
1.4	Summary of the analysis' result and of the proposed improvements for the PHP . . . . .	8
1.5	Summary of the analysis' results and of the proposed improvements for the CSS . . . . .	9
1.6	Summary of the analysis' results and of the proposed improvements for the HTML . . . . .	10
1.7	Summary of the analysis' results and of the proposed improvements for the Responsiveness . . . . .	10
1.8	Summary of the analysis' results and of the proposed improvements for the User Interface Design . . . . .	11
3.1	Database Project Structure analysis . . . . .	31
3.2	SQL Code Style analysis . . . . .	33
3.3	Relational Data Model analysis . . . . .	48
3.4	Database Security analysis . . . . .	51
3.5	JavaScript Library analysis . . . . .	53
3.6	JavaScript Project Structure analysis . . . . .	55
3.7	JavaScript Code Style analysis . . . . .	57
3.8	PHP Framework analysis . . . . .	59
3.9	PHP Project Structure analysis . . . . .	63
3.10	PHP Code Style analysis . . . . .	65

3.11	User Interface Framework . . . . .	66
3.12	CSS Project Structure analysis . . . . .	69
3.13	CSS Code Style analysis . . . . .	71
3.14	HTML Project Structure analysis . . . . .	74
3.15	HTML Code Style analysis . . . . .	76
3.16	Responsiveness analysis . . . . .	85
3.17	User Interface Design analysis . . . . .	96
4.1	Database Project Structure improvements . . . . .	100
4.2	SQL Code Style improvements . . . . .	103
4.3	Relational Data Model improvements . . . . .	109
4.4	Database Security improvements . . . . .	111
4.5	JavaScript Library improvements . . . . .	112
4.6	JavaScript Project Structure improvements . . . . .	113
4.7	JavaScript Code Style improvements . . . . .	115
4.8	PHP Framework improvements . . . . .	116
4.9	PHP Project Structure improvements . . . . .	119
4.10	PHP Code Style improvements . . . . .	121
4.11	User Interface Framework improvements . . . . .	122
4.12	CSS Project Structure improvements . . . . .	124
4.13	CSS Code Style improvements . . . . .	126
4.14	HTML Project Structure improvements . . . . .	129
4.15	HTML Code Style improvements . . . . .	131
4.16	Responsiveness improvements . . . . .	134
4.17	User Interface Design improvements . . . . .	138
5.1	Material Cost . . . . .	142
5.2	Human Resource Cost . . . . .	143
5.3	Total Cost . . . . .	143
7.1	Result of the analysis and the proposed improvements for the Database . .	150



7.2	Result of the analysis and the proposed improvements for the JavaScript .	151
7.3	Results of the analysis and the proposed improvements for the PHP . . .	152
7.4	Results of the analysis and the proposed improvements for the CSS of the User Interface . . . . .	153
7.5	Result of the analysis and the proposed improvements for the HTML of the User Interface . . . . .	154
7.6	Results of the analysis and the proposed improvements for the Respon- siveness of the User Interface . . . . .	155
7.7	Results of the analysis and the proposed improvements for the Design of the User Interface . . . . .	156

## **Clarification of the project**

Before starting to describe the development of this Bachelor Thesis, the previous projects accomplished in the former years are highlighted.

"System for cybersecurity career orientation based on self-assessment of competences" is the analysis of the deficiencies of current implementation of Sorting Hat system as well as the description of a set of improvements proposed to address the elicited deficiencies and a description of the proposal improvements for Sorting Hat.

Sorting Hat was originally developed by Javier Vila as the Master thesis named "Cyber Range Systems: A Cybersecurity Sorting Hat" as part of the Master of Cybersecurity at the University Carlos III of Madrid.

Subsequently it was improved by Javier Sanz López and Sandra Sánchez Esperante in the Bachelor Thesis named "Cybersecurity Sorting Hat: Ampliación de funcionalidades de análisis y desarrollo de interfaz de usuario avanzada 2" in 2018 as part of the Bachelor's Degree in Telematics Engineering at the University Carlos III.

# 1. INTRODUCTION

In this chapter an introduction of the developed project will be provided. Furthermore, the reasons why this project has been chosen will be explained. The main functionalities as well as the sections of the analysis which is going to be perform to Sorting Hat will be briefly described too. Besides, in the last section of this chapter the structure of this Bachelor Thesis will be outlined.

## 1.1. Motivation of Work

It is known that nowadays a great amount of the population if the developed countries are not able of living without access to Internet resulting in a new addition [1]. Due to the fact that nowadays a great majority of people use their smart-phone for accessing to Internet, the smart-phone has become an essential tool for a great amount people.

This increasing of the world wide access to Internet can be clearly understood thanks to the following graph. It shows the exponential growth which mobile cellular telephone subscriptions has experienced in the last years starting with twenty million of subscriber in 2001 up to one hundred and seven million of subscribers in 2018 [2].

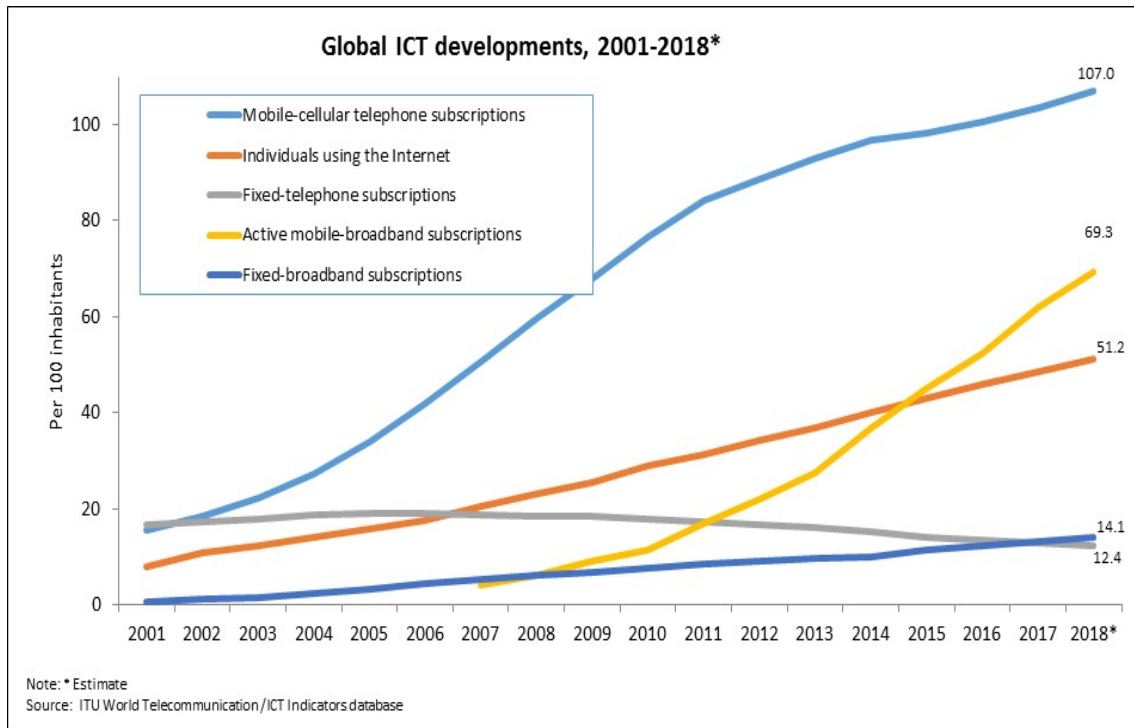


Fig. 1.1. Graph representing the growth of Global ICT developments from 2001 to 2018 [2]

As a consequence of this increase of the number of Internet's users, the number of cyber attacks suffered along the world has drastically raised. Moreover, a significantly figure is the fact that more than 4000 ransomware attacks occur every day which should be enough to raise awareness of the magnitude of the problem [3].

The businesses and enterprises all over the world are the most damaged by this attacks. The focus of this attacks on the business sector result in a huge lost of profit. If the cost to the global economy is taking into account, only in 2017 as much as 600\$ billion were lost as a result of a cyber attack [4]. Furthermore, the effect which one of these attacks could have over the reputation of the attacked company could be much worse creating a sense of mistrust and fear among their clients.

Consequently, nowadays the role of cybersecurity expert is highly valuable for most of the companies in every country. However, there is a significant gap between the number of cybersecurity roles needed to be filled and the number of available cybersecurity experts. This situation does not appear to change in the near future. A report from *Cybersecurity Ventures* estimates that the amount of available jobs in the cybersecurity field in 2021 will raise up to 3.5 millions [5].

In order to improve the current situation of cybersecurity, the National Institute for Cybersecurity Education, known as NICE, proposed the NICE Cybersecurity Workforce Framework, known as NCWF. This framework is the framework used by Sorting Hat which its current state is going to be analyzed and several proposals are going to be described in order to improve it [6].

Sorting Hat is a Web Application whose main service is to specify the cybersecurity profile of the user through the full filling of a self-assessment of competences. Furthermore, Sorting Hat is focused on providing a useful service for students as well as for professionals regardless their nationality.

This framework proposed seven different categories, thirty three different specialization areas and fifty two work roles. These roles are defined in terms of the knowledge, abilities and skills needed in order to be sufficiently qualified to perform correctly and efficiency its job [7].

Sorting Hat has the potential of helping to raise awareness about the importance of cybersecurity. Moreover, it also gives the possibility to security experts to know more about what skills and knowledge are needed for the different roles which are available nowadays for applying and therefore to better orientate their professional careers.

Furthermore, the main problem which the current implementation of Sorting Hat experiences is the lack of an efficient structure regarding the Database, User Interface, Business logic and dealing with errors. The purpose of this Bachelor Thesis is giving an insight of the current situation of SortingHat as well as providing proposals in order to improve the system.

Regarding to the implementation of these suggestions, it has been decided to just give detailed guidelines about their implementation instead of actually implementing them. This is one of the most important decisions which have been taken according to this Bachelor Thesis.

This decision was taken due to the huge amount of work which a detailed and correct analysis of the system, describing all the different improvements and implementing them will take.

At first, the idea of this Bachelor Thesis was refactoring the code in order to improve its quality. However, once that the first analysis of Sorting Hat was performed, it was clear that due to time and Bachelor Thesis workload limitations there was no time enough to perform a high quality analysis, proposing the improvements and implementing them.

This decision was made taking into consideration the best interest for the end user. Consequently, the choice which benefits the most to the user in the long term is accomplishing this improvement plan into two different stages: the analysis and the guidelines for the improvements first, and then fully implement these measurements.

Only in this way Sorting Hat\* could be really improved in order to be useful for the users for a greater amount of time and facilitating any further modification [8].

Furthermore, every decision made during this project will be justified and the reasoning why it is the best possible choice for the current situation of Sorting Hat will be given in order to be as accurate and clear as possible.

In the case that it is desired to run Sorting Hat locally the Appendix A can be used as a guide .

## 1.2. Goals

The aim of this Bachelor Thesis is to analyze the previous architecture of Sorting Hat as well as describing several proposed improvements for it in order to provide an optimal service to the users. Giving this improvements, the amount of students and professionals who could potentially benefit from Sorting Hat\* could be drastically increased.

In order to accomplish the aforementioned objective, the following requirements have to be satisfied.

- *Database analysis*: The current situation of the Database will be described taking into account different aspects such as its Relational Data Model and the Code Style followed.

- Business Logic analysis: The logic of the Business implementation of the current situation of Sorting Hat will be studied too. An analysis of the PHP implementation as well as the JavaScript will be performed.
- User Interface analysis: As well as the Database, the User Interface of Sorting Hat will be evaluated. Aspects such as reactivity and intuitiveness will be taken into consideration during the analysis.
- Database proposed improvements: Several proposals for improving the Database as well as fixing the issues found in the analysis will be described. Furthermore, a diagram of the resulting database for further understanding of the updated system.
- Business Logic proposed improvements: Proposals for improving and enhancing the efficiency of Sorting Hat\* Business Logic will be explained in detail.
- User Interface proposed improvements: Guidelines for a correct User Interface implementation will be delivered in order to facilitate the further implementation of an upgraded User Interface. These improvements will provide a much more satisfactory service when interacting with Sorting Hat\* for the users.

### 1.3. Document's structure

The aim of this section is to provide a guideline of the structure which this document will follow. This document is divided into the following chapters.

- Introduction and goals: In this chapter the current situation of the cybersecurity market will be described providing an insight on the importance that this field has achieved in the last years. Additionally, the objective of this Bachelor Thesis will be discussed in order to provide a general perspective of the document.
- State of the Art: The aim of this chapter is giving a technical description of the background of this project. The approach followed in this section is the comparison of the current project with further available systems which offer similar services to the users. Furthermore, an explanation about the content of NICE Cybersecurity Workforce Framework will be provided. The correct explanation of this framework is essential in order to fully understand this Bachelor Thesis. In addition to this, a brief visual insight of the functionalities of SortingHat is given.
- Analysis of the system: A complete evaluation of the current state of the system will be accomplished. Aspects such as the Database, User Interface and the Business Logic of Sorting Hat will be assessed among others.

- *Improvements proposed for the new system:* Once that the former system has been described and analyzed, a new system will be proposed in order to fulfill all the weakness of the previous architecture.
- *Planning and budget:* This chapter is of great importance due to the fact that it details the planning followed during the development of this Bachelor Thesis. Consequently, a more accurate insight about this project can be obtained. Moreover, an approximate budget for the development of the project accomplished in this Bachelor Thesis will be explained in detail.
- *Conclusions:* In this chapter all the conclusions and ideas which have been obtained thanks to the development of this project will be outlined and explained. Furthermore, the goals proposed for this Bachelor Thesis will be analyzed in order to determine the fulfilling of the prospects of this project. The possibility of further projects which could continue developing Sorting Hat\* will be discussed.

Furthermore, in order to give an insight about the aspects which have been taken into account during the analysis, the following table is going to be used in order to summarize all the different aspects of the analysis.

	Scope of the analysis						
	Database	Business Logic		User Interface			
		JavaScript	PHP	CSS	HTML	Respon- siveness	UI Design
Project Structure	√	√	√	√	√	-	-
Code Structure	√	√	√	√	√	-	-
Code Style	√	√	√	√	√	-	-
Relational Data Model	√	-	-	-	-	-	-
Database Security	√	-	-	-	-	-	-
Library/ Framework	-	√	√	√	√	-	-
Respon- siveness	-	-	-	-	-	√	-
UI Desing	-	-	-	-	-	-	√

Table 1.1: Scope of the analysis accomplished to Sorting Hat

## 1.4. Summary of the results

A summary of the results of the analysis and the improvements proposed is going to be described in the following section. The sections which are summarized are the Database results, page 6; JavaScript results, page 7; PHP results, page 8.

In order to build the tables of the results summary, the most important aspects of the analysis and the most meaningful improvements are the ones which have been chosen to be part of the following tables.

A more detailed summary of the results is accomplished in the chapter 6 section 1, page 148

### 1.4.1. Database Results Summary

In this section a summary of the analysis' results and the proposal of the Database improvements is going to be displayed in the following table.

Database Results Summary			
Module	Section	Analysis	Improvement
Project Structure	File Structure	Only one file in the project folder	Add to the Database's folder the current version of it  Inside the Database's folder create one folder for <i>development</i> and other folder for <i>release</i>
	Code Structure	Only one file of 7173 lines of code	Refactor the original SQL file to fulfill the requirements of the proposed File Structure
Code Style	Table Naming	Inconsistency of tables' names	Use of collective names when possible  Singular form for table's names
	Column Naming	Inconsistency in columns' names	Meaningful and brief names  Use Snake Case as the naming standard
Relational Data Model	Trivial columns	<i>"NCWF_SpecialityArea_NCWF_Category_id"</i> is not needed in table <i>"ncwf_workrole"</i> . With <i>"NCWF_SpecialityArea_id"</i> the category can be accessed	<i>"NCWF_SpecialityArea_NCWF_Category_id"</i> is removed from table <i>"ncwf_workrole"</i>
	Similar columns	<i>"SourceReference"</i> and <i>"Source"</i> columns from <i>"knowledge"</i> , <i>"task"</i> , <i>"skill"</i> , <i>"ability"</i> , have the same values	<i>"SourceReference"</i> column from <i>"knowledge"</i> , <i>"task"</i> , <i>"skill"</i> , <i>"ability"</i> are deleted



Database Results Summary			
Module	Section	Analysis	Improvement
Relational Data Model	Inappropriated column format	"NCWF_SpecialityArea_id", "NCWF_SpecialityArea_NCWF_Category_id" and "NCWF_Category_id" could contain up to 11 digits when only 2-3 are needed.	The columns "NCWF_SpecialityArea_id" and "NCWF_Category_id" are changed from containing 11 digits to 4 digits
	Missing Foreign Keys	"user_has_workrole" is not linked with "user" nor "ncwf_workrole"	The column "type" from "profile" is set to be a foreign key to the table "profile_type"
Database Security	MySQL accounts	No password is defined for the root user	Establishing a password for the root user
	Database Backup	There is no backup to restore the actual Database if an adversity happens	Protecting the access to root privilage

Table 1.2: Summary of the analysis' result and of the proposed improvements for the Database

## 1.4.2. JavaScript Results Summary

In this section a summary of the analysis' results and the proposal of the JavaScript improvements is going to be displayed in the following table.

JavaScript Results Summary			
Module	Section	Analysis	Improvement
Library	jQuery	JQuery Library is loaded two times in each HTML file	Load JQuery Library only once at the end of the HTML body
Project Structure	File Structure	Two equal folders for storing the JavaScript files	Delete the folder called "javascript"
		Duplicated files	Delete the duplicated file called "jquery.js"
	Code Structure	Non descriptive file's name	Rename the file called "arboles.js" to "testPage.js"
Code Style		Absence of JavaDoc comments	Implement JavaDoc comments to all the methods
	Variable Naming	Inconsistency of variables' name	Consistency of variables' name is enforced following the camelCase naming standard
	Hardcoded String Variables	Several String comparisons are performed using hardcoded Strings in the middle of the method	The hardcoded Strings used in the comparisons should be declare at the top of file

Table 1.3: Summary of the analysis' result and of the proposed improvements for the JavaScript

### 1.4.3. PHP Results Summary

In this section a summary of the analysis' results and the proposal of the PHP improvements is going to be displayed in the following table.

PHP Results Summary			
Module	Section	Analysis	Improvement
Framework	Laravel	Laravel feature for the correct implementation of Responsible User Interfaces is not fully used	Implementation of Vue components
Project Structure	File Structure	Mixing of several types of files inside the folder <i>public</i> with no organization	A different folder is created for each type of file in order to group the files by their type
	Code Structure	No refactoring of the files No organization inside <i>.blade.php</i> files	PHP files are divided into different methods Refactor the <i>CSS</i> and <i>PHP</i> code from the <i>blade</i> files
Code Style	Line Breaks	Inconsistency of the amount of line breaks used for separate blocks of code	Only two line breaks between block of codes if they are not related
		Inconsistency of the amount of line breaks used for separate the comments	No line break between a comment and the code to which it refers
	Indenting	Inconsistency in the amount of left indenting used	Be consistent with the left indenting with the previous code
	Control Structures	No consistency in the use of spaces in the control structures	One space after the keyword and between the closing parenthesis and the opening brace
	Comments Language	All the comments are written in Spanish	All the comments should be translated into English

Table 1.4: Summary of the analysis' result and of the proposed improvements for the PHP

### 1.4.4. User Interface Results Summary

In this section the different features of the User Interface which has been assessed during the analysis and the proposal of improvements are going to be summarized.

#### 1.4.4.1 CSS Results Summary

In this section a summary of the analysis' results and the proposal of the CSS improvements is going to be displayed in the following table.

CSS Results Summary			
Module	Section	Analysis	Improvement
Framework	Bootstrap	Incorrect implementation of a responsive User Interface	Upgrade the User Interface to a fully responsive version using the Framework Bootstrap
Project Structure	File Structure	Mixing of CSS files with other type of files inside the folder <i>public</i> with no organization	Inside <i>public</i> a folder called <i>css</i> is created to store the CSS files
	Code Structure	No refactoring of the files CSS code written inside the HTML code	Refactor the code from the independent CSS file Refactor the CSS code from the HTML code
Code Style	Line Breaks	Inconsistency of the amount of line breaks used for separate blocks of CSS code	Use only one Line Break to separate CSS code blocks
	Indenting	Inconsistency in the amount of left indenting used	Use only two spaces as left indenting
	Shorthand Properties	Absence of Shorthand Properties for some rules	Use Shorthand Properties in all the rules which accept this approach
	Capitalization	Inconsistency of the use of lower case	Use only lower case for all the CSS files

Table 1.5: Summary of the analysis' results and of the proposed improvements for the CSS

#### 1.4.4.2 HTML Results Summary

In this section a summary of the analysis' results and the proposal of the HTML improvements is going to be displayed in the following table.

HTML Results Summary			
Module	Section	Analysis	Improvement
Project Structure	File Structure	There is no folder for storing Laravel Layouts	Create the folder <i>layout</i> inside <i>resources/views</i> to store the Laravel layouts
	Code Structure	Layouts are not created Existence of PHP, JavaScript and CSS code inside the HTML code	Implement Laravel Layouts for reusing HTML code Refactor all the PHP, JavaScript and CSS code inside the HTML code into new files
Code Style	Line Breaks	Inconsistency of the amount of line breaks used for separate blocks of HTML code	Use only one line break to separate blocks of HTML code
	Indenting	Inconsistency in the amount of left indenting used	Use only two spaces as left indenting
	Images	No meaningful <i>alt</i> attribute is added to the images	Add a meaningful " <i>alt</i> " attribute to the images

HTML Results Summary			
Module	Section	Analysis	Improvement
Code Style	"Type" Attribute	Specify the "type" attribute when importing a CSS or JavaScript file	Delete the "type" attribute from the imports of CSS and JavaScript file

Table 1.6: Summary of the analysis' results and of the proposed improvements for the HTML

### 1.4.4.3 Responsiveness Results Summary

In this section a summary of the analysis' results and the proposal of the Responsiveness improvements is going to be displayed in the following table.

Responsiveness Results Summary			
Module	Section	Analysis	Improvement
Responsiveness	Computer Responsiveness	Log in page background does not fill the whole screen for large screens	At the Log in page resize the size of the background image or fix it to fit all the screen sizes
		Home images get cut if the screen is smaller, instead of been resized	At the Home page resize or move the images in order to fit the images in all the screen sizes
	Mobile Responsiveness	Log in page background does not fit the whole screen	At the Log in page resize the size of the background image or fix it to fit all the screen sizes
		At the Account page only one quarter of the whole screen is used	At the Account page change the layout used for displaying the elements to use more of the available screen space

Table 1.7: Summary of the analysis' results and of the proposed improvements for the Responsiveness

### 1.4.4.4 User Interface Design Results Summary

In this section a summary of the analysis' results and the proposal of the User Interface Design improvements is going to be displayed in the following table.

User Interface Design Results Summary			
Module	Section	Analysis	Improvement
User Interface Design	Clarity	The figures displayed in the Test page are not clear	At the Test page use only two decimal digits when displaying the results

User Interface Design Results Summary			
Module	Section	Analysis	Improvement
User Interface Design	Color Palette	Diagram at the Home page without any consistency to the color palette used	At the Home page change the color palette of the diagram located at the bottom of the page to be consistent to Sorting Hat* color palette
	Consistency	Inconsistency in the font used for the Navigation Bar	Use the same style and size for the font of the Navigation Toolbar for all the pages
	Effectiveness	The Navigation Toolbar always shows the same tab as the one selected	Mark in the Navigation Toolbar the page in which the user is
	Error Messages	Meaningless error message at the Log in page	Create a customized error messages for each of the input fields of the Log in and Create Certificate
	Symmetry	Asymmetry in the log in and in the register form	At the Log in page place the log in and register forms symmetrically each other from the center of the screen

Table 1.8: Summary of the analysis' results and of the proposed improvements for the User Interface Design

## 2. STATE OF THE ART

### 2.1. Current situation

It is known that there is an increasing need of cybersecurity specialist for the companies from all over the world. As a consequence of the wide spectrum of the cybersecurity field, these companies most of the times demand very specific cybersecurity roles. Based on this need, Sorting Hat and many other Web Applications has been developed in order to fulfill this market's demand.

In the following sections some Cybersecurity Initiatives will be analyzed and their characteristic features will be discussed so as to compare them to the features implemented by Sorting Hat in order to identify their differences.

#### 2.1.1. Cybersecurity Initiatives

According to the Cybersecurity Initiatives which have been implemented by different institutions and countries, the three most interesting and important initiates are the ones which are going to be describe in the following sections.

##### 2.1.1.1 Cybersecurity Challenge UK

*cybersecuritychallenge.org.uk* is a non-for-profit organization aimed to solve the main cybersecurities issues which the United Kingdom is facing. However, its whole website is created in such a way that it gives information about the cybersecurity careers and education in a general manner. Therefore, no matter where the user is from that the he will find useful information at this website for him [9].

Furthermore, its features are slightly different to the ones provided by Sorting Hat. Cybersecurity Challenge provides a specific tab focus on the different education courses and paths needed in order to become a cybersecurity expert. The information given at this web site goes from schools to post-graduated courses increasing notably its scope of action.

As a consequence, the main focus of *cybersecuritychallenge.org.uk* is sharing general information about cybersecurity education resources. Nonetheless, Sorting Hat is focus on helping to the cybersecurity students in order to determine their current cybersecurity

profile regarding their skills and knowledge.

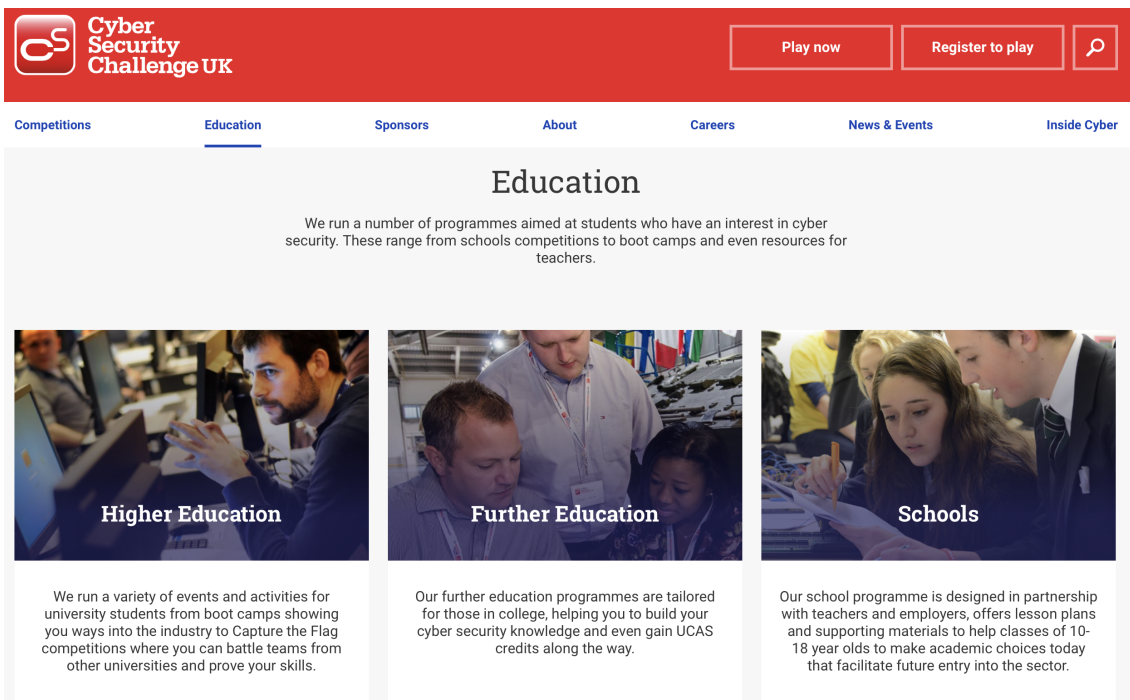


Fig. 2.1. Education's tab from cybersecuritychallenge.com.uk [10]

Another interesting feature which *cybersecuritychallenge.org.uk* includes is a broad definition of several work roles. It contains a page where there is the option of choosing the "typical roles" or choosing "development paths".

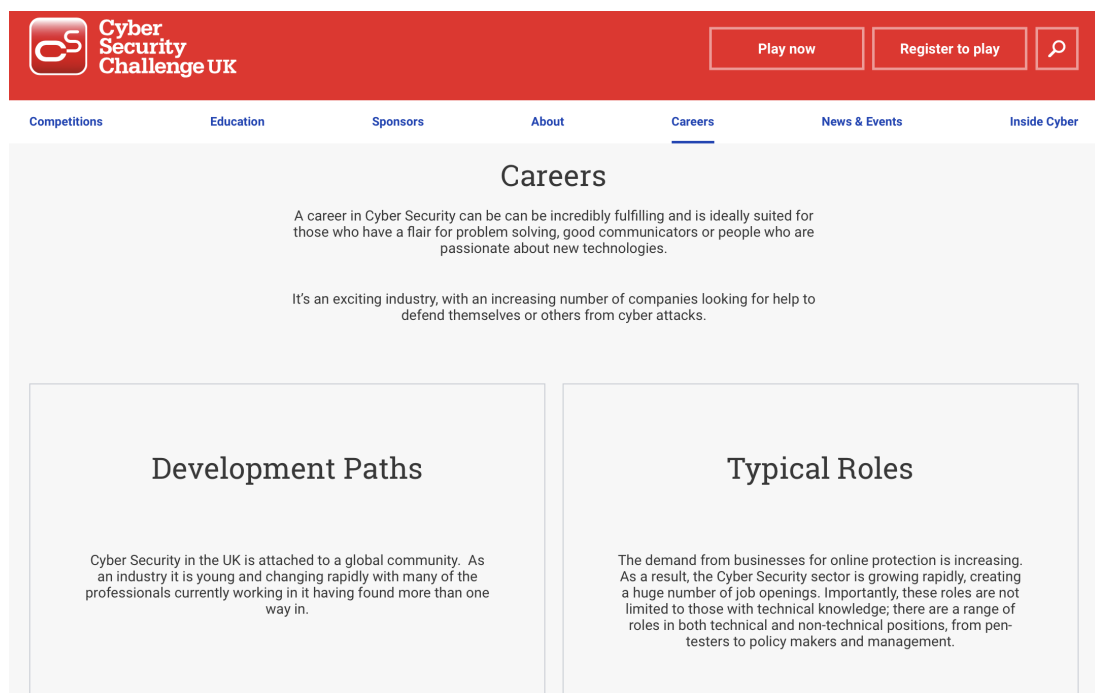


Fig. 2.2. Career's tab from cybersecuritychallenge.com.uk [11]

Both of these options contains very interesting and useful information regarding to these two types of work-roles. The *"typical roles"* page contains details of a great variety of roles with a very intuitive and understandable description of them.

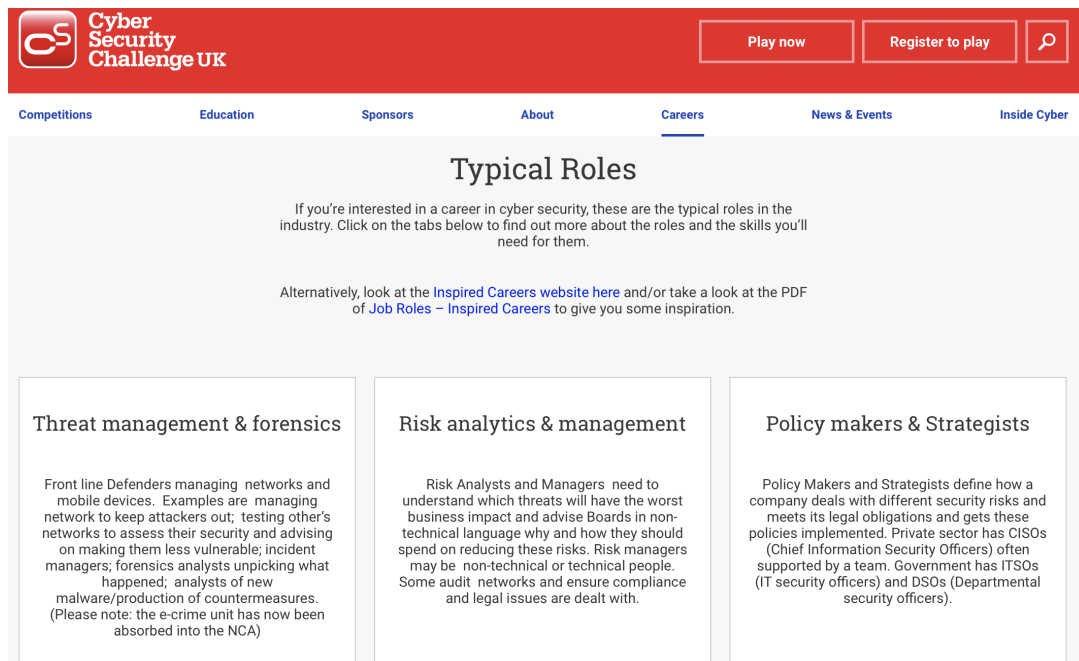


Fig. 2.3. Typical roles' tab from cybersecuritychallenge.com.uk [12]

### 2.1.1.2 National Initiative for Cybersecurity Careers and Studies

The National Initiative for Cybersecurity Careers and Studies, known as NICCS, is a Web Application from the United States of America, known as USA. The department in charge of this website is the Department of Homeland Security. Because of this, this website is focus on USA citizens [13]. However, although this Web Application is not really useful for European students, great ideas and suggestions for other Web Applications of this field, such as Sorting Hat, could be taken from NICCS.

One of the main pages of NICCS Web Application is the Training tab. At this page the user can search for cybersecurity courses and training along the United States of America. Furthermore, useful features in order to optimize the search are implemented.



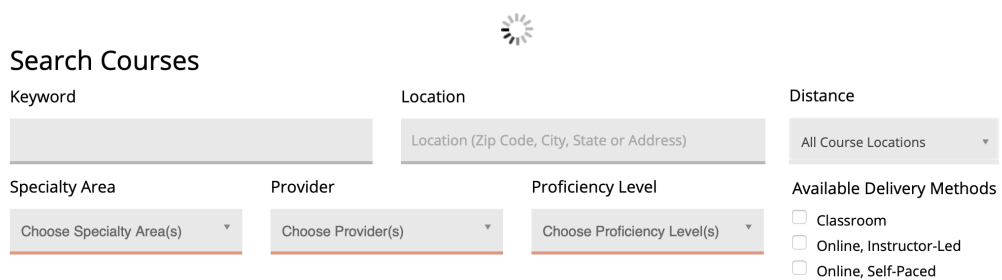
# NICCS Education and Training Catalog


[Become a Provider](#)

The NICCS Education and Training Catalog is a central location where cybersecurity professionals across the nation can find over 3,000 cybersecurity-related courses. Anyone can use the interactive map and filters to search for courses offered in their local area so they can add to their skill set, increase their level of expertise, earn a certification, or even transition into a new career.

All of the courses are aligned to the specialty areas of the [National Cybersecurity Workforce Framework](#).

For any organizations or academic institutions interested in listing courses, apply to [become a provider](#) today! Have any questions? Contact the [NICCS Supervisory Office](#).





### Search Courses

Keyword

Location

Distance

Specialty Area

Provider

Proficiency Level

Available Delivery Methods

- Classroom
- Online, Instructor-Led
- Online, Self-Paced

Fig. 2.4. Training’s page from NICCS [14]

Workforce is another tab which has practical information. At this page the user could access to an online library, cybersecurities resources, a mapping tool or further material about the framework used in this field, which is going to be discussed in the following section. This page is one of the most useful and important pages due to the great amount of information which the user could access from it.

# Workforce Development

Cybersecurity is a national priority and critical to the well-being of organizations. As technology becomes increasingly sophisticated, demand for an experienced and qualified workforce is essential.



Fig. 2.5. Workforce's page from NICCS [15]


Finally, the Framework panel redirects the user to a page in which the NICE Cybersecurity Workforce Framework is described in detail. The different categories, work roles, tasks, skills, knowledges and abilities of this framework are clearly described in several tabs within this page. NICE Cybersecurity Workforce Framework is the framework in which Sorting Hat is based. Consequently, the Web Application Sorting Hat contains a page which has similar functionalities to the aforementioned page.

# NICE Cybersecurity Workforce Framework

The National Initiative for Cybersecurity Education (NICE) Cybersecurity Workforce Framework (NICE Framework), published by the National Institute of Standards and Technology (NIST) in NIST Special Publication 800-181 [\[16\]](#), is a nationally focused resource that establishes a taxonomy and common lexicon to describe cybersecurity work, and workers, regardless of where, or for whom, the work is performed.

Click the blue headers below to search within the NICE Framework components or by keyword. Learn more about how to use the NICE Framework [here](#).

[Categories/Specialty Areas](#) | [Work Roles](#) | [Tasks](#) | [Skills](#) | [Knowledge](#) | [Abilities](#) | [Keyword Search](#)

**Analyze**  
Performs highly-specialized review and evaluation of incoming cybersecurity information to determine its usefulness for intelligence. [Specialty Areas ^](#)

- [All-Source Analysis](#)  
Analyzes threat information from multiple sources, disciplines, and agencies across the Intelligence Community. Synthesizes and places intelligence information in context; draws insights about the possible implications.
- [Exploitation Analysis](#)  
Analyzes collected information to identify vulnerabilities and potential for exploitation.
- [Language Analysis](#)  
Applies language, cultural, and technical expertise to support information collection, analysis, and other cybersecurity activities.

Fig. 2.6. Framework's page from NICCS [16]

In conclusion, the main difference between *The National Initiative for Cybersecurity Careers and Studies* and Sorting Hat is the fact that the target of the first one is providing educational resources and information for USA citizens according to cybersecurity. However, *The National Initiative for Cybersecurity Careers and Studies* does not provide an evaluation for the users in order to determine their cybersecurity profile as Sorting Hat does.

### 2.1.1.3 ENISA

<https://www.enisa.europa.eu> is the official web page of the European Union Agency for Network and Information Security, known as ENISA. ENISA is an expertise centre for cybersecurity in Europe located in Greece. Its main purpose is giving recommendations, helping in the implementation of policies and collaborating with operational teams, all of them related with network and information security [17].

According to <https://www.enisa.europa.eu>, it is mainly focus in cybersecurity news and sharing information, standards and paper related to network and information security. Nevertheless, even though <https://www.enisa.europa.eu> has a great amount of important information it does not provide the functionality which Sorting Hat provides which is the evaluation of the cybersecurity profile of the users.

The main page of <https://www.enisa.europa.eu> has several tabs for searching different

information as well as the recent news posted in the web.

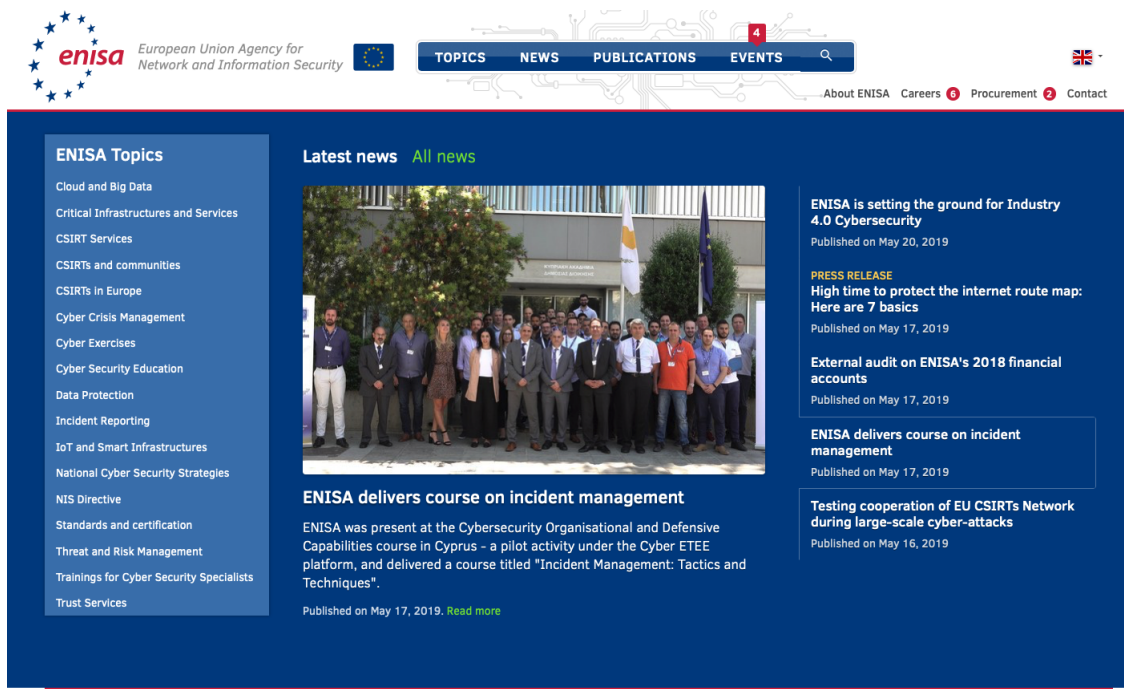


Fig. 2.7. ENISA's main page [18]

At the education page several articles and papers can be downloaded according to this topic.

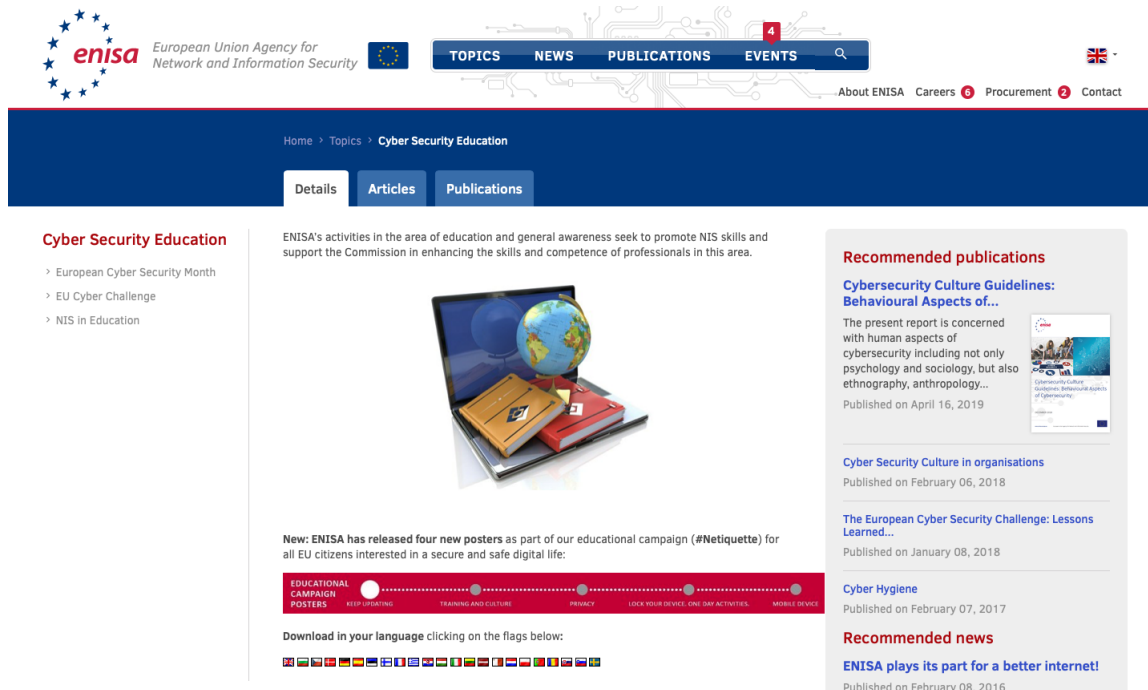


Fig. 2.8. ENISA's education page [18]

At the Cyber Exercises page, there are information regarding the Cyber Europe programme, trainings and other exercises supported by ENISA.

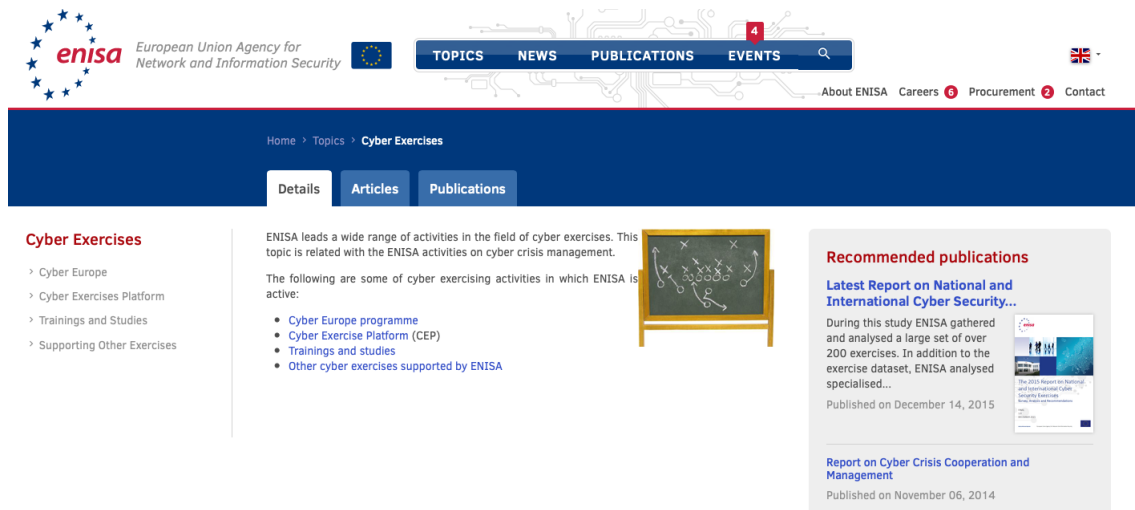


Fig. 2.9. ENISA's cybersecurity exercises page [18]

## 2.1.2. Cybersecurity Frameworks

In the following section the current frameworks regarding to cybersecurity and some forthcoming frameworks will be described. Furthermore, their similarities and differences will be discussed.

The reason why NCWF has been chosen as the cybersecurity framework to be followed in this Bachelor Thesis and in the Web Application Sorting Hat will be presented in order to give a more detailed insight of the reasoning behind the development of this project.

### 2.1.2.1 NICE Cybersecurity Workforce Framework (NCWF)

The National Institute of Standards and Technology, known as NIST, published in the NIST Special Publication 800-181 the The National Initiative for Cybersecurity Education Cybersecurity Workforce Framework, known as the NICE Framework [16].

The main objective of NCWF is to determine a common terminology for defining as specific as possible the different work roles related to cybersecurity as well as the workers, tasks and the needed skills and knowledge needed in this field.

Even though this framework was created by the National Institute of Standards and Technology which belongs to the U.S. Department of Commerce, NCWF can be applied

outside of USA. This standard could be a guideline in order to define the skills and knowledges needed to fulfill a cybersecurity job vacancy regardless where is this job localized or if this job is for a Government Department or for a private company.

The NICE Cybersecurity Workforce Framework is constituted of the following components:

- **Categories:** At NCWF document there are 7 different categories according to cybersecurity workroles.

Categories	Descriptions
Securely Provision (SP)	Conceptualizes, designs, procures, and/or builds secure information technology (IT) systems, with responsibility for aspects of system and/or network development.

Fig. 2.10. Example of a category from NCWF [19].

- **Specialty areas:** 33 particular specialty areas are defined along the document. Each of these specialty areas are related to a previously defined category.

Categories	Specialty Areas	Specialty Area Descriptions
Securely Provision (SP)	Risk Management (RSK)	Oversees, evaluates, and supports the documentation, validation, assessment, and authorization processes necessary to assure that existing and new information technology (IT) systems meet the organization's cybersecurity and risk requirements. Ensures appropriate treatment of risk, compliance, and assurance from internal and external perspectives.

Fig. 2.11. Example of a specialty from NCWF [20].

- **Workroles:** There are 52 cybersecurity work roles described in the analyzed framework.

Category	Specialty Area	Work Role	Work Role ID	Work Role Description
Securely Provision (SP)	Risk Management (RSK)	Authorizing Official/Designating Representative	SP-RSK-001	Senior official or executive with the authority to formally assume responsibility for operating an information system at an acceptable level of risk to organizational operations (including mission, functions, image, or reputation), organizational assets, individuals, other organizations, and the Nation (CNSSI 4009).

Fig. 2.12. Example of a workrole from NCWF [21].

- **Tasks:** There are up to 1007 different tasks related to cybersecurity which are depicted in the NICE Cybersecurity Workforce Framework.

Task ID	Task Description
T0001	Acquire and manage the necessary resources, including leadership support, financial resources, and key security personnel, to support information technology (IT) security goals and objectives and reduce overall organizational risk.

Fig. 2.13. Example of task from NCWF [22].

- *Skills*: At this framework 374 different skills needed in order to perform correctly a cybersecurity job are described.

Skill ID	Description
S0001	Skill in conducting vulnerability scans and recognizing vulnerabilities in security systems.

Fig. 2.14. Example of skill from NCWF [23].

- *Knowledge*: As well as the skills, 630 knowledges are defined regarding to the different cybersecurity work roles.

KSA ID	Description
K0001	Knowledge of computer networking concepts and protocols, and network security methodologies.

Fig. 2.15. Example of knowledge from NCWF [24].

- *Abilities*: There are 176 cybersecurity abilities described in the above-mentioned document which a cybersecurity expert could have depending on the role.

Ability ID	Description
A0001	Ability to identify systemic security issues based on the analysis of vulnerability and configuration data.

Fig. 2.16. Example of ability from NCWF [25].

As it can be seen in the previous examples, as well as defining the different tasks, knowledge and skills they are gather together in different combinations in order to describe the different work roles regarding to the cybersecurity field of study. Furthermore, these work roles are defined in a more exhaustive way in the Appendix B of NCWF in a set of tables, been each table a different workrole.

<b>Work Role Name</b>	<b>Authorizing Official</b>
<b>Work Role ID</b>	<b>SP-RSK-001</b>
<b>Specialty Area</b>	<b>Risk Management (RSK)</b>
<b>Category</b>	<b>Securely Provision (SP)</b>
<b>Work Role Description</b>	Senior official or executive with the authority to formally assume responsibility for operating an information system at an acceptable level of risk to organizational operations (including mission, functions, image, or reputation), organizational assets, individuals, other organizations, and the Nation (CNSSI 4009).
<b>Tasks</b>	T0145, T0221, T0371, T0495
<b>Knowledge</b>	K0001, K0002, K0003, K0004, K0005, K0006, K0013, K0019, K0027, K0028, K0037, K0038, K0040, K0044, K0048, K0049, K0054, K0059, K0070, K0084, K0089, K0101, K0126, K0146, K0168, K0169, K0170, K0179, K0199, K0203, K0260, K0261, K0262, K0267, K0295, K0322, K0342, K0622, K0624
<b>Skills</b>	S0034, S0367
<b>Abilities</b>	A0028, A0033, A0077, A0090, A0094, A0111, A0117, A0118, A0119, A0123, A0170

Fig. 2.17. Example of workrole from NCWF [26].

### 2.1.2.2 IISP Skills Framework

The Institute of Information Security Professionals, known as IISP, is focus on helping the further develop of cybersecurity professionals. In regards to IISP, it is a non-profit organization from the United Kingdom [27].

The Institute of Information Security Professionals created the IISP Skills Framework. This framework defines a set of competences which are supposed to be needed in order to perform correctly the work role of an Information Security and Information Assurance [28].

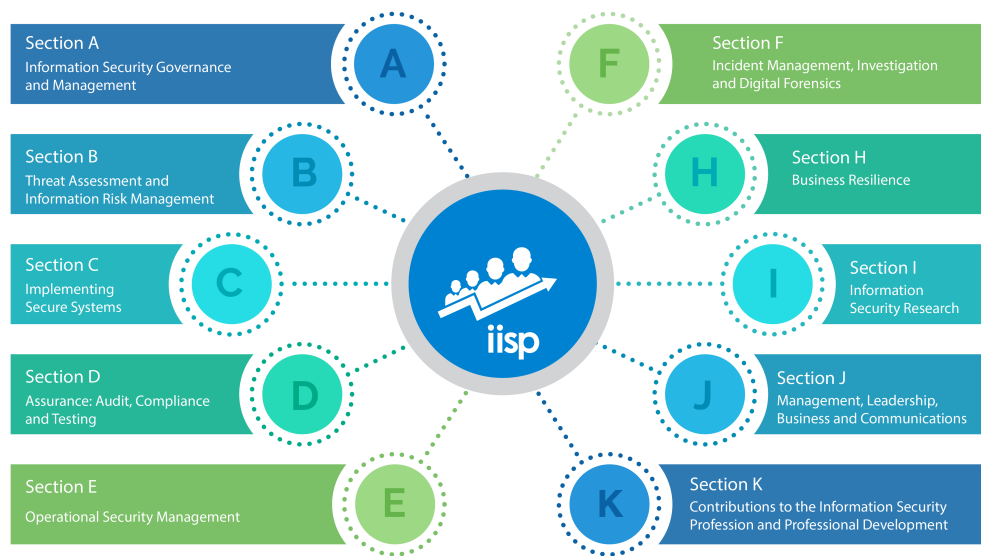


Fig. 2.18. Example of ability from NCWF [28].

## 2.2. Cybersecurity Sorting Hat

As it has been described in the previous sections, there are several applications that perform similar functions to the ones provided by Sorting Hat. Nevertheless, most of these Web Applications are designed for very specific users or just for sharing educational information. However, none of them are focus on students which are already engaged in the cybersecurity area.

Sorting Hat has the potential of becoming of great value for different profiles, since it can be useful for students, workers and companies in the area of cybersecurity. In addition, the decision to choose the NCWF framework for being the one used in this application brings the possibility of increasing the amount of potential users from every country. This is based on the fact that the NCWF framework is the most common and most used framework in the cybersecurity sector worldwide.



Furthermore, in order to have an idea about how Sorting Hat works some pictures of it are going to be displayed as well as a brief explanation about them.

First of all, a page for signing in or registering into Sorting Hat is showed. Once that the user has been created an account it can access to Sorting Hat through it.

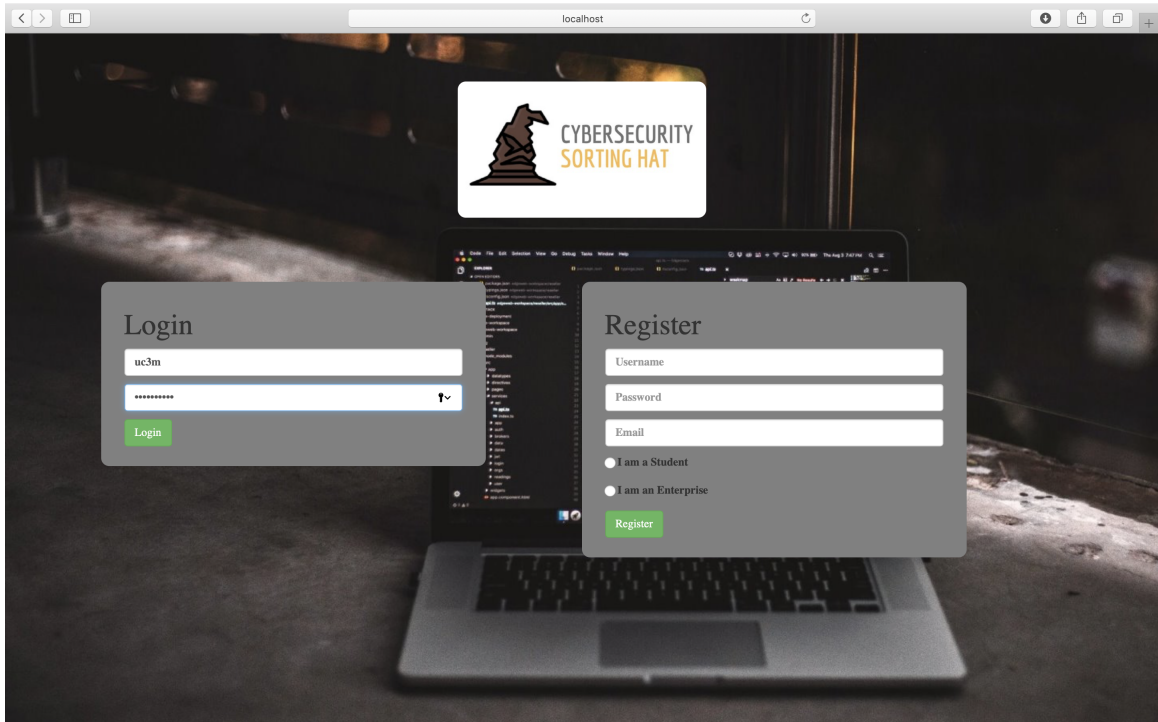


Fig. 2.19. Sorting Hat's sign in page [18]

Once that the user has introduced his credentials into the log in page, the user is redirected to the home page of Sorting Hat. In this page some images according to cybersecurity are shown as well as the navigation bar for accessing the rest of pages of Sorting Hat.

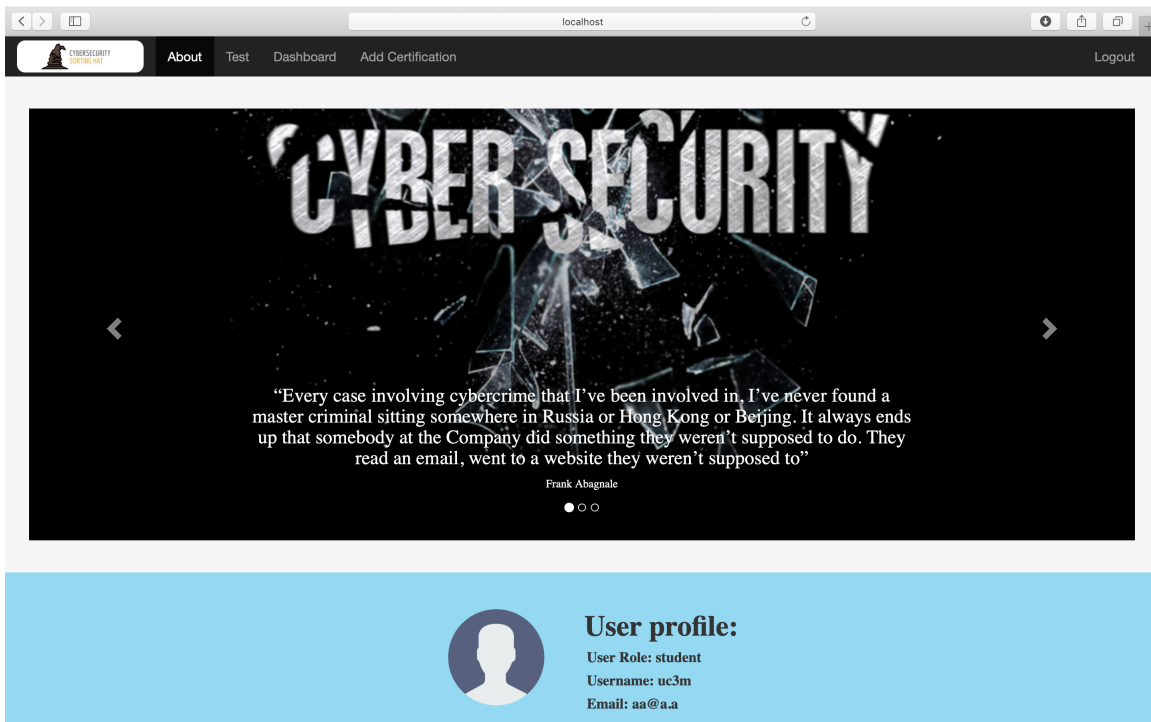


Fig. 2.20. Sorting Hat's home page [18]

The most interesting page of Sorting Hat is the *test* page. In this page the user can take a test about his skills and knowledge in order to determine the cybersecurity profile which best fits with him.

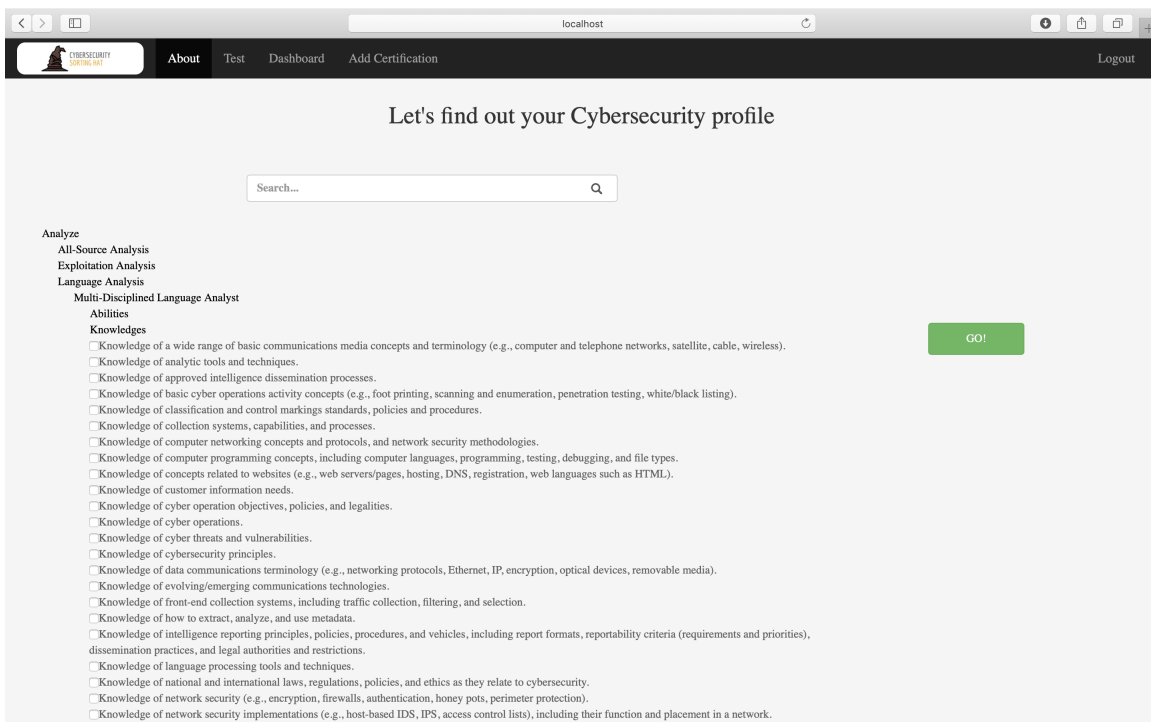


Fig. 2.21. Sorting Hat's test page [18]

Once that the test has been accomplished, the results are displayed in the same page.

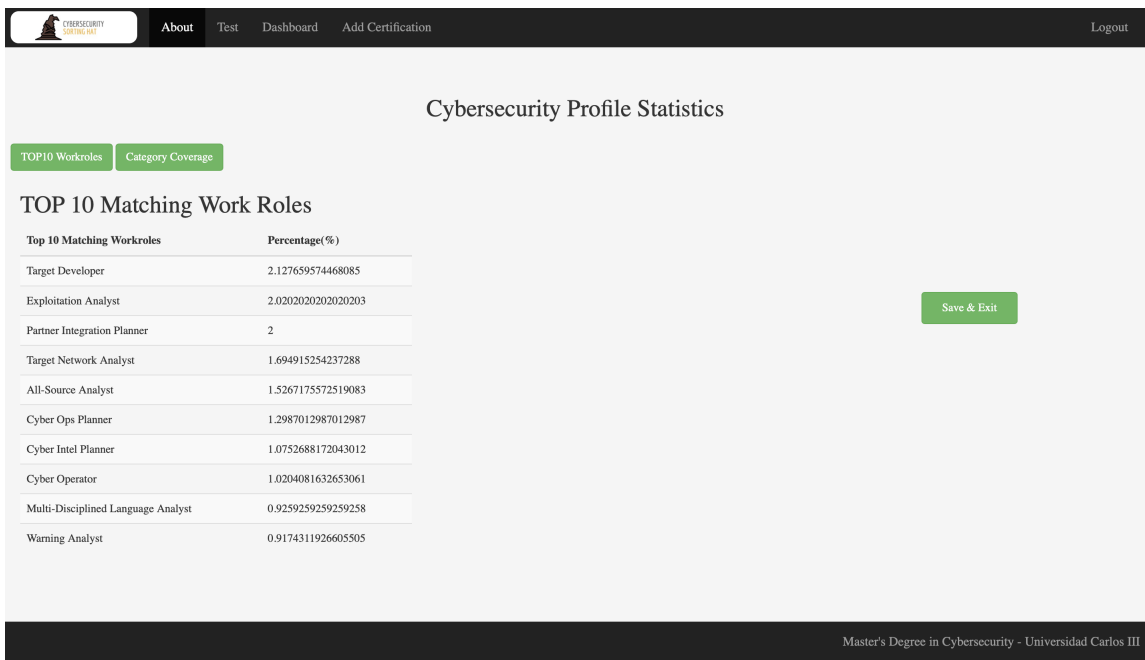


Fig. 2.22. Results of Sorting Hat's test page

Furthermore, Sorting Hat displays several graphs representing the results of all the users registered at Sorting Hat

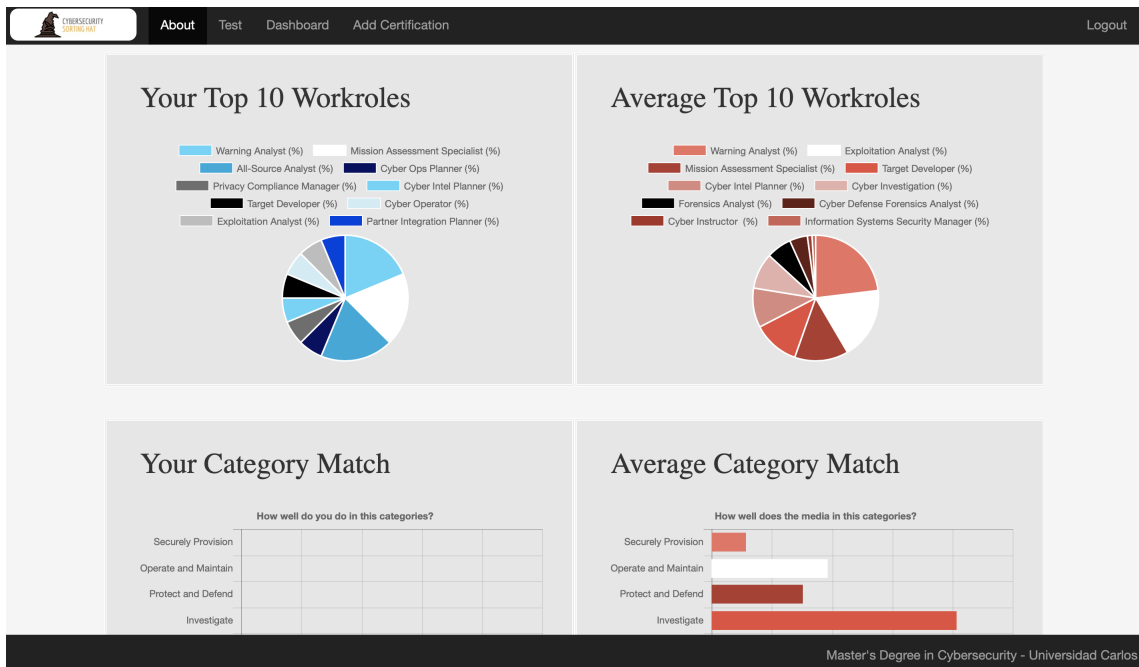


Fig. 2.23. Graphs representing the results of Sorting Hat

### **3. ANALYSIS OF THE SYSTEM**

In this section an analysis of Sorting Hat will be performed. The modules of Sorting Hat which will be assessed are the Database, which corresponds with the MySQL Database; the User Interface, which corresponds to CSS, JavaScript and JavaScript; and its Business or Business logic, which is represented by JavaScript and Laravel. Laravel is a PHP Framework which is used in order to develop Sorting Hat. It will be explained in detailed in the following sections.

As it has been explained in the previous chapter of this document, the objective of Sorting Hat is to help cybersecurity students to find the profile which best suits to their skills and knowledges. A profile corresponds to each of the different positions that are expected to be present in a cybersecurity team.

Each profile has associated to it several skills and knowledges which are the ones supposed to be needed in order to be able of accomplishing correctly the goal of its work role.

To know the most suitable profile for the user and which is the category or department on the field of cybersecurity which fits the best for them, the users must complete a test available in Sorting Hat. They must select the TKSAs they have, and the Sorting Hat will calculate the convenient result.

Nevertheless, in order to be able of providing this service to the students, Sorting Hat has to be designed correctly and implemented following the proper approach.

It is known that most of the users that visit a web page do not spend more than 10 seconds in it. Among the reasons and causes why the users leave a web page the most influential ones are poor design, slow loading, bad layout and not mobile friendly [29].

As it can be observed, in order to be able of providing the service which Sorting Hat is supposed to offer, several design and quality standards need to be fulfill so as to avoid that users leave dissatisfied from Sorting Hat.

#### **3.1. Architecture of Sorting Hat**

A Web Application is designed following one of the several available Web Applications architectures patterns. Some Web Applications have a monolithic architecture. These are usually the simplest websites in terms of complexity. The whole application is deployed

as a single unit and its core behaviour is run just as a single process [30].

Nevertheless, in the case that the complexity of the application increases, the difficulty of implementing it using the same monolithic architecture highly increases too. In order to solve this problem, the "N-Layer" architecture appears.

The "N-Layer" architecture for Web Applications is based on the division of the complexity of the application in several "layers". These layers usually corresponds to the User Interface, Business Logic and the Data Access [31].

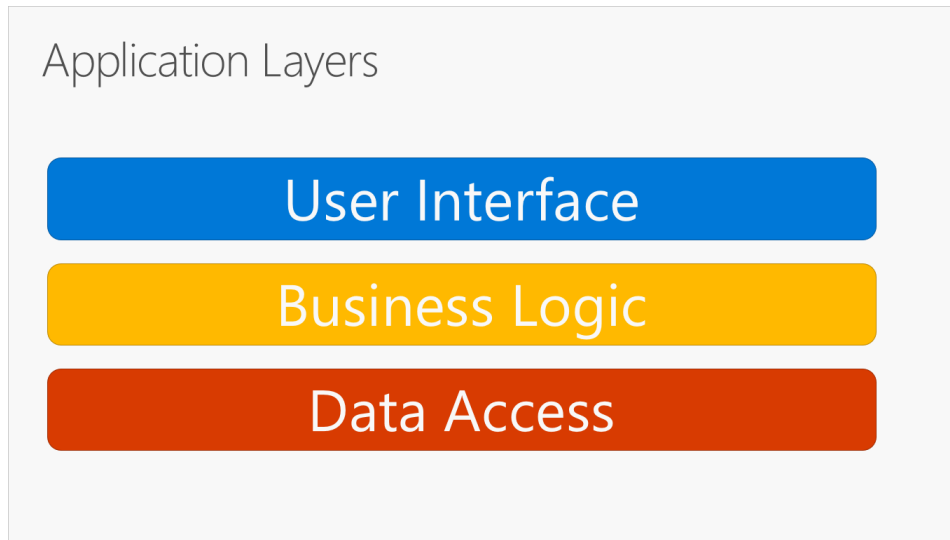


Fig. 3.1. Typical three layers application[31]

Furthermore, Sorting Hat follows the architecture Model-View-Controller which is a variation of the three layered Web Application architecture. It follows the approach of the three layers model dividing then into Controller (Business Logic), View (User Interface) and Model (Data Access and Business Logic) [32]

As a consequence, the implementation of future changes and updates for the Sorting Hat will be smoother and quicker as long as the implementation of the architecture Model-View-Controller has been done correctly.

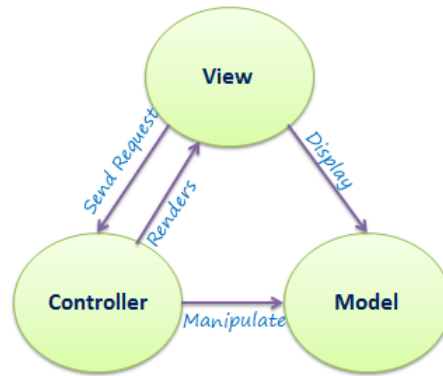


Fig. 3.2. MVC architecture[32]

According to Sorting Hat, its architecture is divided into three different layers:

- ***User Interface***: The presentation layer is the layer which the user can see and interact with. It collects the information introduced by the user and send it to the Business Logic layer. They receive the information that is generated from the logic layer and displays it to the user in a graphical way. In this project, it corresponds with the CSS files, the Laravel views and the HTML files. The JavaScript library called Bootstrap is used in order to modify the User Interface in the case it is needed.
- ***Business Logic***: The Business Logic layer is the layer in which all the logic of Sorting Hat resides. It collects user requests, performs the necessary calculations, requests data to the Data Access Layer if necessary and returns the final information to the user. In this case the Business Logic is implemented through the JavaScript files and the Laravel's framework. The JavaScript library jQuery is used too.
- ***Data Access***: The Data Access layer is the place where the data needed for the correct execution of Sorting Hat is stored. Furthermore, it is also responsible for queering the requested information and to ensure the safeness of the data stored. It is essential that the Data Access layer works perfectly, since it contains critical data needed for the correct operation of Sorting Hat. In the case of Sorting Hat, a MySQL Database has been used in order to access to the data stored in it.

### 3.2. Analysis of Sorting Hat

In this section of the third chapter, the current version of Sorting Hat will be described as well as an analysis of it will be performed. The layers to which the analysis will be focus are the Database, the Logic which is composed by the JavaScript and the Laravel framework implementation, and the User Interface. Each of the layers' analysis will have different goals according to its functionalities.

### 3.2.1. Database analysis

According to the Database analysis, the project structure, code style, relational model, data quality and security will be considered.

#### 3.2.1.1 Database Project Structure

The first stage of the analysis is assessing the project structure of the Database belonging to Sorting Hat.

##### Definition of a proper Database Project Structure

A good Database Project Structure is the one divided into folders and files making the action of finding the query or table which is needed intuitively and efficiently.

##### Importance of a proper Database Project Structure

It is crucial to have a proper Database project structure if efficiency and easy to work is desired to be achieved. It is known that when writing SQL most of the times there is the need of reviewing previous queries or checking the creation of a specific table in order to find the core of a problem [33].

Furthermore, an appropriate Database Project Structure allows the utilization of control version's software such as Git in a more efficient way letting several developers to work in the same project at the same time [34].

##### Project Structure analysis

The analysis of the project's structure is going to be divided into two sections.

1. *File Structure*: the structure of the folders and the files inside them will be analyzed in terms of intuitively and efficient organization.
2. *Code Structure*: the code structure inside the files will be studied in order to resolve if the correct abstraction and refactoring of the code into different files has been correctly performed.

##### File Structure

According to the analysis of the file structure of the project, it was one of the fastest to be accomplished. The Database folder is called "*Bases de datos*" and inside it there is only one SQL file called *cyberhatdb.sql*. The structure of the project is similar to the following diagram:

Bases de datos  
└─ cyberhatdb.sql

As it can be seen, this structure is not efficient for development purposes. It is true that if the Databases' files are not going to be changed and it is only needed an import file for the database this could be a good structure for that case.

Nevertheless, as most of the times there is the need to perform modifications to the SQL files in order to adapt its Database to the changes of Sorting Hat, this file structure is not efficient.

### **Code Structure**

Following the analysis done in the previous section, it is known that there is only one SQL file in the folder structure. Once that the file is opened and examined in detail it is observed that the file has 7173 lines of code. Among these lines of code there are all kinds of SQL statements such as *ALTER TABLE* to *INSERT INTO*.

It is concluded that there is no refraction nor abstraction of the SQL files. As a consequence, it is almost impossible to use efficiently a control version software in order to make any changes in the Database.

Because of this, in the case that any error in the Database is found so as its structure is needed to be changed, the amount of time needed in order to perform correctly a change in this file could be tremendously greater than if a correct file and code structure were implemented.

### **Summary**

In order to outline the main issues of the previous analysis, the following table is going to be used as a reference.



Database Project Structure analysis		
Section	Analysis	Effect
File Structure	Only one file in the project folder	Inefficient development
		Inefective debugging process
Code Structure	Only one file of 7173 lines of code	Impossibility of using control version software
		Inefficient development
		Inefective debugging process

Table 3.1: Database Project Structure analysis

### 3.2.1.2 Code Style

In this section the Code Style of the Database's SQL files will be described.

#### Definition of SQL Code Style

A SQL Code Style is a set of guidelines used when SQL code is written. Due to the fact that there is a wide variety of SQL variations such as MySQL, PostgreSQL or SQLite. As a consequence, there is not an official code style for SQL code.

Although it is true that there is no official style guide for SQL code, it is recommended to follow any good style code and specify which one is the project using in an informative file. In the case that no explicit code style is used, at least a consistency according to the coding style should be maintained for the good of the quality of the code [35].

#### Importance of following a SQL Code Style

The main usefulness of following a SQL Code Style is the improvement of productivity by means of being able to recognize the type of SQL statement in a much faster way. Furthermore, the efficiency of debugging a SQL code is much greater if the analyzed code follows a defined code style rather than if it does not follow any established pattern.

#### SQL Code Style

As it can be recognized when *cyberhatdb.sql* file is opened, there are mainly three types of SQL statements: creation of tables, insertions of values into already created tables and altering the existing tables. All of them are going to be analyzed independently in order to determine the syntax followed in this file.

- CREATE TABLE: According to the *CREATE TABLE* statements, mainly there are the following elements:

- Tables' names: In the case that the names of the tables are analyzed, it is easy to realize that there is no consistency according to the preference used for naming. There are tables named using the lower snake case, such as "*evaluatedprofile\_covers\_targetprofile*", however others tables' names are just all the different words written together with no pattern followed, such as "*proficiencylevel*" [36].
- Columns' names: According to the columns' names inside the tables, the situation is similar as with the tables' names. Sometimes the columns are named after the lower snake case, such as "*created\_at*", following the Pascal Case, such as "*SourceReference*", or a variation of snake case with capital letters, such as "*Evaluated\_Profile\_id*".

The identification field is one of the most important variables inside most of SQL tables. Nevertheless, in this case the identification variable is named after two different names: "*ID*" and "*id*". As it can be seen, the only result which naming the same variable in two different ways can have is the raising of issues in future implementations.

Additionally, naming the identifier of a table as "*id*" is not recommended due to the fact that when several identifiers are retrieved from different tables it could get confusing leading to implementation issues.

- Prefixes and suffixes: As well as column naming is important to maintain the consistency inside a SQL project, using the same prefixes and suffixes style is also of high relevance.

An example of a wrong usage of prefixes and suffixes is related to *id*. There are "*NCWF\_ID*", "*NCWF\_Category\_id*" and "*id\_user\_selected*". As a consequence, there is a high possibility of referencing incorrectly the desired column of the table if there is no code standard followed.

- INSERT INTO: Regarding to the *INSERT INTO* statements, as it uses the variables defined on the creation of the tables the issues are the same as in the previous statement type.
- ALTER TABLE: In the *ALTER TABLE* statements it is shown more consistency than in the previous statements. The order of the declarations of the keys follows in each statement the same order, being the first one the "*primary key*" and the following ones just "*key*"

Furthermore, each of the types of alterations' statements are grouped together mak-

ing it much easier to find a specific key or constraint in order to debug the SQL code.

To sum up, although *ALTER TABLE* statements shows an efficient consistency in its code, the *CREATE TABLE* statement and the fields inside them have no consistency at all. As tables and its fields are the core components of a Database and of SQL code, this makes that the overall consistency according to code style is insufficient and it is highly recommended to improve it.

### Summary

In order to outline the main issues of the previous analysis, the following table is going to be used as a reference.

SQL Code Style analysis		
Section	Analysis	Effect
CREATE TABLE	Inconsistency of tables' names	High possibility of making mistakes referencing a table or column
	Inconsistency in columns' names	
	Inconsistency in prefixes and suffixes	Inefficient development
INSERT INTO	Inconsistency of tables' names	High possibility of making mistakes referencing a table or column
	Inconsistency in columns' names	
	Inconsistency in prefixes and suffixes	Inefficient development
ALTER TABLE	Consistency in the order of keys	Agile recognition of the desired key

Table 3.2: SQL Code Style analysis

#### 3.2.1.3 Relational Data Model

Regarding to the Relational Data Model of the Database, an analysis of *cyberhatdb* design and implementation will be assessed.

## Definition of a Relational Data Model

A Relational Data Model is one of the most popular models for storing and processing data. It represents a database as a collection of relations, been the data in the database described by a compilation of inter-related tables [37]. Each table is composed by a set of columns which represents the attributes of the table and a set of rows which are the records belonging to that entity [38]. It is important to realize that this is the logic model and it is different from the physical model of storage which is totally different.

Often Relational Data Models are represented through diagrams which are called *Enhanced Entity Relationship* diagrams, known as EER. The Enhanced Entity Relationship provides the same features as an Entity Relationship diagram as well as giving a more detailed insight about the data. It is useful when the database to be represented contains complex entities and relationships [39].

## Importance of a proper Relational Data Model

The importance of an efficient Relational Data Model are several. Among them the most relevant are the following ones:

- *Data retrieval*: It allows a more accessible way of retrieving data thanks to the possibility of querying any table belonging to the database. These results can be combined with other queries as well as been filtered in order to get the desired result [40].
- *Simplicity*: A proper Relational Data Model can avoid a great amount of the complexity that in a non relational model or non accurate model would have. This is related to the importance of a Code Style which has been explained in the previous section.
- *Data integrity*: A good Relational Model should ensure the integrity of the data which it contains. This feature will be discussed in the following section.
- *Scalability*: An efficient Relational Data Model enhance and facilitates the modification of the Database if any change is needed, specially because scalability reasons.
- *Security*: Relational Databases incorporates the concept of users and user rights in order to meet with security requirements. This is only effective if it is configured correctly. If not, the result could be the opposite to the desired one letting unwanted users to access and modify the database [41].

The main importance of a Relational Data Model is the fact that if it is not implemented in the correct way, using the Database become a much more complex process with a high possibility of originating new problems. Because of this, designing a correct Relational Data Model should be one of the most important aspects when a project is

being developed.

## Database's Data Relational Model

### Framework

The framework which is going to be used in order to display the Relational Data Model of *cyberhatdb* is *MySQL Workbench*. It provides the possibility of modeling data, SQL development and displaying a data base model using an Enhanced Entity Relationship Diagram. This is the main tool which is going to be used in order to gain a more comprehensive understanding about the database [42].

### Enhanced Entity Relationship Diagram

The *Enhanced Entity Relationship Diagram*, known as EER, is a data model which represents the information of a system in a graphical, direct and easy way to understand it. EER represents how the entities inside a database relate to each other. Furthermore, it uses some of the elements and terminologies of the *Entity Relationship* and the *Unified Modeling Language*, known as UML.

The main usage of these diagrams is during the design and the debug phases of a Database. In the design phase the EER are created and during the debug process the EER are review in order to find the origin of the problem.

The main elements of an Enhanced Entity Relationship diagram are going to be described in order to be able of understanding correctly the meaning of the EER diagram of *cyberhatdb* which is going to be analyzed in the following section [43]. The most important elements at an EER are the following ones:

- Entity: An entity is a determinable "thing" such an user, a knowledge or a skill which can be translated into data. Usually an entity refers to a table in the Database when an EER is being use for database management purposes.

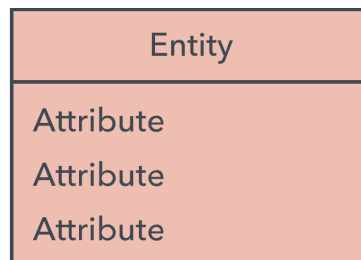


Fig. 3.3. EER's entity [43]

- Relationships: The relationships are the description of how the entities are associated to each other. There are several types of relationships depending of the carnality of the relationship:

- Zero or one: It is a relationship in which the two entities are related by one or none instances. For example, a house could have one or no owners.



Fig. 3.4. EER's Zero or one relationship [43]

- One: The relationship relates two entities through one instance. For example, a house has one owner.



Fig. 3.5. EER's one relationship [43]

- One and only one: Two entities are related by only one instance. The relationship is enforce to have only one instance due to constrains of the data model. For example, a house can only have one owner by law.



Fig. 3.6. EER's one and only one relationship [43]

- One or many: The relationship relates two entities through one or many instances. For example, a house can have one or more owners, it depends on the circumstances.



Fig. 3.7. EER's one or many relationship [43]

- Many: Two entities are related trough many instances. For example, one house have many owners and it cannot have one or none owners.



Fig. 3.8. EER's many relationship [43]

Furthermore, depending on the optionality of the relationships, they can be divided into two types:

- Mandatory: It is a relationship which is mandatory to exist. For example, a house have to have one owner.



Fig. 3.9. EER's mandatory relationship [43]

- Optional: It is a relationship which can exist or not. For example, a house can have or not an owner but in the case that it has an owner it has to have only one.



Fig. 3.10. EER's optional relationship [43]

Once that the basic elements of an EER diagram has been explained, the Enhanced Entity Relationship Diagram of the Database can be described. The EER which corresponds to *cyberhatdb* is the following one:

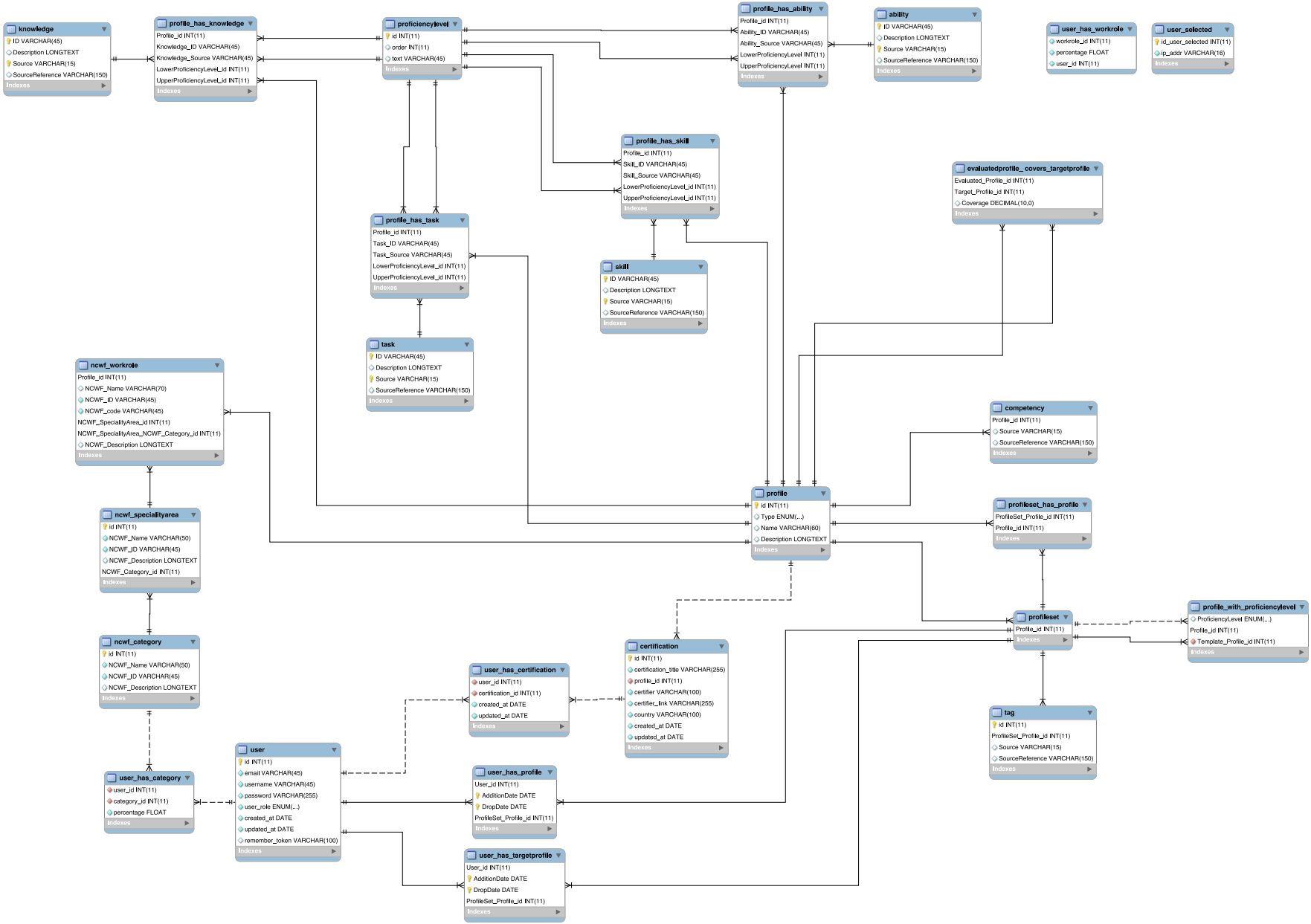


Fig. 3.11. EBR of cyberhardb Database



## Relational Data Model

In order to explain the Relational Data Model of *cyberhatdb* the former EER diagram is going to be used as a reference.

As it can be observed, even though the fact that the database is constituted by 28 different tables there are some of them which are the core tables. These tables are "*knowledge*", "*task*", "*skill*", "*ability*", "*user*", "*ncwf\_workrole*" and "*profile*". As Sorting Hat is based on the NCWF framework which is based on knowledges, skills, tasks and roles too, the Relational Data Model is coherent with it.

The different tables of the database are going to be analyzed in order to determine if there are inefficiencies in its Relational Data Model.

- "*proficiencylevel*": Before describing the core groups of tables which are in the database, the table "*proficiencylevel*" needs to be described so as to have a better understanding of the whole database. This table has three columns: the "*proficiency level identifier*", the "*order*" of this levels and their name which is called "*text*" in the table. There are just 5 instances stored in this table which corresponds to the different levels of proficiency.

However, the problem comes when this level is not specified in the NCWF framework. Furthermore, if all the tables using this table are checked and their values retrieved it is showed that all the instances of all the tables using this table have the same two columns with identical values: the "*LowerProficiencyLevel\_id*" equal to 1 and the "*UpperProfficiencyLevel\_id*" equal to 5.

```
select * from profile_has_knowledge where LowerProficiencyLevel_id != 1  
and UpperProficiencyLevel_id != 5;
```

(3.1)

```
select * from profile_has_task where LowerProficiencyLevel_id != 1  
and UpperProficiencyLevel_id != 5;
```

(3.2)

```
select * from profile_has_skill where LowerProficiencyLevel_id != 1  
and UpperProficiencyLevel_id != 5;
```

(3.3)

```
select * from profile_has_ability where LowerProficiencyLevel != 1
and UpperProficiencyLevel != 5; (3.4)
```

Once that these commands are executed, the following result is displayed for all the former commands.

```
Empty set (3.5)
```

This shows that all the values are trivial due to the fact that all of them are the same and are not based in NCWF.

- "knowledge": This table holds the information of all the different knowledges which the framework NCWF establishes. It contains the identifier, the description of the knowledge and two more columns. These two columns are called "Source" and "SourceReference" with different length for its varchar. If the *insert into* statements are review, it can be seen that there is no difference between the data inserted into "Source" or "SourceReference". Nevertheless, in order to determine if there is any difference between the data stored in both columns the following command is run.

```
select * from knowledge where Source != SourceReference; (3.6)
```

Once that this command is executed the following result is displayed.

```
Empty set (0.01 sec) (3.7)
```

As it is shown, there is no difference from the data store in them. Consequently, all the 614 instances of the knowledge table has one of their four columns completely replicated being the 25% of the information stored into them completely useless.

Furthermore, if the primary key of *knowledge* is checked it is seen that the primary key is the identifier of the knowledge and the Source. Nevertheless, by definition the identifier of a knowledge of the NCWF framework is unique so there is no point of using both columns as the primary key of this table.

The tables related to "knowledge" are:

- "profile\_has\_knowledge": According to the table "profile\_has\_knowledge", it is the table which contains the relationships between the profiles and the knowledges. This kind of tables, which represents the union of a many to many relationship between two tables, are called *junction tables*. It is a very common practise in Database modeling. This table has as columns the pro-

file's identifier, the knowledge's identifier, the source's identifier and the lower and upper proficiency level. According to these two last columns, there is the same issue as the one discussed at the first point.

Furthermore, as it was described in the previous point, there is no need to use the knowledge's source in order to reference it. Consequently, storing this value is trivial and it only wastes storage space as well as it decreases the efficiency of the Database.

- "task": According to the "task" table, it is very similar to the "knowledge" table. It contains the same columns: an identifier, a description and two sources which as in the former table one of them is useless. In order to verify this assumption the same command is used with the only exception that the values are get from "task". As well as in the table "knowledge", there is no need for the source to be part of the primary key of "task".

- "profile\_has\_task": This table is the junction table of "task" and "profile". It has the same problems as "profile\_has\_knowledge". It contains the trivial proficiency columns, the useless source and the identifiers of the task and of the profile.

Just to have an insight about how this issues affect the efficiency of the table's storage. If all the instances of "profile\_has\_task" are displayed, which they are 1287 rows. In all these rows three out of five columns which have the same value for all the rows. Consequently, there are more wasted storage space in this table than useful.

- "skill": This table has similar columns and problems as the previous ones. It contains an identifier, a description of the skill which is related to the NCWF framework and the redundant columns and primary keys previously described.

- "profile\_has\_skill": This junction table joins, as its name suggests, the "profile" table and the "skill" table. It has exactly the same problems as the previous junction tables explained before.

- "ability": Regarding to the table containing the information about the skills described in the NCWF framework, it has the same structure as the previous ones. It contains the two source' columns which one of them is not necessary, the identifier and the description.

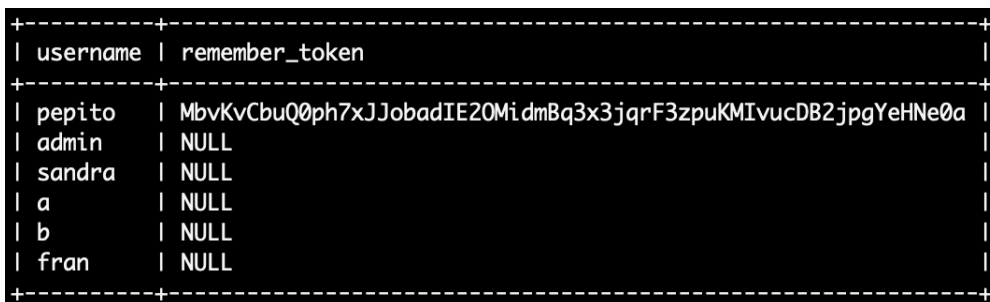
- "profile\_has\_ability": It is the junction table of the "profile" table and the "ability" tables. It has the same structure as the other junction table. Because of this, it shares their problems too. The columns referring to the proficiency levels as well as the reference to the ability's ability are useless.

- *"user"*: According to the *"user"* table, it has the common parameters of a user of a website. It contains an *"id"*, which in this case is an integer of 11 digits, an *"email"*, a *"password"*, an *"user\_role"* which can take the values of *"student"*, *"enterprise"* or *"admin"* and the dates at which the user was created and updated, which names are *"created\_at"* and *"updated\_at"*. In addition to all of these parameters, there is another column that has the name of *"remember\_token"* which its purpose is unclear.

In order to try to gather more information about the purpose of *"remember\_token"* column, the following query was executed so as to retrieve the information regarding to the existing users.

```
select username, remember_token from user; (3.8)
```

The result of this query is the following.



username	remember_token
pepito	MbVkvCbuQ0ph7xJJbadIE20Mi dmBq3x3jqrF3zpuKMIvucDB2jpgYeHNe0a
admin	NULL
sandra	NULL
a	NULL
b	NULL
fran	NULL

Fig. 3.12. Output of user's *remember\_token* query

It can be seen that the only user who has a value different to null in this column is *"pepito"*. Nevertheless, if the insertions to the database are read, it is showed that this value is *"hardcoded"* into the database. Consequently, there is no sign of usefulness of this column.

In addition to this, if the *"email"* column is taken a look, it could be seen that there is no rule which ensures the insertion of correct values into this column. The following query is executed in order to show this situation.

```
select username, email from user; (3.9)
```

Being the result of this query the following.

```

+-----+-----+
| username | email |
+-----+-----+
| pepito   | pepito@gmail.com |
| admin    | admin@cyberhat.es |
| sandra   | sandra@gmail.com  |
| a        | a@a.a             |
| b        | b@a.a             |
| fran     | b@b.b             |
+-----+-----+

```

Fig. 3.13. Output of user's email query

It clearly shows that the input values for "email" are not correctly checked in order to ensure the storing of acceptable data into the database.

Furthermore, the *user* table has several foreign keys to four different junction tables: *user\_has\_category*, *user\_has\_certification*, *user\_has\_profile* and the last one called *user\_has\_targetprofile*. All of them will be explained when their main tables will be assessed.

- "*ncwf\_workrole*": This table holds the information of the different workroles defined by NCWF. The first and most important value which this table contains is the "Profile\_id" of the workrole, which is the identifier of each workrole. However, the issue with this column is that it is set to be up to 11 digits when there is only 52 categories defined in NCWF, which is translated into 2 digits. This decision is not the best one in order to ensure the efficiency of the database.

*ncwf\_workrole* also contains two different identifiers: one for its speciality area and another identifier for its category. They are named "*NCWF\_SpecialityArea\_id*" and "*NCWF\_SpecialityArea\_NCWF\_Category\_id*" respectively. The problem in this case is that the identifier of the category is associated to each speciality area. Consequently there is no need to store both due to the fact that only with the specialty identifier and accessing to the specialty area table would be enough.

Furthermore, according to these two columns, they are set to have up to 11 digits when the higher value which they have is "33", which corresponds to only 2 digits. It is the same issue as with the workrole's identifier.

Additionally, there are four more columns belonging to this table: one column named "*NCWF\_Name*", its NCWF identifier named "*NCWF\_ID*", "*NCWF\_code*" and "*NCWF\_Description*".

There are three tables which are related to "*ncwf\_workrole*": "*ncwf\_specialityarea*", "*ncwf\_category*" and "*user\_has\_category*".

- "*ncwf\_specialityarea*": It holds the information about the different speciality

areas defined in the NCWF. It contains an identifier called *"id"* which has the same issue with its length as the previous ones.

Moreover, it contains *"NCWF\_Name"*, *"NCWF\_ID"*, *"NCWF\_Description"* and *"NCWF\_Category\_id"* with the same length as the other identifiers.

- *"ncwf\_category"*: It contains the information of the different categories defined in NCWF. Additionally, this table contains the identifier *"id"* with the same length's issue, *"NCWF\_Name"*, *"NCWF\_ID"* and *"NCWF\_Description"*.
- *"user\_has\_category"*: This table describes the different categories which the users has. The columns which it contains are *"user\_id"*, *"category\_id"* and *"percentage"* which is a floating point number.

This table is the junction table between *"ncwf\_category"* and *"user"*

- *"profile"*: This table contains the information according to the 52 different profiles defined in NCWF. It contains an identifier *"id"*, *"Type"* which can take the values of *"NCFW\_WorkRole"*, *"Competency"* or *"ProfileSet"*, and *"Description"*.

This table is one of the main tables due to the fact that it is referenced by most of the main tables of the database.

According to the problems of this table, although *"Type"* can take three different values in the practise all the instances are defined as *"NCFW\_WorkRole"* which makes pointless the existence of this column. Additionally, the identifier in this case has up to eleven digits too.

- *"certification"*: It contains all the information according to the cybersecurity certifications. This table stores an identification *"id"*, *"certification\_title"*, *"profile\_id"* which is related with *profile* table, *"certifier"*, *"certifier\_link"*, *"country"*, *"created\_at"* and *"updated\_at"*.
  - *"user\_has\_certification"*: It is the junction table between *user* and *certification*. It just contains the identifications of the user who holds the certification, the identification of the certification, *"created\_at"* and *"updated\_at"*. These two last columns are trivial due to the fact that they already exist with the same information in the table *certification*. However, it could be changed to the date when the user added to his profile this certification which would make more sense.
- *"profileset"*: This table just has one column which is *"Profile\_id"*. However, this table and all the tables which are related to *profileset* are empty. In order to prove

this the following query has been executed.

```
select * from profileset; (3.10)
```

With the following result.

```
Empty set (0.00sec) (3.11)
```

- "user\_has\_profile": This table is the junction table between *user* and *profileset*. Nevertheless, as *profileset* is empty this table is also empty.
- "user\_has\_targetprofile": Junction table between *user* and *profileset*. Because *profileset* has no data stored this junction table has no data too.
- "tag": Related table from *profileset* which is also empty.
- "profile\_with\_proficiencylevel": Table which relate *profileset* with proficiency levels. As the previous tables it is empty.
- "profileset\_has\_profile": It does not have data stored in it. It is the junction table between *profile* and *profileset*.
- "competency": This table is also empty. It is related to *profile* and it is formed by "*Profile\_id*", "*Source*" and "*SourceReferencere*". The purpose of this table is also unclear.

The purpose of *profileset* and all the tables related to them is unclear and it is not documented whether is it needed or not for the correct execution of the Web Application.

- "user\_has\_workrole": This table contains the information of the workroles which each user has. It is not empty and its columns are three: "*workrole\_id*", "*percentage*" and "*user\_id*". Although it is intuitive to link this table with *user* and *ncwf\_workrole* there is no foreign key linking them with this table.
- "user\_selected": It contains the identification of an user called "*id\_user*" and an ip address "*ip\_addr*". This table is also empty and its purpose is imprecise.

## Summary

In order to outline the main issues of the previous analysis, the following table is going to be used as a reference.

Relational Data Model analysis		
Section	Analysis	Effect
Trivial tables	All the columns referencing to <u>"proficiencylevel"</u> table have the same values	
	<u>"profileset"</u> , <u>"user_has_profile"</u> , <u>"user_has_targetprofile"</u> , <u>"tag"</u> , <u>"competency"</u> , <u>"profile_with_proficiency_level"</u> , <u>"user_selected"</u> and <u>"profileset_has_profile"</u> has no data stored in them	Waste of storage Downgrade of Database efficiency
Trivial columns	<u>"LowerProficiencyLevel_id"</u> and <u>"UpperProfficiencyLevel_id"</u> have the same values in all the tables	
	<u>"remember_token"</u> belonging to the table <u>"user"</u> has no purpose nor data	Waste of storage
	<u>"NCWF_SpecialityArea_NCWF_Category_id"</u> is not needed in table <u>"ncwf_workrole"</u> . With <u>"NCWF_Speciality Area_id"</u> the category can be accessed	Downgrade of Database efficiency
	<u>"Type"</u> column belonging to <u>"profile"</u> contains the same data for all the instances	



Relational Data Model analysis		
Section	Analysis	Effect
Similar columns	<p><i>"SourceReference"</i> and <i>"Source"</i> columns from <i>"knowledge"</i>, <i>"task"</i>, <i>"skill"</i>, <i>"ability"</i>, have the same values</p> <p><i>"created_at"</i> and <i>"updated_at"</i> belonging to <i>"user_has_certification"</i> are already stored at <i>"certification"</i></p>	<p>Waste of storage Downgrade of Database efficiency</p>
Inappropriated column format	<p>The columns <i>"Profile_id"</i> could contain up to 11 digits when only 2-3 are needed</p> <p><i>"NCWF_SpecialityArea_id"</i>, <i>"NCWF_SpecialityArea_NCWF_Category_id"</i> and <i>"NCWF_Category_id"</i> could contain up to 11 digits when only 2-3 are needed.</p> <p><i>"Type"</i> column belonging to <i>"profile"</i> replicates the data of the chosen type instead of referencing it</p>	<p>Waste of storage Downgrade of Database efficiency</p>

Relational Data Model analysis		
Section	Analysis	Effect
Trivial Primary Keys	The tables <u>"knowledge"</u> , <u>"task"</u> and <u>"skill"</u> contain as primary key its identifier and its source. Only the identifier is needed because by definition it is unique	Inefficiency of the Relational Data Model
	The tables <u>"profile_has_knowledge"</u> , <u>"profile_has_task"</u> , <u>"profile_has_skill"</u> , <u>"profile_has_ability"</u> , does not need the source and the proficiency's level as primary keys	Database inefficiency
Missing Foreign Keys	<u>"user_has_workrole"</u> is not linked with <u>"user"</u> nor <u>"ncwf_workrole"</u>	Inefficiency of the Relational Data Model
		Database inefficiency

Table 3.3: Relational Data Model analysis

### 3.2.1.4 Database Security

According to the Database Security, several security features of the Database belonging to Sorting Hat will be assessed in the following section.

#### Definition of Database Security

Database Security is a wide term. It refers to all the features and measures which can be accomplished in order to avoid and prevent and illegitimate use of the Database's services [44].

#### Importance of proper Database Security

It is known that nowadays data is one of the most, if not the most, valuable asset for many companies. If an enterprise deals correctly and in an intelligent way with the data that it collects from the users, it could be transformed in an immense competitive advantage. This goal could be achieved using techniques and following an approach of Data Governance [45]. Nevertheless, as this is a very broad topic it is out of the scope of this Bachelor

Thesis.

Consequently, there is a real need for ensuring that the data stored in the databases is not compromised so as to be able to monetize it correctly. Furthermore, apart from the monetary incentive, what is even more important is securing the Database in order to ensure the fulfilling of data protection regulations such as the General Data Protection Regulation, known as GDPR [46].

In addition to this, if a data breach occurs the reputation of the company could be irremediably damaged. There are several examples of companies which in the last years due to their insufficient security measures they have suffered important data breaches in their databases.

One of the most relevant situations was the case of Facebook data breach which occurred the last September. In this data breach at least 30 million of user accounts were affected in this attack and half of these accounts were compromised been some confidential information such as their name or birth date exposed [47]. As a result of this data breach, the company could face a multi-billion fine by regulatory institutions such as the Irish Data Protection Commission [48].

### **MySQL accounts**

To start with, as it can be assumed the most important security issue to avoid is letting anyone accessing the Database with no authentication. This vulnerability can be easily verified by trying to run *MySQL* as root user [49]. In order to perform it the following command is used:

$$mysql -u root \tag{3.12}$$

Once that this *MySQL* command is run, the user is granted access to the whole database with no need of introducing any password. Clearly, this is a great insecurity as anyone with access to the device where the Database is located or through SSH could access to it as a root user. This situation could have dramatic consequence to the integrity of the Database belonging to Sorting Hat.

### **MySQL file permission**

Apart from the Database's users, another important aspect of a Database Security are the actual Database files containing the actual data. The permissions assigned to these files need to force that the only users accessible to them are the users designated to it.

In order to check that these files have the correct permissions, first of all the path to the databases' files is obtained using the following command in *MySQL*.

$$select @@datadir; \tag{3.13}$$

After that, the path obtained need to be accessed in order to navigate there and running the command "*ls -l*" in order to see the permissions of the files [50]. The output to this command is the following:

```
drwx - - - - - 57 _mysql _mysql 182428Mar18 : 50cyberhatdb      (3.14)
```

This shows that the only user who is accessible to the database of Sorting Hat is "*\_mysql*".

## **Backups**

A backup is a copy of the actual or previous state of the Database into a secondary storage. The main point of performing a backup is that the secondary device storage is independent from the main device storage so if the main Database is compromised the backup is not affected by it. This situation could happen by a great amount of different reasons such as an incorrect implementation of the software, an human error or because of an hacker attack [51].

According to *Credhub*, there is no backup policy in order to perform a copy of the Database each predefined period of time. Consequently, if the Database is compromised by any reason, Sorting Hat as a whole would be spoil and probably it could not offer the service to the clients. Furthermore, the Database would become useless and a repopulation of it from scratch would be needed in order to provide the service again.

## **Summary**

In order to outline the main issues of the previous analysis, the following table is going to be used as a reference.

Database Security analysis		
Section	Analysis	Effect
MySQL accounts	No password is defined for the root user	Exposure of the whole Database to unauthorized users
	There is no backup to restore the actual Database if an adversity happens	
Database Backup	There is no procedure to automatize backups	Sorting Hat unaccessible
	It is not clearly stated the period of time while a backup is valid until it is needed to create a new one	Data leaks

Table 3.4: Database Security analysis

### 3.2.2. JavaScript analysis

In regards to the JavaScript analysis, the used Libraries, the JavaScript project structure and the followed code style will be assessed.

#### 3.2.2.1 JavaScript Library

In this section the Libraries used for developing the JavaScript code of *Credhub* will be described and the decision of choosing it will be analyzed.

#### Definition of a JavaScript Library

A JavaScript Library is a collection of class definitions in order to be reused in your own code implementation. It performs exact and well-defined operations. They are usually group and divided into different categories, such as the functions for mathematical operations. It is accessed through an API and it is needed a correct and full understanding of its scope and dependencies in order to be able of implementing it in the correct way [52].

#### Definition of a good JavaScript Library

Deciding which JavaScript Library to use in a project is one of the most important decisions to be taken. This is based on the fact that once a library is implemented into a

project, removing it in order to implement another library is very difficult and time consuming.

There are many different characteristics which could be considered at the moment of deciding which JavaScript library is the best choice for incorporating it to a project. The main reasons are the following ones [53].

- Community: It is essential for a Library to have a wide and active community in order to give the needed feedback and version updates when an issue or inconsistency arises.
- Compatibility: If a Library is considered to be included into a project, its compatibility with the rest of the project modules as well as the main Libraries and Frameworks available in the Internet should be taken into account.
- Features: The more straightforward reason why choosing a Library is the fact that it contains the functionalities for accomplishing what is needed to be done in the project.
- Performance: It is known that speed is a key point regarding to web development due to its importance to the global User Experience. Consequently, the Library chosen do not have to downgrade the performance of the Web Application in a significantly way.
- Stability: One of the reasons why many projects get delayed or even canceled are the bugs and error encountered during its development. As a result, the Library chosen to be implemented in the project should have as few bugs and errors as possible in order to avoid slowing down the whole project development.
- Team: If a Library is being used by big and reliable companies, such as Google or Facebook, it is probable that it is a good option to include it into a project.

## jQuery

The main purpose of using jQuery is easing the implementation of JavaScript into a web development project. Furthermore, JQuery is used by many big companies, such as Uber or Twitter, and it has one of the biggest communities of developers [54]. Consequently, the choice of using jQuery in this project is correct.

Nevertheless, the implementation of this library in the HTML documents is not the optimal. It is known that in order to improve the efficiency of a web page and to let the User Interface of a web page load faster, the scripts, such as JavaScript files, should be imported at the bottom of the body of the Web Application [55].

However, in the HTML files of Sorting Hat, jQuery is loaded two times, one in the

footer and other in the header. This is not the best approach for importing the jQuery library into a project due to the fact that it could slow down the loading of the whole page.

## Summary

Before summarizing the main issues detected during the analysis, it should be outlined the correctness of the choice of using jQuery as the JavaScript library for Sorting Hat.

In order to outline the main issues of the previous analysis, the following table is going to be used as a reference.

JavaScript Library analysis		
Section	Analysis	Effect
jQuery	JQuery Library is loaded two times in each HTML file	Inefficient loading of the User Interface of Sorting Hat
	JQuery is loaded in the header of the HTML	

Table 3.5: JavaScript Library analysis

### 3.2.2.2 JavaScript Project Structure

In this section of the analysis, the JavaScript project structure will be assessed.

#### Definition of a proper JavaScript project structure

As a general rule, a proper JavaScript project structure is the one that improves readability, easy to use and, if possible, it reduces the number of HTTP requests which the user needs to perform in order to load the Web Application.

#### Importance of a proper JavaScript project structure

Building and maintaining a proper JavaScript project structure is crucial for enforcing the development of clean and consistent code. Consequently, searching for a specific element or function within the project would be much more efficient if a project structure is established.

Additionally, following a defined JavaScript project structure will make easier for external or future developers to understand the behaviour of the code in order to improve it [56].

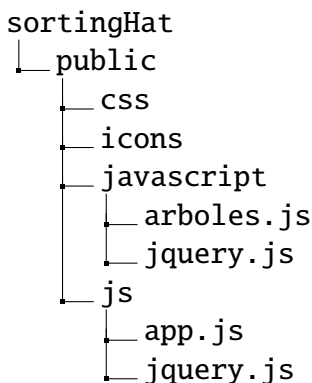
## Project structure analysis

The analysis of the project's structure is going to be divided into two sections.

1. *File Structure*: the folder and file organization will be assessed in terms of usability and efficiency.
2. *Code Structure*: the code of the JavaScript files will be analyzed in order to determine if it is correctly organized and refactored into the needed JavaScript methods and files.

### File Structure

The JavaScript File Structure is simple as well as inefficient. All the JavaScript files are inside two folders: "*javascript*" and "*js*". These two folders are inside a folder called "*public*" inside which different assets are stored too, such as the CSS files. The structure of the project is similar to the following diagram:



Once that the File Structure of the JavaScript files are examined, it can be distinguished its inefficiency. To start with, there are two folders which have almost the same names, "*javascript*" and "*js*". Consequently, it is not clear the purpose of both of them.

Furthermore, if the files inside these folder are analyzed, it is seen that there is a JavaScript file which is duplicated. This file is "*jquery.js*".

As a consequence, not only there are two folders with the same name but there is the same file in both folder increasing the complexity, the probability of errors and the storage dedicated to Sorting Hat.

### Code Structure

According to the Code Structure of the JavaScript files, all the JavaScript from Sorting Hat is located in a single file. This file is called "*arboles*". As it was suggested in previous sections, the names of the files should be more descriptive and, if it is possible, be written



in English in order to facilitate the association of non-Spanish speakers developers to the project.

Even though this file is only executed in the page "*public/test*", it is imported in all the pages from Sorting Hat. This is a known bad practice due to the fact that the user performs unnecessary HTTP requests which could downgrade the efficiency of Sorting Hat.

Furthermore, if the code is analyzed, it is discerned that there is no method with JavaDoc comments. As a consequence, there is no guideline or reference of the execution flow or the purpose of the methods. The probable consequence which this situation could lead to is the misused of the code inside this file originating errors and bugs due to the lack of documentation.

Additionally, there are several logs along the code printing into the console the value of several variables. This action is useless due to the fact that the user do not need to be able of performing any log.

## Summary

In order to outline the main issues of the previous analysis, the following table is going to be used as a reference.

JavaScript Project Structure analysis		
Section	Analysis	Effect
File Structure	Two equal folders for storing the JavaScript files	Inefficient development
	Duplicated files	Ineffective debugging process
Code Structure	Non descriptive file's name	
	Unnecessary importation of the JavaScript files in several pages	Downgrade of the efficiency of the efficiency of Sorting Hat
	Absence of JavaDoc comments	Inefficient development
	Appearance of logs	Ineffective debugging process

Table 3.6: JavaScript Project Structure analysis

### 3.2.2.3 Code Style

In this section the Code Style of the JavaScript file will be described.

#### **Definition of JavaScript code style**

A JavaScript code style is a set of guidelines of how all the developers in a project should write JavaScript code. It includes how to indent, the use of commas, quotes or how to name variables among other features [57].

#### **Importance of following a JavaScript code style**

The importance of following a JavaScript code style is great. It enhances the readability and consistency of the code improving the efficiency of the developers when they are scanning the code searching for a specific function or variable.

Additionally, it eases the incorporation to the project of new developers without confusing the rest of the developers with the way the new incomer writes code. Moreover, this situation could lead to a misunderstanding between the developers resulting into bugs and errors in the final product.

#### **JavaScript code style**

The main issue according to the JavaScript code style of the JavaScript file is the fact that no consistency is followed according to the naming of variables. In some cases the *Camel Case* is used while in other situations the words are separated just by a slash.

Furthermore, if the code inside the JavaScript file is analyzed it is seen that all the comments are in Spanish. As it was mentioned in a previous section with file naming, it is recommended to use English when writing code in order to let developers from other nationalities to join to the project.

Another issue which is found when the code is read in detail is the wide amount of hardcoded String comparisons. There are several String comparisons where the compared String is a constant value defined by the developer or by the environment and this value is hardcoded inside the methods. This is a bad practise due to the fact that in the case that any of these values needs to be changed, all the occurrences of this comparison need to be changed as well. The possible outcome of this practise is the appearance of bugs and errors due to incorrect comparisons and assignment of constant variables.

Finally, the last important issue according to code style in the JavaScript file is the fact that there are global variables which are declared between the methods and not in the header of the file.

It is a good and widely used practise in several programming languages to define all the global variables in the header of the file in order to increase the readability and intuitively

of the written code. In the case that this guideline is not followed, as it happens in this project, it is much more difficult to follow the execution flow of the methods in an proper way and to read the methods correctly understanding their behaviour.

### Summary

In order to outline the main issues of the previous analysis, the following table is going to be used as a reference.

JavaScript Code Style analysis		
Section	Analysis	Effect
Variable Naming	Inconsistency of variables' name	High possibility of making mistakes referencing a variable  Inefficient development
Variable Declaring	Global variables declared between methods	High possibility of making mistakes referencing a variable  Inefficient development
Hardcoded String Variables	Several String comparisons are performed using hardcoded Strings in the middle of the method	Inefficient development  Inefficient modification of the code
Comments Language	All the comments are written in Spanish	Difficulty finding international developers to join the project

Table 3.7: JavaScript Code Style analysis

### 3.2.3. PHP analysis

According to the PHP analysis, the used Framework, the PHP Project Structure and the PHP Code Style will be assessed.

#### 3.2.3.1 PHP Framework

In this section the Framework used in order to develop the PHP code belonging to Sorting Hat will be analyzed and the correctness of its decision will be determine.

## Definition of a PHP Framework

A PHP Framework is a set of files or libraries which allow to the developers to create and implement a Web Application in a more agile way. Furthermore, most of the PHP Frameworks follow the Model View Controller approach, which has been explained in detail in the previous sections [58].

## Definition of a proper PHP Framework

A proper PHP Framework should ease the use of build up functions in order to avoid the re-implementation of functions which have been already created. Consequently, a main feature of a great PHP Framework is the reduction of the amount of code needed to be written in a program to achieve the desired functionality.

Furthermore, it should provide Database support as well as a Speed-up Custom Web Application Development. In addition to this, all the features corresponding to a proper JavaScript Library, such as *Community*, *Stability* and *Team*, are also important for a proper PHP Framework [59].

## Laravel

Laravel is a PHP Framework which supports and endorse the rapid development of Web Application as well as writing expressive and beautiful code. Thanks to its rapidly popularization, several platforms and extensions of the core Laravel Framework has been created under the name of *The Laravel Ecosystem* [60].

Furthermore, the fame and reputation of Laravel has increased in the recent years up to the point of been considered the best PHP Framework which is currently available [61]. Laravel's popularity is based on the great amount and quality of the features which it provides to the developers. The main components of Laravel are the following ones [62].

- Authorization: Laravel provides to the developers with a simple way to implement authentication techniques. Authentication is a key feature which almost all Web Application include. Consequently, this feature could be helpful to a great amount of developers.
- Artisan: Artisan is a Laravel built in tool which its main purpose is interacting between the developer and the Framework through the command line in order to perform configuration and installation processes more agile.
- MVC: As it has been explained previously in this section, most of the PHP Frameworks are based in the Model View Controller approach, and Laravel is not an exception. As a consequence, the development of PHP code is cleaner and more intuitive improving the overall quality of the project.

- Security: Nowadays security is a key concern for any information technology application or infrastructure. Consequently, the correct managing of passwords and the sanitation of user inputs is a requisite for any Web Application and therefore to any PHP Framework. Laravel provides all these features enhancing the security of the Web Application and simplifying its implementation.
- Responsible User Interface: According to the User Interface, Laravel provides features in order to ease the implementation of Responsible User Interfaces. Taking into account the great amount of users which access to the Internet through a smart-phone, this is a crucial feature which any Web Application should implement.

For all these reasons, the choice of using Laravel as the PHP Framework for this project is correct.

Nevertheless, some of the features which Laravel provides are not fully implemented into the project. Especially, the implementation of a *Responsible User Interface* is not correctly done and it will be described in detail in the following section regarding to the Analysis of the User Interface.

### Summary

Before summarizing the main issues detected during the analysis, it should be outlined the correctness of the choice of using Laravel as the PHP Framework for Sorting Hat.

In order to outline the main issues of the previous analysis, the following table is going to be used as a reference.

PHP Framework analysis		
Section	Analysis	Effect
Laravel	Laravel feature for the correct implementation of Responsible User Interfaces is not fully used	Inconsistency and issues while displaying Sorting Hat in different devices

Table 3.8: PHP Framework analysis

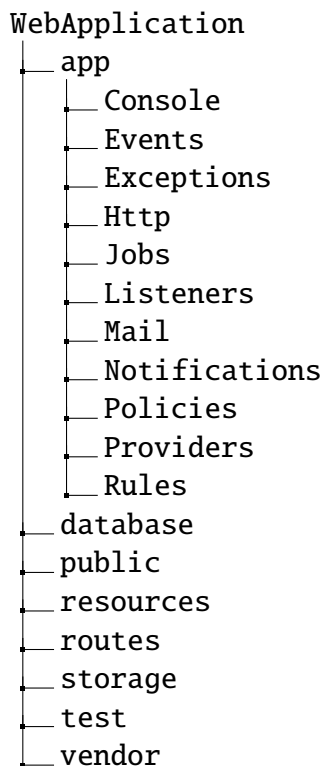
### 3.2.3.2 PHP Project Structure

In this section of the PHP analysis, the PHP project structure will be assessed. However, as Laravel is the PHP Framework which is being used in Sorting Hat, the PHP Project Structure is going to be focused in Laravel.

## Definition of a proper PHP's project structure

A proper PHP's project structure is the structure which eases the development of new functionalities as well as it facilitates the modification of already implemented code. Consequently, a clean and well organized project structure will fulfill these requirements.

Nevertheless, as Laravel is implemented in Sorting Hat, a proper Laravel Project Structure is the one that follows with Laravel Application Directory Structure. This structure is similar to the following one [63].



## Importance of a proper PHP's project structure

In the case that a proper PHP's project structure is followed, the development and debugging process will be much more efficient than in the case that no clear project structure is followed.

Consequently, if efficiency is desired while developing a Web Application, a proper project structure is needed to be defined and accomplished in the project.

## Project structure analysis

The analysis of the PHP project structure, as well as in the previous sections, is going to be divided into two different sections.

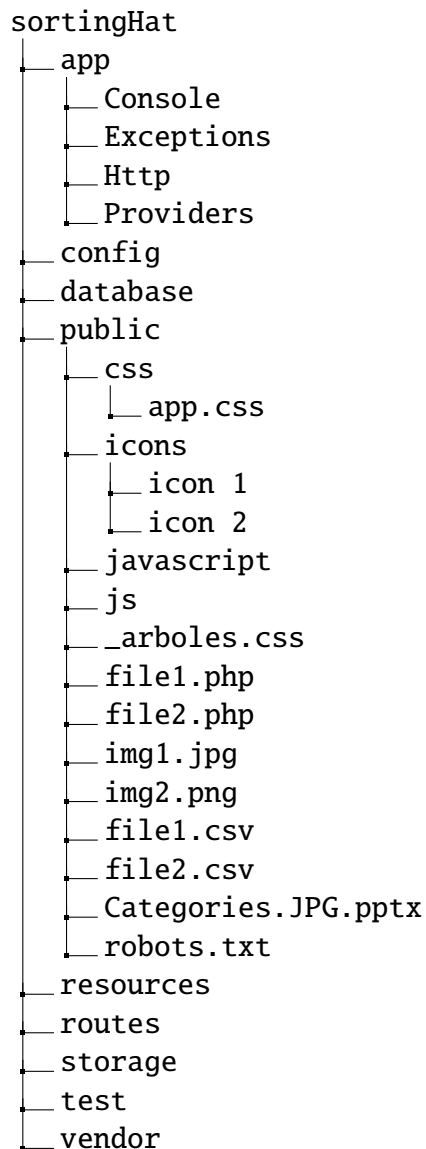
- *File Structure*: the file and folder organization followed in Sorting Hat will be assessed in terms of usability, efficiency and correctness according to Laravel Appli-

cation Directory Structure.

- *Code Structure*: the code of the PHP and Laravel files will be determined in order to determine if they have been correctly organized and written.

## File Structure

According to the File Structure of the PHP code it is similar to the following diagram.



If the previous diagram is compared with the one displayed in the previous section called *"Definition of a proper PHP's project structure"*, it can be discerned several differences.

Firstly, at the folder *"app"* there are several folder which does not exist in Sorting Hat. These folder are *"Events"*, *"Jobs"*, *"Listeners"*, *"Mail"*, *"Notifications"*, *"Policies"*

and "Rules". However, as these functionalities are not implemented into Sorting Hat there is no issue with its absence.

Furthermore, if the folder *public* is analyzed in detail it can be discerned the main problem of the project structure. This is the folder which is supposed to store all the JavaScript, CSS, images and PHP files.

The main issue about the File Structure is the fact that inside *public* there are *php*, *css*, *png*, *jpg*, *pptx* and *txt* files mixed with each other. As a consequence of the absence of organization inside the folder *public* is the fact that the complexity of this folder and the probability of encountering errors while the development is very high.

## Code Structure

According to the Code Structure of the PHP files, the PHP files are located inside the folder *public* and *resource/views*. The files inside the first folder are common PHP files while the ones located inside the second folder are *.blade.php*. This type of files are a special type of file from Laravel which is similar to a HTML template.

According to the files belonging to *public*, there are several issues in these files. The main issue is the fact that there is no function in all the five files getting to the point that the file *querystats.php* has 450 lines of code with no refracting. Consequently, it is almost impossible to follow the execution flow and the understand the behaviour of the PHP files.

Furthermore, inside the PHP files located in *public* there is no descriptive comments explaining what is being doing. If the absence of comments is join with the nonexistence refactoring, the efficiency of modifying the files or developing further functionalities is negligible.

Regarding to the *.blade.php* files located inside *resource/views* the situation is different. These kind of files are a special type of files of Laravel which behaveds similar to HTML templates. However, in these files are analyzed there is CSS, HTML and PHP code all together in these files with no clean organization.

Therefore, reading the *.blade.php* files is almost impossible due to the great amount of different types of code increasing the complexity of the code to very high levels.

## Summary

In order to outline the main issues of the previous analysis, the following table is going to be used as a reference.



PHP Project Structure analysis		
Section	Analysis	Effect
File Structure	Mixing of several types of files inside the folder <i>public</i> with no organization	Inefficient development Ineffective debugging process
	No refactoring of the files	Understanding the execution flow and behaviour of the PHP files is too difficult
Code Structure	Absence of descriptive comments	Inefficient development
	No organization inside <i>.blade.php</i> files	Ineffective debugging process

Table 3.9: PHP Project Structure analysis

### 3.2.3.3 PHP Code Style

The aim of this section is analyzing the Code Style used in the PHP code of Sorting Hat

#### Definition of a proper PHP Code Style

A proper PHP Code Style is a set of rules which aim to maintain a style of coding in order to increase the efficiency of a group of developers working in the same project. Furthermore, a proper PHP Code Style should also produce the sense that the code is formal and software industry oriented [64].

#### Importance of a proper PHP Code Style

PHP is one of the programming languages which follows the fewest rules in order to establish its Code Style. Consequently, due to the fact that developers has more freedom while developing PHP code it is more probable that it becomes messy and very different from one developer to other.

The main reason why maintaining a proper PHP Code Style is straightforward. If several developers are working in the same project it is crucial that the code that they create are understandable by the rest of them.

In addition to this, when a consistency in the code is followed, scanning the code for a specific variable or method is faster. Consequently, a proper PHP Code Style could improve the readability of the code and the overall efficiency of the developers [65].

## **PHP Code Style**

According to the Code Style of the PHP code of Sorting Hat, in this case the variable naming is consistent. However, the PHP files has an issue regarding to the Code Style which in the previous analysis did not appear. This issue is the consistency of line breaks and indenting.

To start with, there is no consistency of the line breaks used for separating blocks of code inside the same file. In some cases one line break is used, and other times two or three. If the code is examined in order to determine if this difference separate special execution blocks from each other it is showed that there is no reason for this practise, it is just coding inconsistency.

The main consequence which this practice could have is the misunderstanding of the code by the developers which think that the difference in line breaks mean the separation of blocks of code from different functionality.

Moreover, if the lines of code are analyzed they experience a similar problem. There are some comments which are separated by a line break with the code to which they refer while in other cases no line break is used. This situation increases the complexity of the code and increase the difficulty to understand it.

Furthermore, regarding to the indenting the situation is similar to the previous one. In some cases there is no left indentation and other times the normal indentation, four spaces, is doubled with no reason.

As in the previous case, as well as being much difficult to read and understand the code, the efficiency of the developers if they are using this code would be much lower than if a consistency is followed.

Another issue which was noticed while accomplishing the analysis of the PHP Code Style is the fact that there is no consistency with the format used for the control structures. The use of spaces to separate the parenthesis, the braces and the variables is arbitrary and no consistency is used. This situation makes more demanding the reading of the PHP code.

Finally, as in the analysis of the previous sections, the comments in the PHP code are in Spanish, Therefore, if a non Spanish speaker desires to join the project in order to improve it the developer would not be able of correctly understand the code.

## **Summary**

In order to outline the main issues of the previous analysis, the following table is going to be used as a reference.

PHP Code Style analysis		
Section	Analysis	Effect
Line Breaks	Inconsistency of the amount of line breaks used for separate blocks of code	Inefficient development
	Inconsistency of the amount of line breaks used for separate the comments	Ineffective debugging process
Indenting	Inconsistency in the amount of left indenting used	Inefficient development
		Ineffective debugging process
Control Structures	No consistency in the use of spaces in the control structures	Inefficient development
		Ineffective debugging process
Comments Language	All the comments are written in Spanish	Difficulty finding international developers to join the project

Table 3.10: PHP Code Style analysis

### 3.2.4. User Interface analysis

In this section of the analysis, several elements of the User Interface such as the Framework used, CSS code, HTML code, the Responsiveness and General Inconsistencies of the User Interface are going to be analyzed.

#### 3.2.4.1 User Interface Framework

In this part of the analysis, the User Interface Framework used in Sorting Hat is going to be assessed.

#### Definition of a proper User Interface Framework

A proper User Interface Framework helps to the developers to create beautiful and fully responsive User Interfaces. Furthermore, a proper Framework does not require highly technical skills and it reduces the time needed for the development of the User Interface.

## Importance of using a proper User Interface Framework

The use of a proper User Interface is crucial for the development of a efficient, beautiful and responsive User Interface. If a User Interface Framework is not used or a poor quality Framework is used, the amount of time needed to develop the User Interface will be much greater than if a proper Framework would have been chosen [66].

### Bootstrap

The User Interface Framework used in Sorting Hat is Bootstrap. Bootstrap is one of the most popular Frameworks due to its features to build responsive Web Applications and the great support from the developers which it has [67].

In Sorting Hat, the main issue according to Bootstrap is its implementation in order to achieve a responsive User Interface. As it is going to be described in the following sections, the responsiveness of Sorting Hat is not correctly implemented.

Consequently, as it was Bootstrap the Framework which was used in order to achieve this responsiveness, its implementation is performed incorrectly. Its result is the decrease of the quality of the User Experience while accessing to Sorting Hat.

### Summary

In order to outline the main issues of the previous analysis, the following table is going to be used as a reference.

User Interface Framework		
Section	Analysis	Effect
Bootstrap	Incorrect implementation of a responsive User Interface	Downgrade of the User Experience

Table 3.11: User Interface Framework

### 3.2.4.2 CSS

#### CSS Project Structure

According to the Project Structure of the CSS code, its correctness is going to be determined through the following analysis.

## Definition of a proper CSS Project Structure

A proper CSS Project Structure is the one which enhances readability as well as it promotes the development of modular and reusable CSS code.

## Importance of a proper CSS Project Structure

CSS stands for Cascading Style Sheets. The core feature of CSS is related to the concept of "*Cascading*" which could be briefly described as the order of the CSS rules in order to display the User Interface components [68].

However, one of the main issues regarding to the CSS "*Cascading*" is the fact that it is very probable that its Cascade flow gets confussing if no standard or rules are followed in order to maintain a consistency. As a consequence, establishing a proper CSS Project Structure is the first layer and basis of the consistency of the CSS files.

## Project Structure analysis

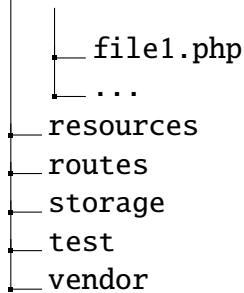
As in the previous sections, the analysis of the Project Structure of the CSS files is going to be divided into two sections.

1. *File Structure*: the organization according to the files and the folders will be analyzed in order to improve the usability and efficiency.
2. *Code Structure*: the CSS code will be analyzed in order to assess if it is correctly organized and refactored so as to promote the reuse of CSS code.

## File Structure

According to the File Structure of the CSS files of Sorting Hat, its structure is similar as in the following diagram.

```
sortingHat
├── app
├── config
├── database
├── public
│   ├── css
│   │   ├── app.css
│   │   ├── icons
│   │   ├── javascript
│   │   ├── js
│   │   └── _arboles.css
```



The main issue related to the File Structure of the CSS files has been noticed in the previous analysis of the PHP files. This issue is the fact that inside the *public* folder there are files of every type with no organization. Among these documents there are CSS files.

Furthermore, if this folder is analyzed in detail, it can be seen that there is a folder called "*css*" which stores the file "*app.css*". This file is the file needed in order to use Bootstrap.

As a consequence, there are CSS files in two different locations increasing the complexity of the project as well as the difficulty of understanding its code.

### Code Structure

Regarding to the Code Structure of the CSS files, there are two locations where there is CSS code. One of them is the previously mentioned in the File Structure section, which is in the folder *public*. The other place is inside the *blade* files where the CSS code is written inside the HTML code. This is not the expected location for CSS code.

With respect to the *\_arboles.css* file, which is the CSS file located inside *public*, the more obvious issue is the amount of lines of code commented. A great amount of the lines of code inside the file are commented which greatly increase the size of the file and confuse the developers which are reading the code.

In addition to this, another problem which this file presents is the unnecessary rewrite of code. There is a comment inside the code which explicitly says "*DUPLICADO*", which means in English "*duplicated*".

This situation shows one of the main problems which CSS code, and in general all the code, in Sorting Hat faces. This problem is the lack of reusable and modular code. Rewriting code is a well known bad practise in all of the programming languages and it should be avoided. This situation increases enormously the difficulty of understanding the code of a project.

Furthermore, as in the previous sections of the analysis, there is no descriptive comments in the code. It is true that in this case there are more comments than in other cases previously described. However, they are insufficient and they are confusing.

According to the CSS files inside the HTML code from the *blade* documents, apart from the already described issues which can be applied to these files too, there is a basic problem. It is known that no CSS file should be coded inside the HTML code itself. The CSS should be written in different files in order to be imported in the header of an HTML file, but it should never be written inside the HTML itself.

The consequence of this issue is the increasing of the complexity and illegibility of both the CSS and the HTML code.

### Summary

In order to outline the main issues of the previous analysis, the following table is going to be used as a reference.

CSS Project Structure analysis		
Section	Analysis	Effect
File Structure	Mixing of CSS files with other type of files inside the folder <i>public</i> with no organization	Inefficient development
	CSS files outside the " <i>css</i> " folder where it is their expected place to be located	Ineffective debugging process
Code Structure	Unnecessary lines of code commented	Understanding the execution flow and behaviour of the CSS files is remarkably difficult
	Duplicated code	
	No refactoring of the files	Inefficient development Ineffective debugging process
	Absence of descriptive comments	
	CSS code written inside the HTML code	

Table 3.12: CSS Project Structure analysis

### CSS Code Style

In this section the Code Style of the CSS code from Sorting Hat is going to be analyzed in order to determine its correctness.

## **Definition of a proper CSS Code Style**

A proper CSS Code Style is the one which contains the best practices to write CSS code of high quality. This Code Style should be followed by all the developers who are engaged in the same project [69].

## **Importance of a proper CSS Code Style**

It is common that each developer has his own way of writing CSS code. However, if several developers who works in the same project write CSS code each in a different way, it is probable that the resulting code is confusing and difficult to read. In order to avoid this situation, a CSS Code Style should be used in order to promote the consistency of all the CSS code, regardless of who write it [69].

## **CSS Code Style**

According to the Code Style of the CSS code, the first issue noticed is related to the line breaks. There is a difference in the amount of line breaks used to separate different CSS elements. This difference goes from no line break to two line breaks, with no reason behind the use of one or another.

Furthermore, a problem with regards the indenting was discovered too. There is a inconsistency about the indenting used for the CSS code. In the CSS code from Sorting Hat, it is used for indenting two spaces, four spaces or one tab. As a consequence, apart from decreasing the clarity of the code, it could lead to errors due to the combination of tabs and spaces.

Once that the code has been analyzed, it is observed the absence of the use of short-hand properties. These properties are a way of collapsing several related properties, such as the rules related to the font, in order to reduce the amount of code written as well as enhancing the readability of the code.

Another issue which was noticed was related to Capitalization. It is true that the consistency of the Capitalization, in this case the use of lower case, is sufficient. However, there are still some cases in which this consistency is not maintain, such as definition of colors.

Finally, the last inconsistency discovered regarding to the CSS Code Style is the declaration block separation. The separation between the last selector and the opening brace is, as it happened with the line breaks, almost random. Sometimes no space is used and other times one space or a line break is used. The result of this practice is the increasing of the difficulty for reading the code as well as the increasing of the lines of code.



## Summary

In order to outline the main issues of the previous analysis, the following table is going to be used as a reference.

CSS Code Style analysis		
Section	Analysis	Effect
Line Breaks	Inconsistency of the amount of line breaks used for separate blocks of CSS code	Increase of the file size
		Inefficient development
		Ineffective debugging process
Indenting	Inconsistency in the amount of left indenting used	Problems regarding the understanding of the scope of the code
		Inefficient development
		Ineffective debugging process
Shorthand Properties	Absence of Shorthand Properties for some rules	Inefficient development
		Ineffective debugging process
Capitalization	Inconsistency of the use of lower case	Increase of the file size
		Inefficient development
		Ineffective debugging process
Declaration Block Separation	Inconsistency of the separation between the last selector and the opening brace	Increase of the file size
		Inefficient development
		Ineffective debugging process

Table 3.13: CSS Code Style analysis

### 3.2.4.3 HTML

Several features of the HTML of Sorting Hat are going to be analyzed. As a consequence of the use of Laravel, the HTML is very related to the *blade* files. Therefore, the HTML

analysis is going to be related to Laravel and to the *blade* files.

## HTML Project Structure

The Project Structure of the HTML code is going to be assessed in order to determine its quality and efficiency.

### Definition of a proper HTML Project Structure

A proper HTML Project Structure in a Laravel environment is the one which encourage the reusing of components, views and code in order to produce quality code.

### Importance of a proper HTML Project Structure

Following the reasoning behind the definition of a proper HTML Project Structure, if a proper HTML Project Structure is followed the quality and readability of the HTML and blade files will increase. Furthermore, if re-usability and modularity of HTML code is desire in the project, which it should be, establishing and maintaining a proper HTML Project Structure is a core feature to achieve it.

### Project Structure analysis

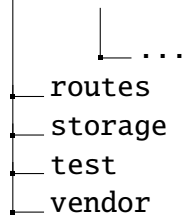
The analysis of the Project Structure of the HTML files is going to be accomplished in two different sections.

1. *File Structure*: the files and the folders containing HTML code will be assessed so as to determine its efficiency
2. *Code Structure*: the HTML code as well as the Laravel code inside the *blade* files related to the HTML will be analyzed.

### File Structure

According to the HTML File Structure, its structure is similar to the following diagram.

```
sortingHat
├── app
├── config
├── database
├── public
├── resources
│   ├── assets
│   ├── lang
│   └── views
│       ├── view1.blade.php
│       └── view2.blade.php
```



First of all, before performing the analysis of the HTML File Structure, some concepts related to *blade* files should be explained in order to gain a better understanding of the objective of this analysis.

In Laravel, the HTML code is written into the *blade* files. These files get the name of *views* and they are located inside the folder *resources/views*. One of the key features of these files is the possibility of defining *Layouts*, which basically is an HTML template which can be reused by other views [70].

It is known that most of the Web Applications have a base User Interface Layout which is slightly different for each page. Sorting Hat User Interface could be divided into two different layouts: the log in layout and the layout of the home page with the top bar.

However, in Sorting Hat there is no implementation of Laravel Layouts. As a consequence, the main issue according the File Structure is that there is no specific folder for creating and storing these Layouts which could be used by other *blade* files reusing a great amount of code.

## Code Structure

Following the reasoning behind the previous analysis of File Structure, the main issue according to the HTML Code Structure is the fact that the HTML code has not been divided into Layouts. Consequently, there is no HTML code reusing in Sorting Hat. This consequence highly increase the complexity and the number of lines of code needed.

Furthermore, as it has stated in the previous analysis, all the PHP, JavaScripts and CSS code which is inside the HTML code should be refactored to new files. The result of the actual situation is the increase of the difficulty to understand the behaviour of the HTML code, which could possibly lead to errors in future developments.

The next problem found during the analysis is related to the scripts too. However, in this case these scripts are imported from external files, but the place to import them is not the optimal. These scripts are loaded in the header of the HTML. Nevertheless, if the scripts are loaded at the header of the HTML the rendering of the User Interface could be delayed.

Moreover, another issue is the fact that many of these scripts are loaded in the pages and views in which they are not being used. One example are the scripts needed to display the charts of the *dashboard* page. These scripts are loaded in pages not related to this

functionality, such as the *editProfile* page. The obvious consequence of this situation is the slowing of the load of the page due to the extra data been transferred to the user.

Finally, the last issue regarding to the Code Structure is the absence of descriptive comments explaining the different components of the HTML.

## Summary

In order to outline the main issues of the previous analysis, the following table is going to be used as a reference.

HTML Project Structure analysis		
Section	Analysis	Effect
File Structure	There is no folder for storing Laravel Layouts	Difficulty for implementing Layouts in the <i>blade</i> files
	Layouts are not created	
Code Structure	Existence of PHP, JavaScript and CSS code inside the HTML code	Increase of the file size
	The scripts are loaded on the header of the HTML	Delay of the display of the User Interface
	Scripts loaded in pages where they are not beign used	Inefficient development Inefficient upgrading
	Absence of descriptive comments	

Table 3.14: HTML Project Structure analysis

## HTML Code style

The Code Style of the HTML code of Sorting Hat is going to be analyzed in the following sections.

### Definition of a proper HTML Code Style

A proper HTML Code Style is a set of style rules which, if they are followed by all the developers in a project, the quality and readability of the code will drastically increase [71].

## Importance of a proper HTML Code Style

HTML, as well as CSS, is one of the languages in which Code Style is less used. Consequently, if quality of code in terms of consistency and ease to understanding the code is desired, a proper HTML Code Style should be implemented.

### HTML Code Style

In this analysis of the HTML Code Style, the Google HTML Style Guide is going to be used as a reference [72].

The first problem related to the HTML Code Style is easily perceived. This problem is the great inconsistency of the use of line breaks along the code. In some cases, these line breaks are correctly used in order to separate the different components. However, sometimes it is only used one line break while in other cases two or three line breaks are used. This situation is noticeable in the body of the HTML as well as in the header.

According to the indenting of the code, even though the HTML is much better indented than the previous code analyzed there are still some situations of inconsistency related to the indenting. In addition to this, there is an inconsistency of the use of spaces or tabs in order to indent the code.

Another issue which was discovered during the analysis is regarding the images. Most of the images added into the HTML has the attribute *alt* with a standard name, such as *"icon"*, it is empty or it does not exist. This attribute is very important in the case that the images do not render because is what the users will see in the User Interface in the place of the images. If no meaningful *alt* is added, in the situation that the user cannot display the images he will have no information about them, damaging the perception of Sorting Hat by the user.

Since HTML5 the attribute *"type"* is set as default to *"text/css"* and *"text/javascript"*. Consequently, it is not necessary to specify the *type* attribute if the asset to be imported is a JavaScript or CSS file. Nevertheless, this is done in the HTML code from Sorting Hat.

Finally, the last issue which was determine by the analysis is the fact that when attributes values are being quoting in the HTML single quotation marks *' '* are being used. It is known that double quotation marks *" "* are recommended to be used when it is possible.

### Summary

In order to outline the main issues of the previous analysis, the following table is going to be used as a reference.

HTML Code Style analysis		
Section	Analysis	Effect
Line Breaks	Inconsistency of the amount of line breaks used for separate blocks of HTML code	Increase of the file size
		Inefficient development
		Ineffective debugging process
Indenting	Inconsistency in the amount of left indenting used  Inconsistency in the use of spaces or tabs for the indenting	Problems regarding the understanding of the scope of the code
		Inefficient development
		Ineffective debugging process
Images	No meaningful <i>alt</i> attribute is added to the images	In the case that the images are not loaded, the quality of the User Experience of the user will decrease
"Type" Attribute	Specify the <i>"type"</i> attribute when importing a CSS or JavaScript file	Increase of the file size
Quotation Marks	Single quotation marks used for quoting attribute values	Decrease the readability

Table 3.15: HTML Code Style analysis

#### 3.2.4.4 Responsiveness

The Responsiveness of Sorting Hat is going to be analyzed in the following sections.

##### Definition of a Responsive Web Application

A Responsive Web Application is a Web Application which shows the same main content to all the devices, but depending of the space available it displays a different version of the User Interface [73].

##### Importance of implementing a Responsive Web Application

The implementation of a Responsive Web Application has several advantages. One of the main advantages of a Responsive Web Application is the fact that the amount of devices which can access to the Web Application successfully will highly increases.

Furthermore, another important benefit of implementing a Responsive Web Application is related to the ranking on Google search. Since 2015 Google Ranking Algorithm takes into account if a Web Application implements a Responsive User Interface or not. If it is not implemented, the ranking of the Web Application will decrease. Consequently, it is crucial to any Web Application to implement a proper Responsive User Interface [74]. [74]

## **Responsiveness analysis**

The analysis of the Responsiveness of Sorting Hat is going to be divided into two sections.

- Computer Responsiveness: Sorting Hat is going to be evaluated with different computer screen resolutions in order to determine if the User Interface is Responsive or not.
- Smartphone Responsiveness: Sorting Hat User Interface is going to be analyzed to determine if it is "*mobile friendly*".

### **Computer Responsiveness**

The analysis of the Computer Responsiveness is going to be performed showing a comparison between the User Interface displayed in a large computer screen of 1920x959 and a smaller computer screen of 1440x779.

Three different pages are going to be analyzed in the following sections. These pages are the ones which have the most important inconsistencies related to responsiveness.

#### **Log in**

In the Log in page the main inconsistency is regarding to the background image. As it is showed in the following images, the image in the smaller screen fit all the screen. However, in the larger screen the background fit only a part of the screen been the rest of it filled with a gray background colour.

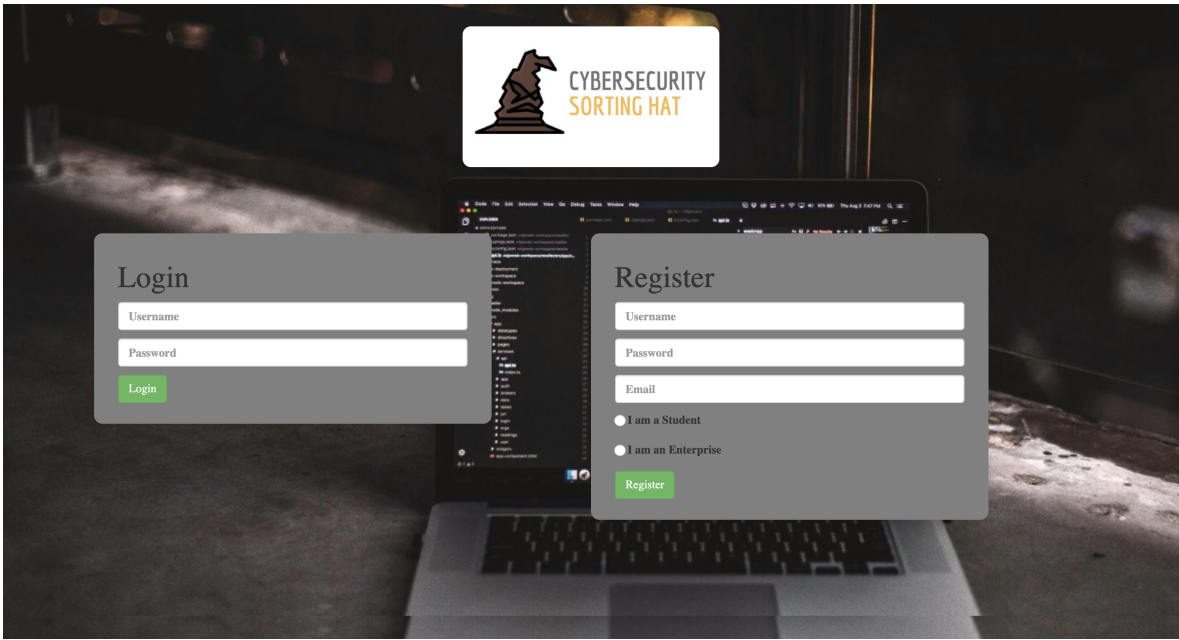


Fig. 3.14. Log in page in a small computer screen

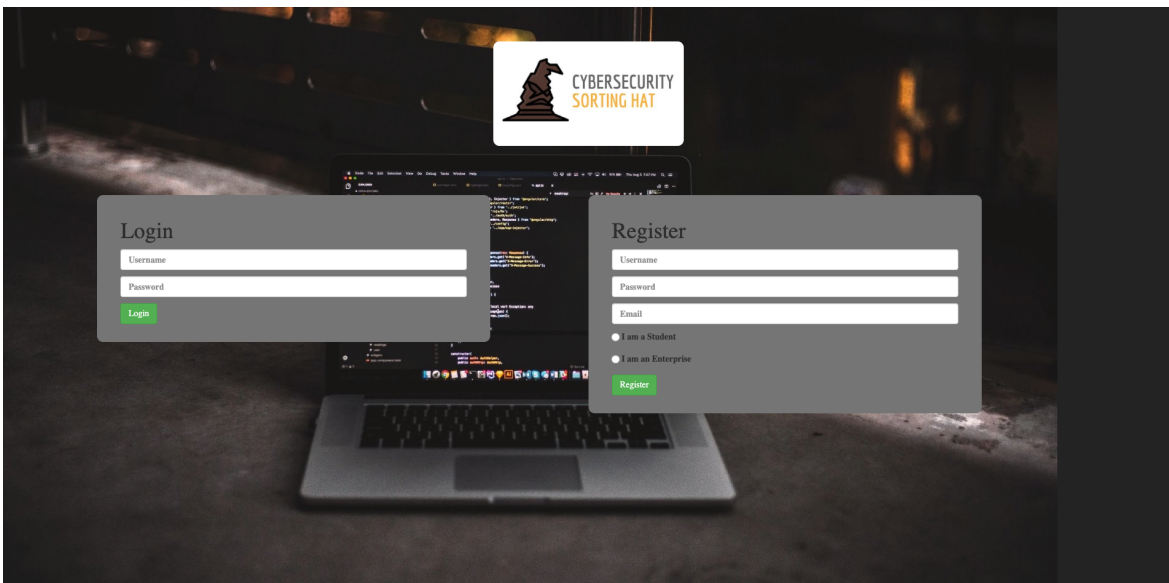


Fig. 3.15. Log in page in a large computer screen

## Home

The responsiveness inconsistency in the Home page is related to the images displayed. The problem in this case is that in the large screen all the picture is correctly displayed but in the smaller screen it is abruptly cut in order to fit the screen



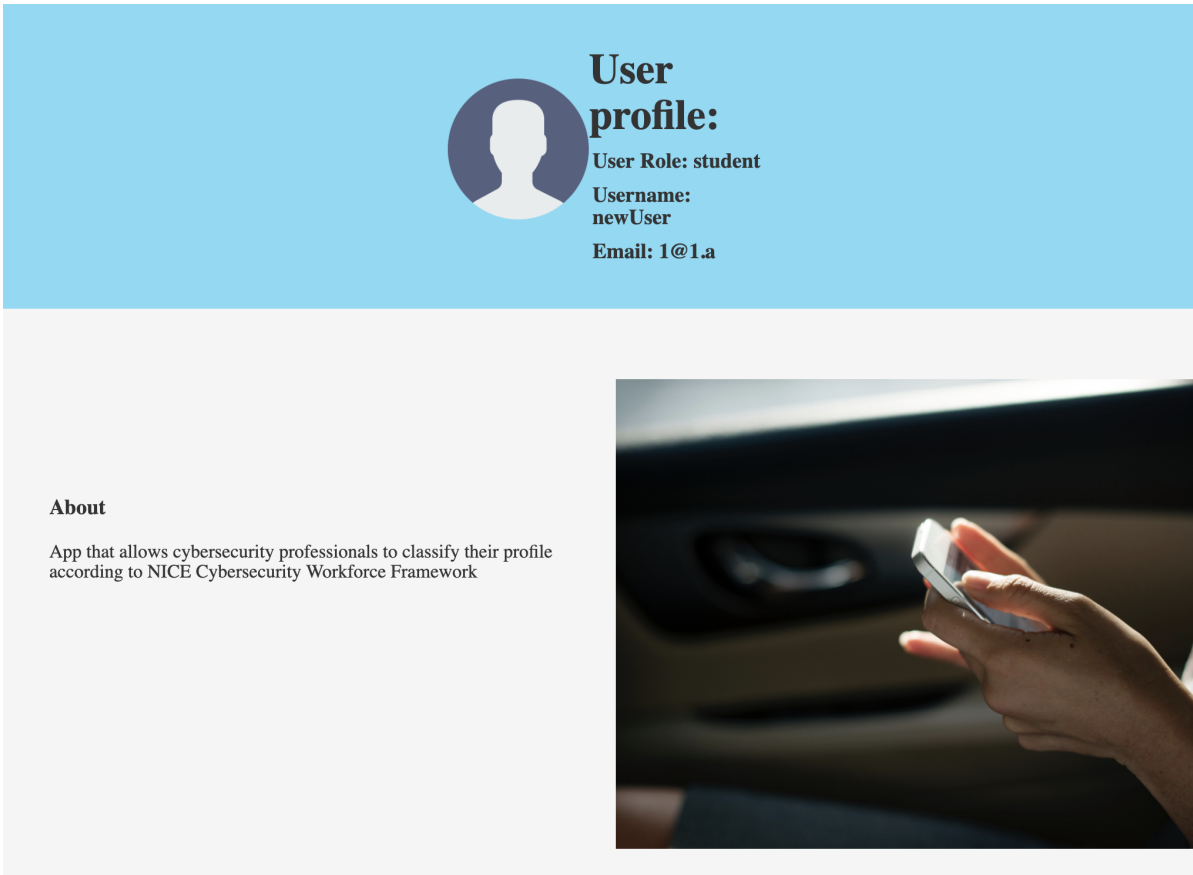


Fig. 3.16. Home page in a small computer screen

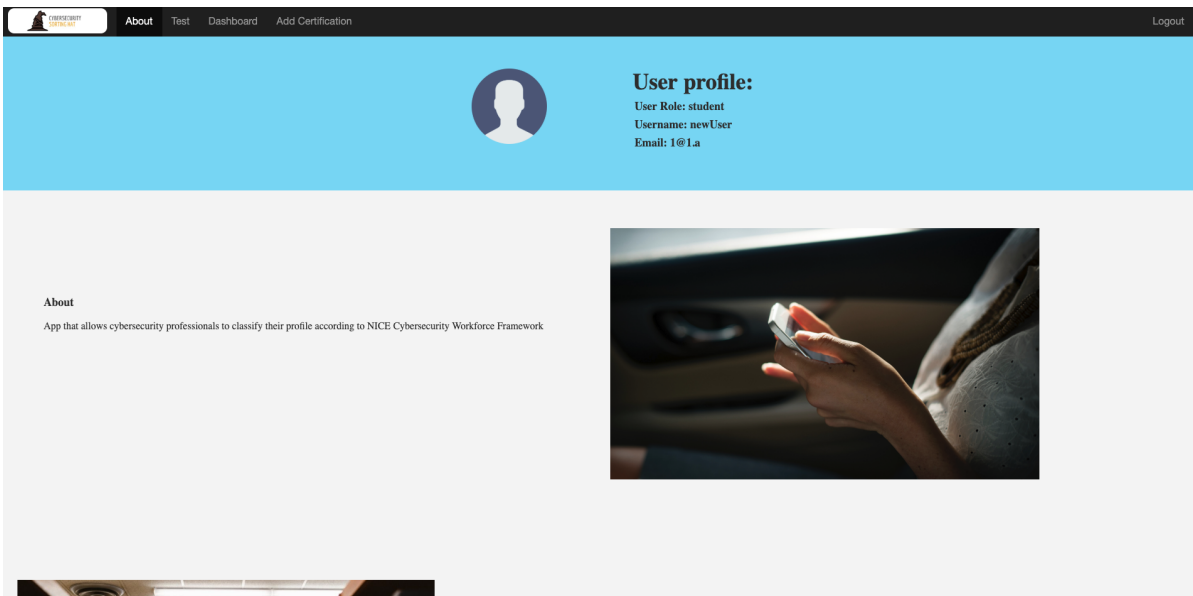


Fig. 3.17. Home page in a large computer screen

## Account

In the Account page the issue is regarding to the lower bar of the screen. The issue is that this bar is not set to stay in the bottom of the screen. Nevertheless, depending on the size of the screen it is located in one place or in another.

In this case, in the smaller screen the bar is closer to the bottom of the screen but in the larger screen it has moved upwards.

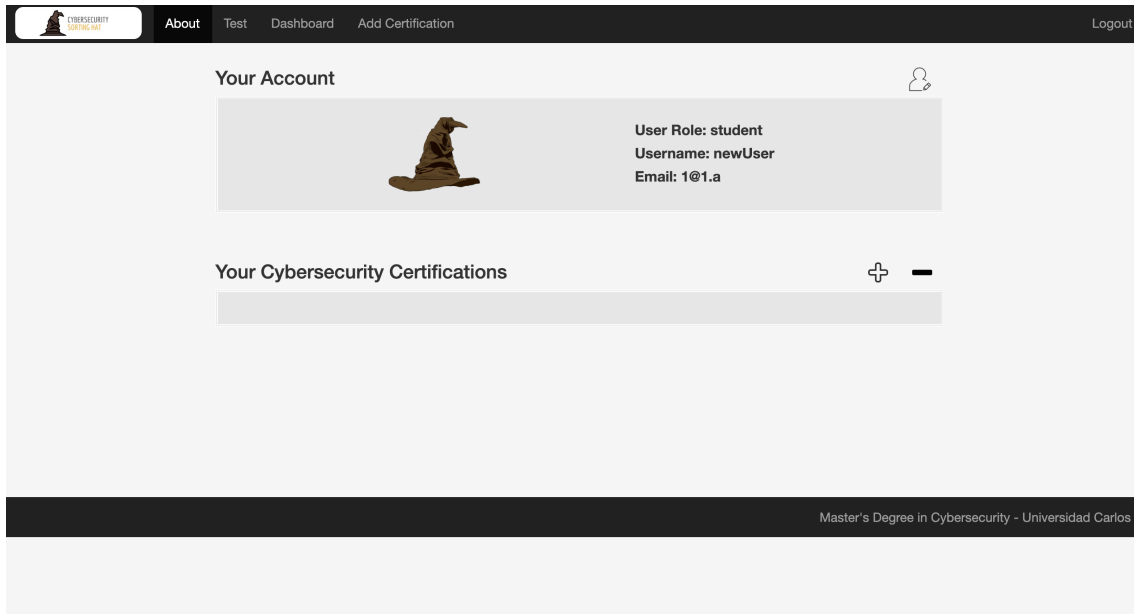


Fig. 3.18. Account page in a small computer screen

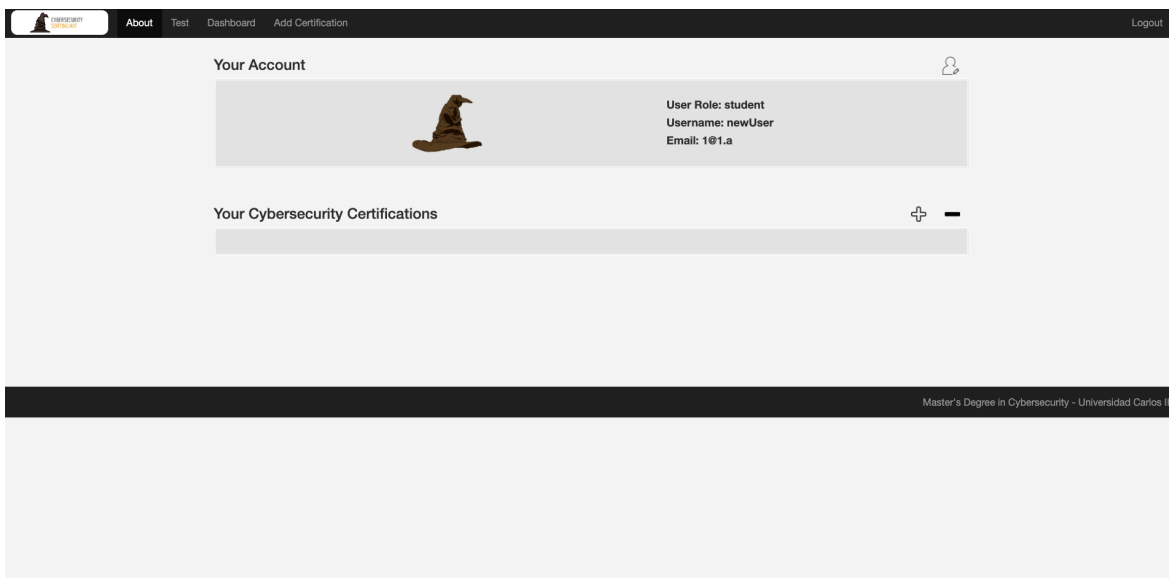


Fig. 3.19. Account page in a large computer screen

## **Smartphone Responsiveness**

The analysis of the Smartphone Responsiveness is going to be performed in the screen of an iPhone 5 of 320x568. The pages analyzed are going to be the same as in the previous section in order to show the differences between both User Interfaces so as to determine the possible issues in them.

### **Log in**

According to the Log in page in mobile devices, the issue in this case is related to the log in and register boxes and to the background. Even though the log in boxes are functional, they are not optimal for mobile devices. These boxes have not been changed for mobile devices, they have only been placed one below the other one due to the lack of space on the side.

Even though this could be a valid option, from the point of view of the User Experience it is not. There is no space left in the sides of the boxes, the boxes are one next to the other and the bottom half of the screen is empty.

Regarding to the background, it does not fill the last half of the screen which is below the register box harming even more the quality of the User Experience at the Log in page.

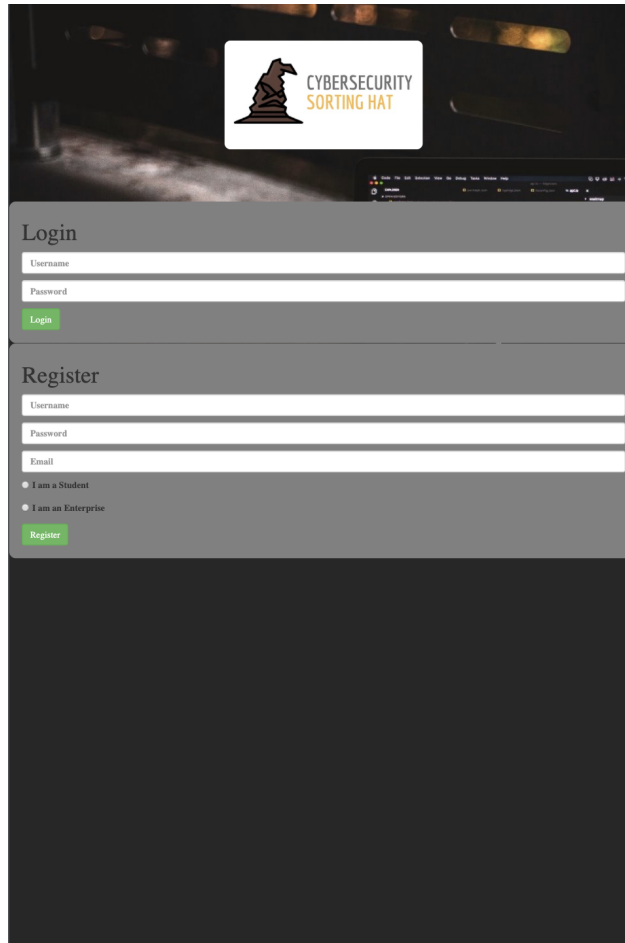


Fig. 3.20. Log in page in a mobile phone screen

## Home

In the case of the Home page, the issue for the mobile devices users is not related to the images but to the text associated to them. In the following screenshots it is displayed a clear example of this situation. The text, which is displayed in rows in the right part of the screen, should be next to the image above it, as it happens in the computer version of this page.

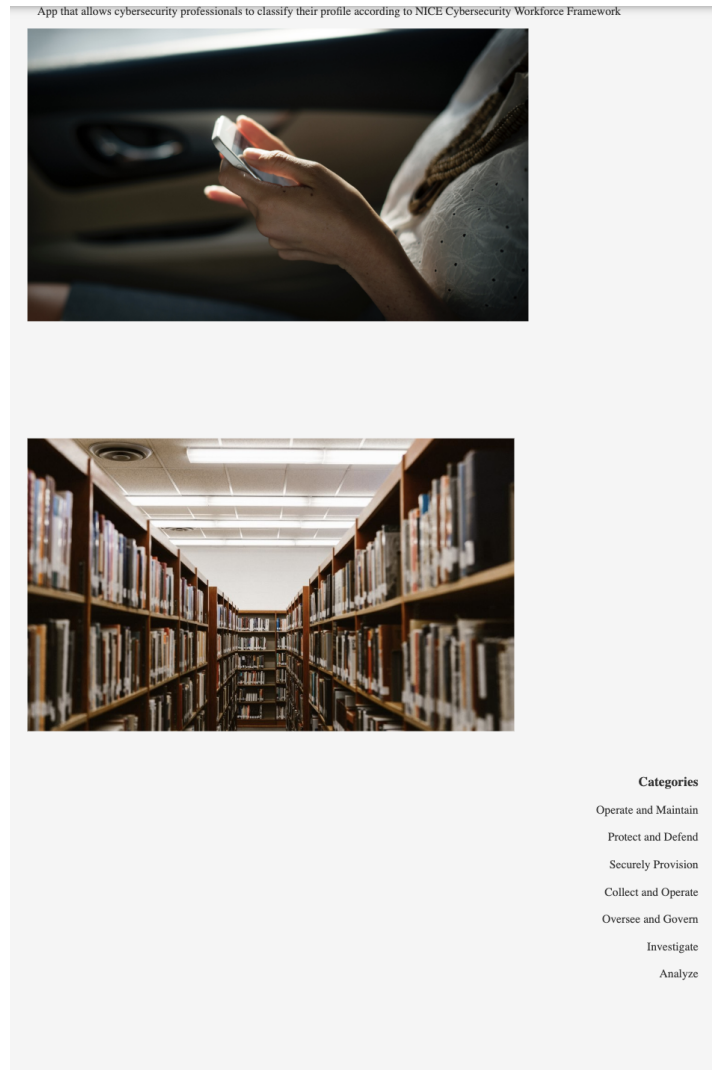


Fig. 3.21. Home page in a mobile phone screen

## Account

In the case of the Account page, even though the page is functional, it is just the same page as it is displayed in the computer but with the boxes smaller in order to fit in the screen. However, the problem is that this approach is not optimal due to the fact that only one quarter of the whole screen is used while the rest of it is just empty. As a consequence, the perceived quality of the User Interface of Sorting Hat decreases.

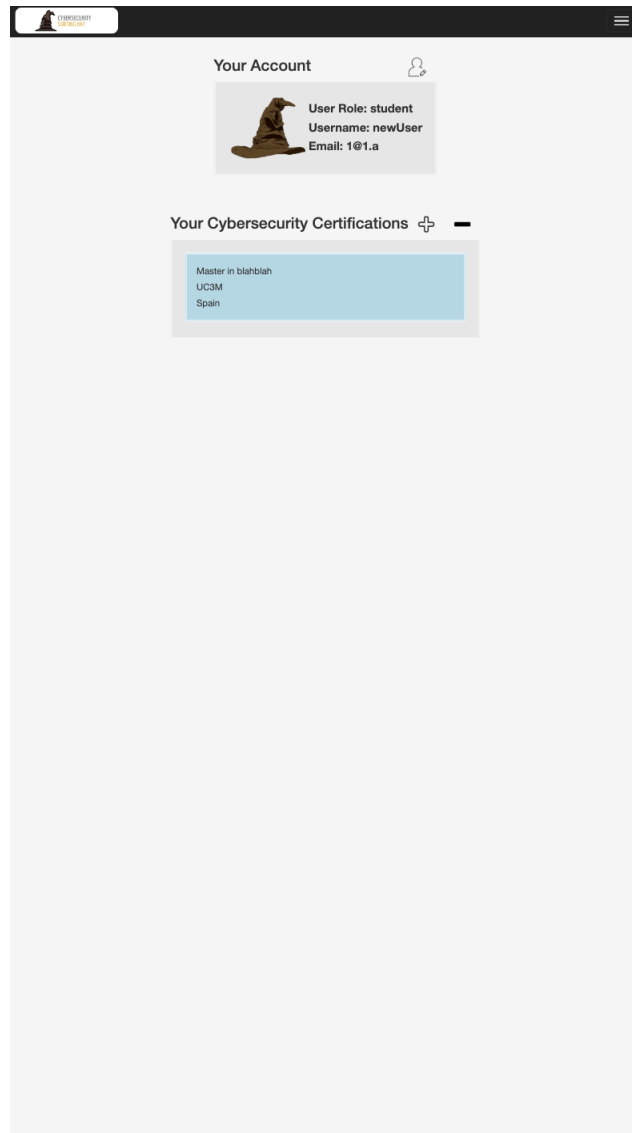


Fig. 3.22. Account page in a mobile phone screen

## Summary

In order to outline the main issues of the previous analysis, the following table is going to be used as a reference.

Responsiveness analysis		
Section	Analysis	Effect
Computer Responsiveness	Log in page background does not fill the whole screen for large screens	Downgrade of the User Experience
	Home images get cut if the screen is smaller, instead of been resized	
	The bottom bar of the Account page is not fix to the bottom of the page. Its position change depending on the size of the screen	
Mobile Responsiveness	Log in page background does not fit the whole screen	Downgrade of the User Experience
	Log in page boxes to register and log in are placed one next to the other leaving half of the screen empty	
	Home images are not resized correctly forcing to the text next to them to be moved below or over them	
	At the Account page only one quarter of the whole screen is used	

Table 3.16: Responsiveness analysis

### 3.2.4.5 User Interface Design

The design of the User Interface is going to be analyzed in order to determine inconsistencies in it.

#### Definition of a proper User Interface Design

There are several definitions of what a good User Interface Design is. However, almost all of them converge in four points which summarizes correctly the core fundamentals of

a good User Interface Design [75].

- *Error free*: the basis for any User Interface is the fact that it has no errors. If this happens, it does not matter how intuitive or visually appealing it could be because it is useless.
- *Intuitivity*: this principle is straightforward too. The ultimate objective of a User Interface is that the user use it in order to use the services and functionality of the Web Application. If the user does not need any instruction of how to use the User Interface and he can perform the task as fast as possible, the better User Experience that the user will experience.
- *Effective*: as well as for a User interface being intuitive is important, achieving the desired task which it is supposed to perform is also crucial to the overall User Experience.

### **Importance of a proper User Interface Design**

There is a major importance on the implementation of a proper User Interface Design if it is desired that the Web Application is used by as many people as possible. If the User Interface is not appealing and easy to use, simply the user will not use the Web Application.

### **User Interface Design analysis**

The User Interface Design is going to be divided into several categories regarding to the Design Principle which is incorrectly implemented [76] [77].

#### **Clarity**

According to the clarity of the User Interface of Sorting Hat, there are some issues located mainly in two different pages: the Test and the Dashboard pages.

With regards to the Test page, the issue is that when the results of the test and the top 10 matching roles are displayed, the figures have too many decimal digits. The main consequence is the fact that the user do not pay attention to the actual figures, but to the fact that there are too many digits.



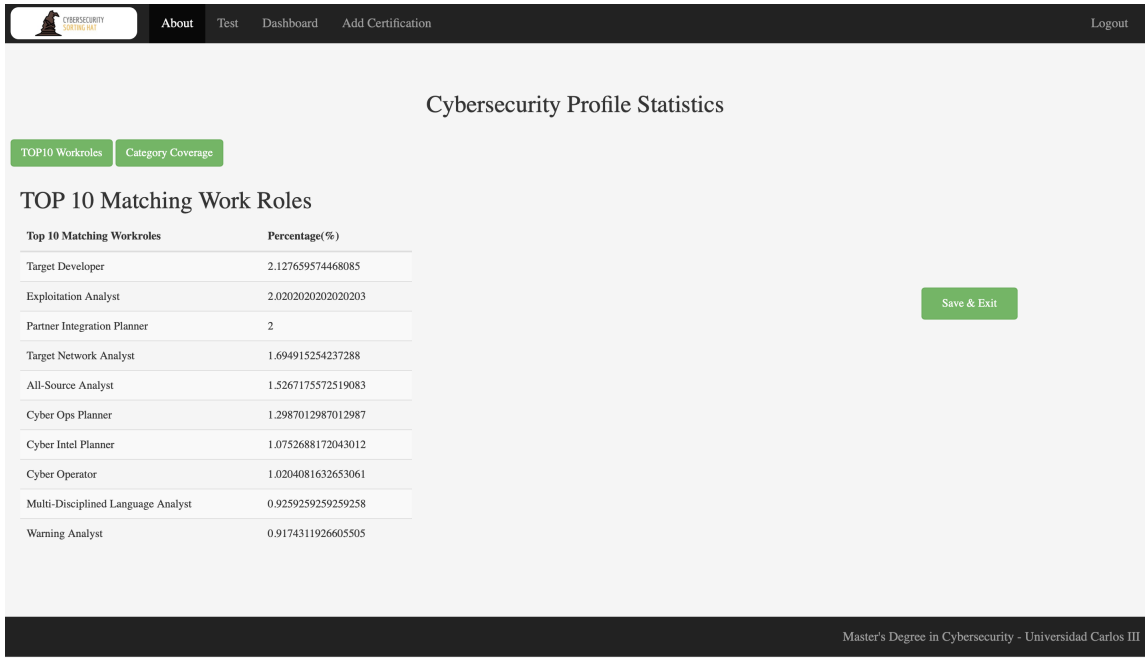


Fig. 3.23. The figures displayed in the Test page are not clear

The other Sorting Hat page in which there are issues related to the clarity is Dashboard. In this page the problem is that when the graphics are displayed only the symbol "%" is showed but no figures are showed.

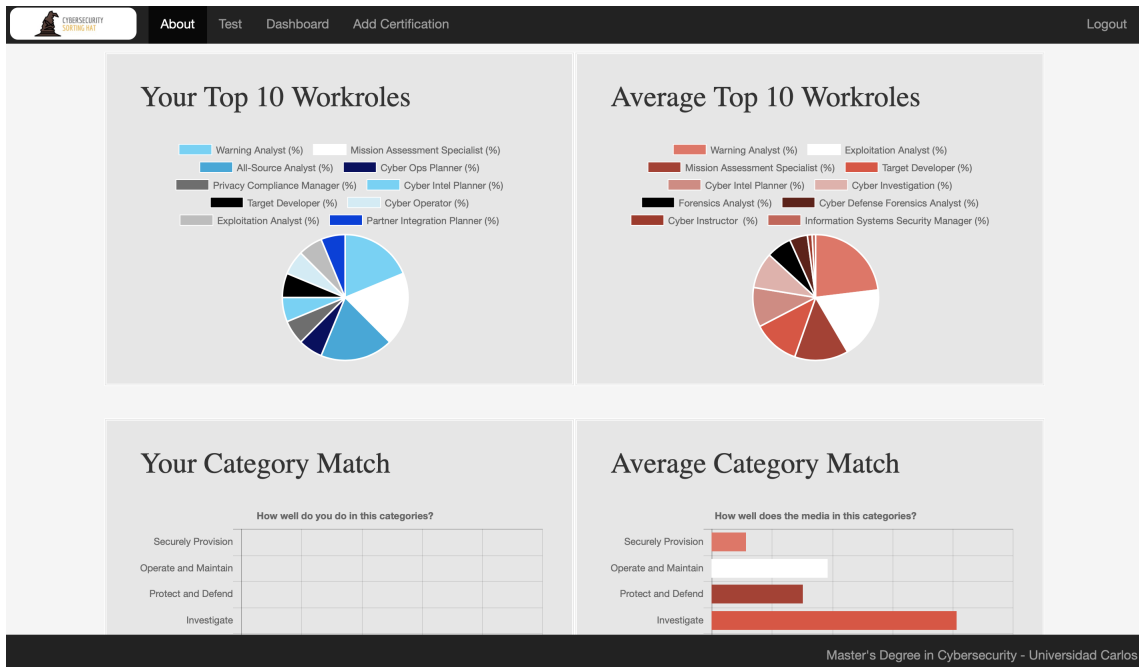


Fig. 3.24. No figures are displayed next to "%" at the DashBoard page

The last issue of the clarity of Sorting Hat pages is related to the first one described. In this case the issue is in the Dashboard page regarding to the last graph. If the cur-

sor is placed over it, the number of decimal digits which are displayed is excessive and unnecessary to the user.

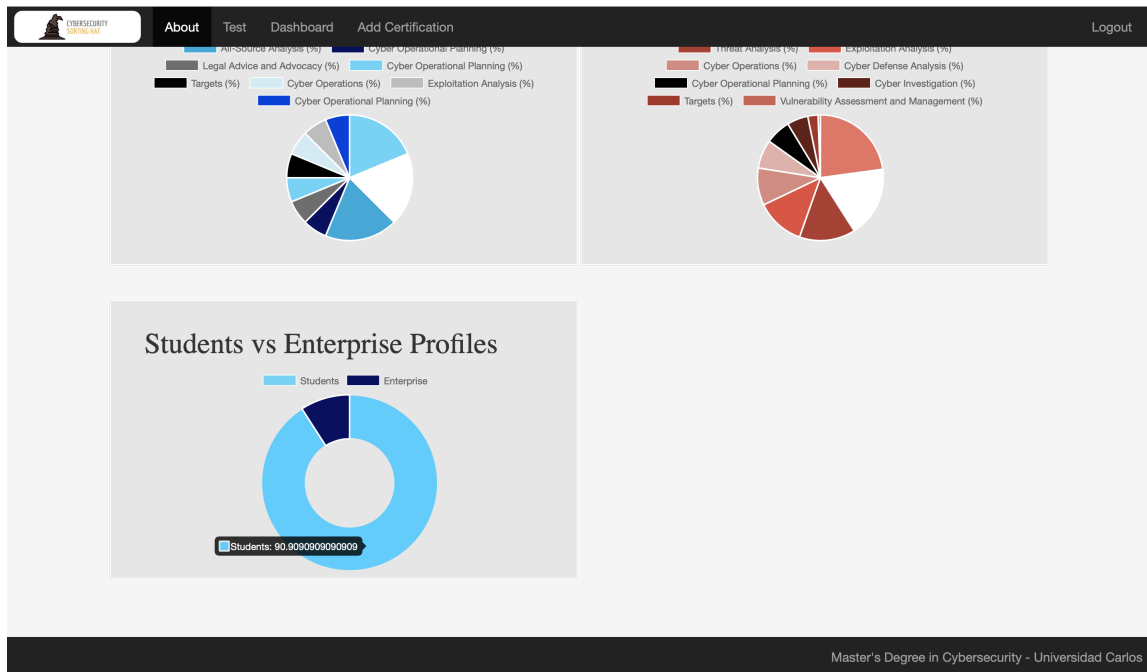



Fig. 3.25. The figures displayed over the graphs are not clear

## Color Palette

With regards to the Color Palette of the elements of Sorting Hat, one issue is found during the analysis. This problem is the inconsistency of the color of which the last diagram of the Home page has, compared to the rest of colors used in Sorting Hat.

The consequence of this inconsistency in the Color Palette is the feeling of unappealing which the user get from Sorting Hat.



**Audience**

Employers, to help assess their cybersecurity workforce, identify critical gaps in cybersecurity staffing, and improve position descriptions

Current and future cybersecurity workers, to help explore Tasks and Work Roles and assist with understanding the KSAs that are being valued by employers for in-demand cybersecurity jobs and positions. The NICE Framework also enables staffing specialists and guidance counselors to use the NICE Framework as a resource to support these employees or job seekers

Training and certification providers seeking to help current and future members of the cybersecurity workforce gain and demonstrate the KSAs

SECURELY  
PROVISION

OPERATE AND  
MAINTAIN

PROTECT AND  
DEFEND

INVESTIGATE

COLLECT AND  
OPERATE

ANALYZE

OVERSEE AND  
GOVERN

Fig. 3.26. Diagram at the Home page with no consistency with the color palette used in Sorting Hat

## Consistency

According to the Consistency analysis of the User Interface of Sorting Hat, several issues were noticed. The first issue was the inconsistency of the text paragraphs in the Home page. Each one of these paragraphs have different width size.



In addition to this, one of these paragraphs is just a column of categories which is not consistent with the rest of paragraphs in this page.

**NICE Framework is comprised of**

7 Categories: A high-level grouping of common cybersecurity functions

33 Specialty Areas: Distinct areas of cybersecurity work

52 Work Roles: The most detailed groupings cybersecurity work comprised of specific knowledge, skills, and abilities required to perform tasks in a work role

**Audience**

Employers, to help assess their cybersecurity workforce, identify critical gaps in cybersecurity staffing, and improve position descriptions

Current and future cybersecurity workers, to help explore Tasks and Work Roles and assist with understanding the KSAs that are being valued by employers for in-demand cybersecurity jobs and positions. The NICE Framework also enables staffing specialists and guidance counselors to use the NICE Framework as a resource to support these employees or job seekers

Training and certification providers seeking to help current and future members of the cybersecurity workforce gain and demonstrate the KSAs

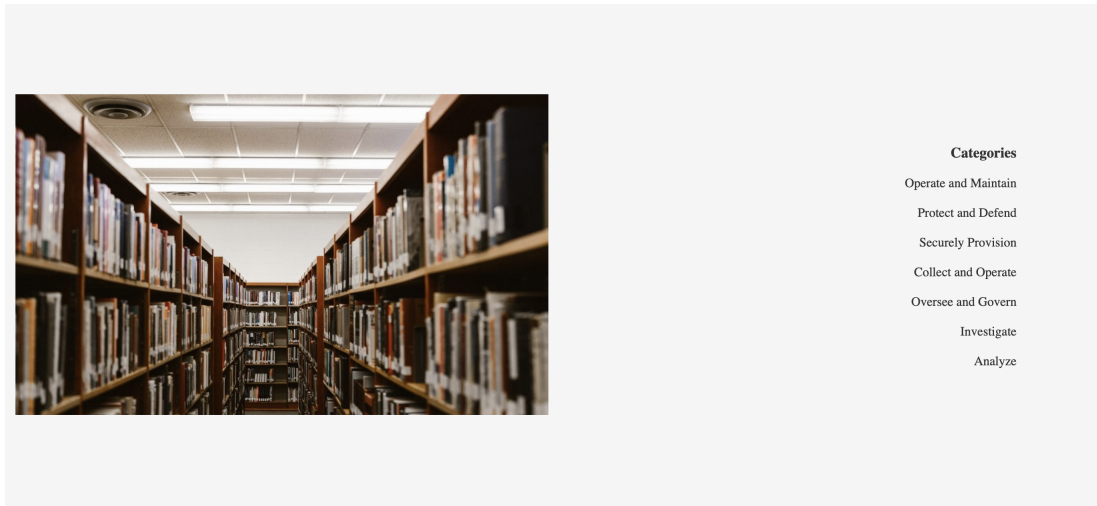
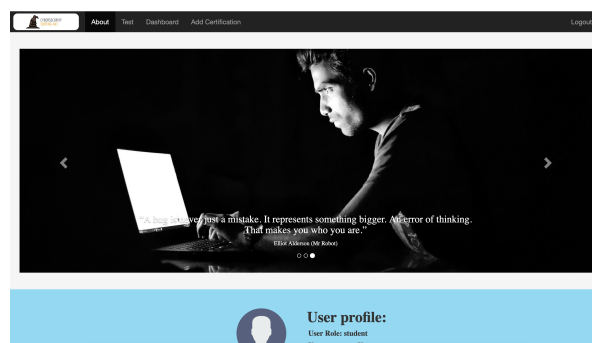


Fig. 3.27. Inconsistency of the text paragraphs at the Home page

Furthermore, another problem regarding the consistency is related to the font used in the navigation toolbar. The problem is the fact that this font is different in the Home page compared with the one used in the Test page.



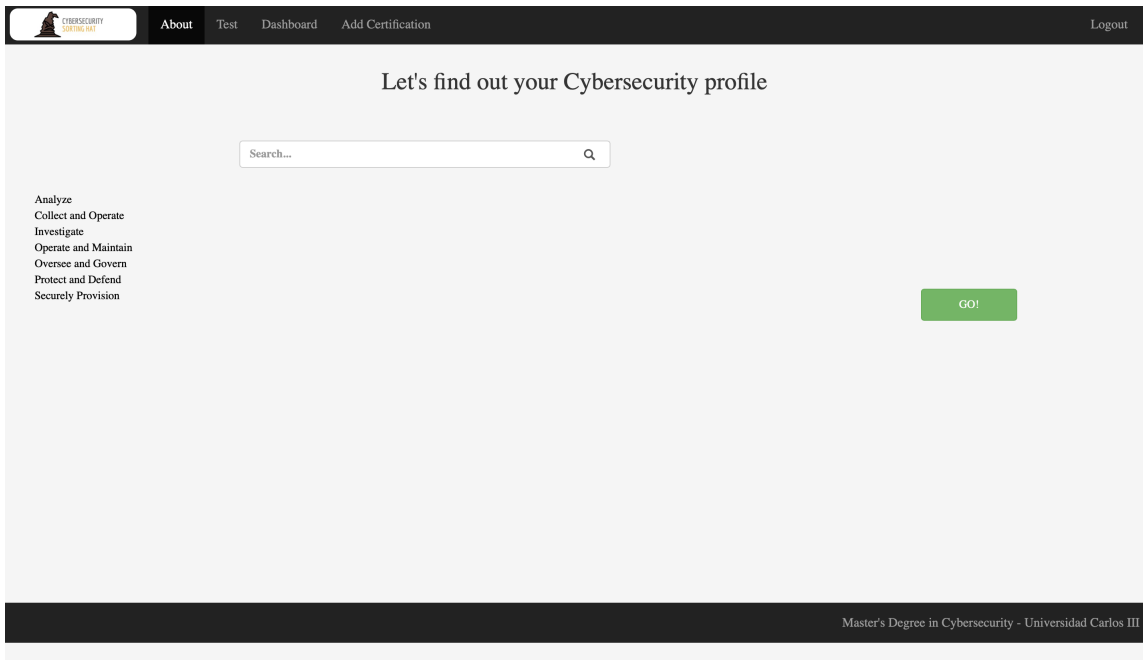


Fig. 3.28. Inconsistency in the font used for the Navigation Bar

## Effectiveness

The effectiveness analysis is one of the most important analysis performed to the User Interface due to its importance. In this analysis several issues were discovered.

The first issue was the fact that in the Home page, one of the images showed do not let to read the text over it correctly.

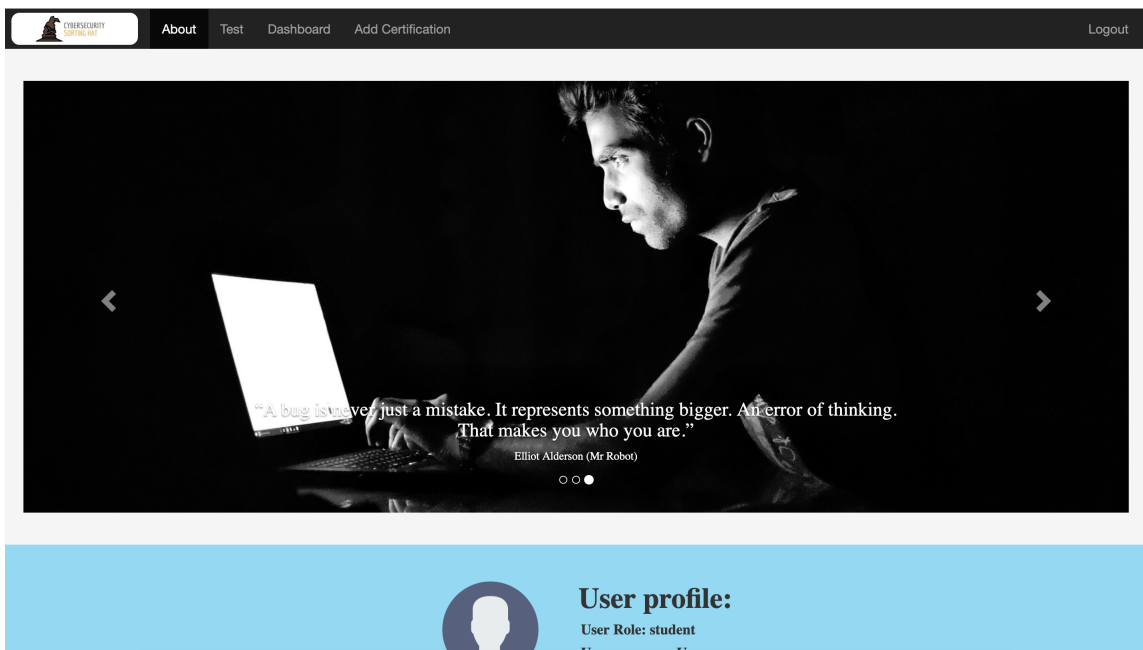


Fig. 3.29. The text in the Home picture cannot be correctly read

In addition to this, another more visual issue is related to the navigation bar. The problem in this case is that it does not matter in what page the user is that it always displays the About tab as if the user is in that page.

The consequence of this issue is the damaging of the User Experience of the user due to the fact that he could get lost in Sorting Hat without knowing the page in what he is.

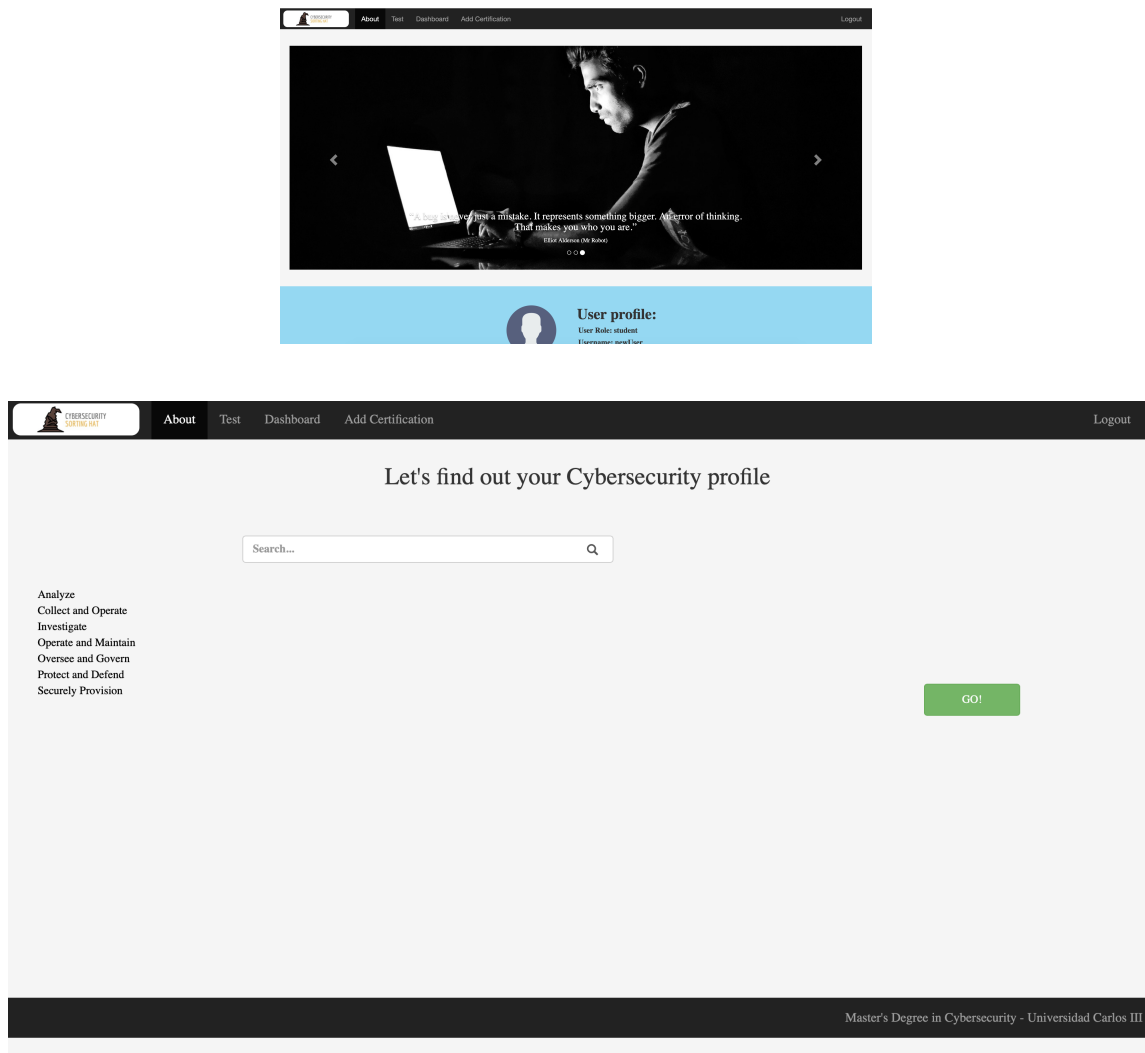


Fig. 3.30. The Navigation Toolbar always shows the same tab as the one selected

## Error Messages

In Sorting Hat there are some forms, such as the one to register into it, which in the case that the inputs from the user are not correct, Sorting Hat do not provide meaningful information about it.

One example of this situation is at the Log in page. If the user introduces wrong the values for registering Sorting Hat displays a message in which it explains all the possible fields which the user could have introduced wrong. It can be discerned that this is not the optimal way of displaying an error message to an user.

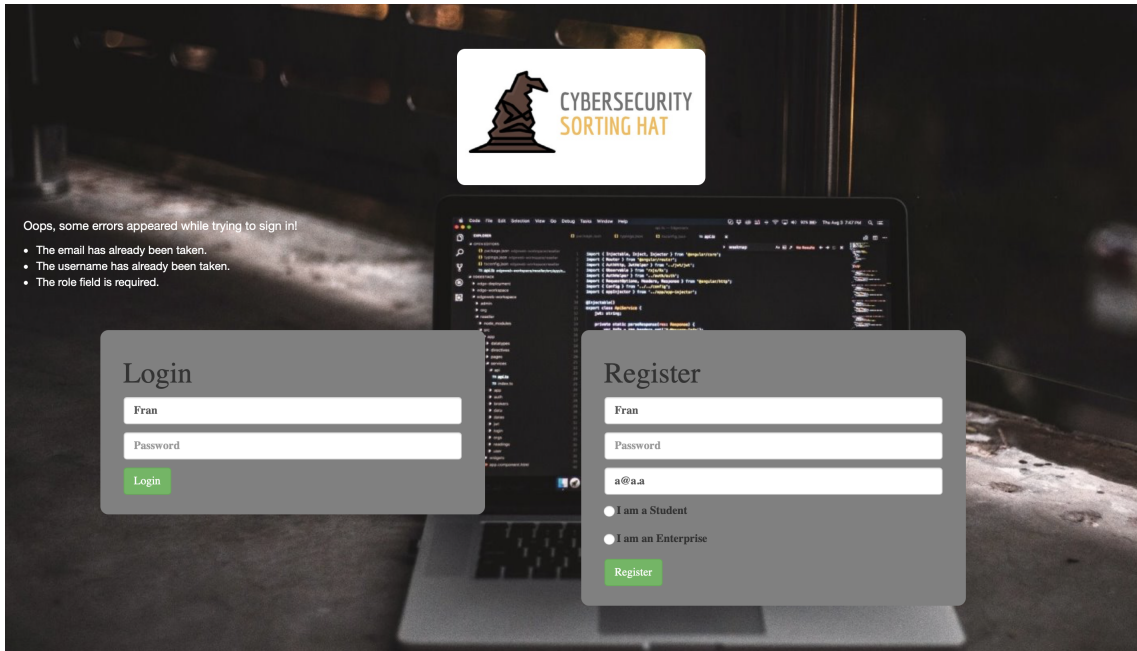


Fig. 3.31. Meaningless error message at the Log in page

As in the previous case, at the page of the Certificate Creation the situation is similar. If any of the fields are introduced incorrectly, a generic message is showed to the user.

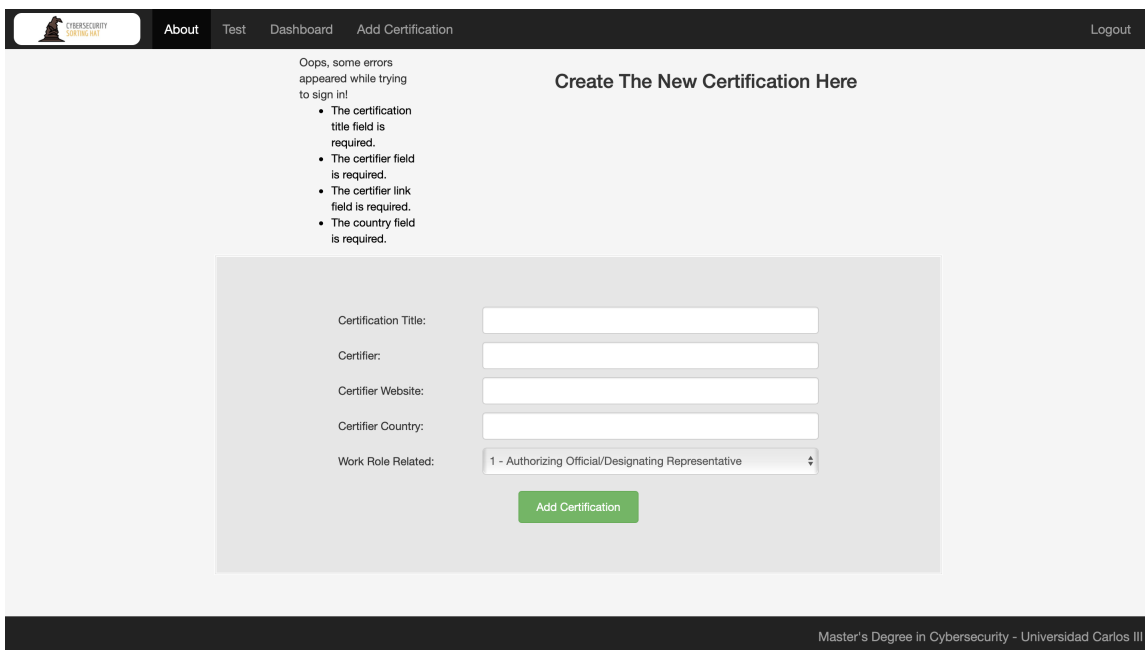


Fig. 3.32. Meaningless error message at the Certificate Creation page

## Symmetry

Symmetry is one of the most important features of an User Interface for been perceived as appealing to the user. If this is not achieved, it is probable that the User Interface does

not attract the attention of the user.

In Sorting Hat there are several examples of asymmetry in the User Interface. The first issue is at the Log in page. The problem is that the boxes to introduce the parameters to log in and register are not placed in the middle of the page. They are placed closer to the left side than to the right side creating a sense of inconsistency of the display of the User Interface.

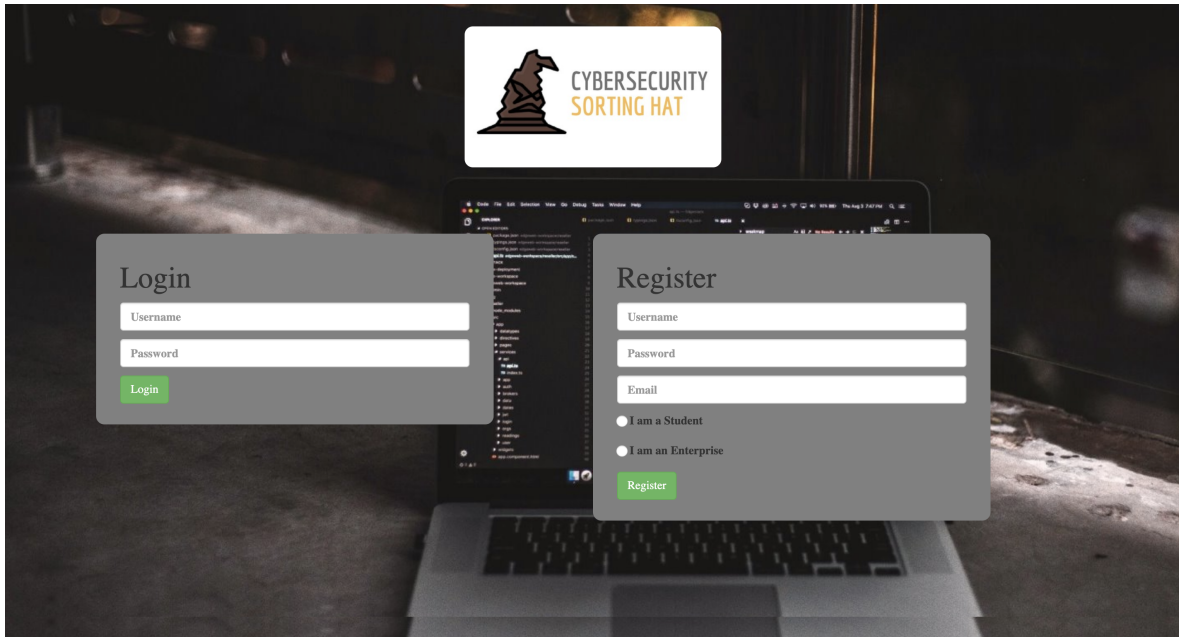


Fig. 3.33. Asymmetry in the Log in and Register form

This issue is related to the previous one described. In this case it is regarding to the diagram at the bottom of the Home page. This diagram is not set to the middle of the page, it is closer to the left side than to the left side.

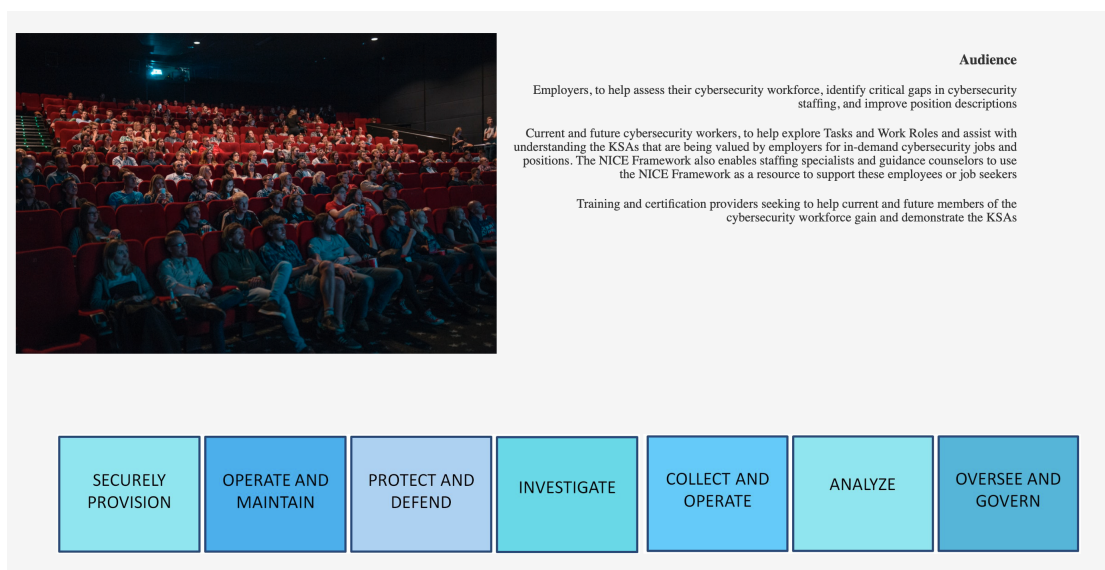


Fig. 3.34. Asymmetry of the diagram at the Home page



The last issue regarding to symmetry is at the Home page. The problem is that the width and height of the images are different each other. The same issue happens to the paragraphs of this page. The result of this situation is the sense of disorder of the positioning of the elements.

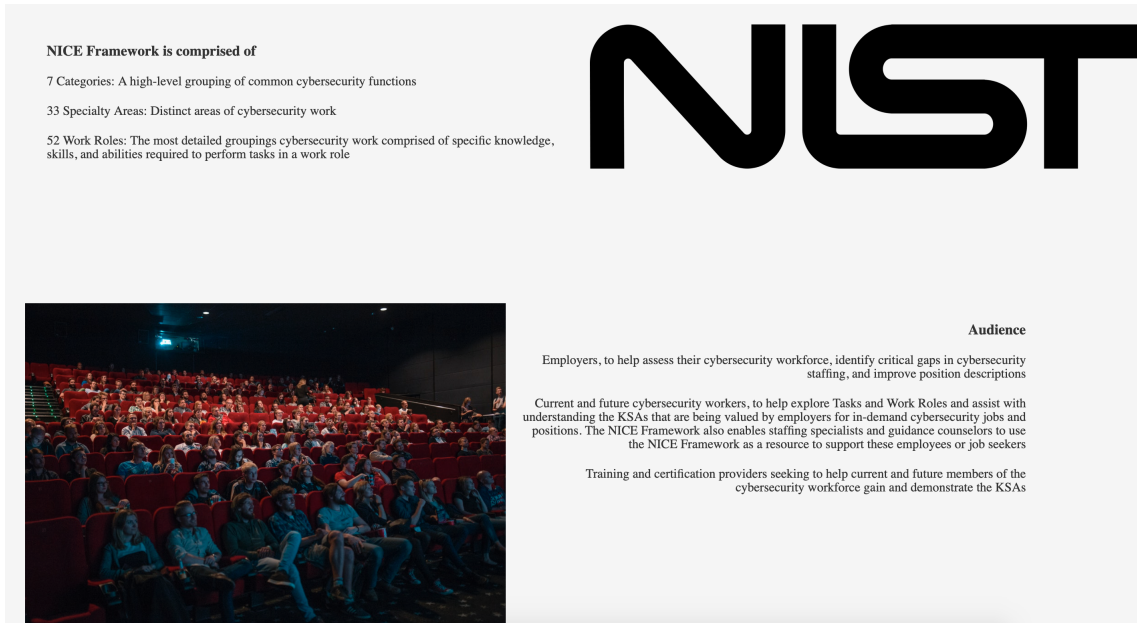


Fig. 3.35. Asymmetry of the text paragraphs at the Home page

## Summary

In order to outline the main issues of the previous analysis, the following table is going to be used as a reference.

User Interface Design analysis		
Section	Analysis	Effect
Clarity	The figures displayed in the Test page are not clear	Downgrade of the User Experience
	No figures are displayed next to "%" at the DashBoard page	
Color Palette	The figures displayed over the graphs are not clear	Downgrade of the User Experience
	Diagram at the Home page without any consistency to the color palette used	

User Interface Design analysis		
Section	Analysis	Effect
Consistency	Inconsistency of the text paragraphs at the Home page	Downgrade of the User Experience
	Inconsistency in the font used for the Navigation Bar	
Effectiveness	The text in the Home picture cannot be correctly read	Downgrade of the User Experience
	The Navigation Toolbar always shows the same tab as the one selected	
Error Messages	Meaningless error message at the Log in page	Downgrade of the User Experience
		Meaningless error message at the Certificate Creation page
Symmetry	Asymmetry in the log in and in the register form	Downgrade of the User Experience
	Asymmetry of the diagram at the Home page	
	Asymmetry of the text paragraphs at the Home page	

Table 3.17: User Interface Design analysis

## 4. IMPROVEMENTS PROPOSED FOR SORTING HAT\*

In this section of the Bachelor Thesis, various proposed improvements for Sorting Hat\* are going to be described. The objective of this proposals is to facilitate their future implementation in order to enhance the service offered to the user through Sorting Hat\*.

Regarding to the implementation of these suggestions, as it was explained in the first section of this Bachelor Thesis, they are not going to be accomplish in order to accomplish an analysis and proposing the improvements as accurate as possible.

### 4.1. Proposed Architecture for Sorting Hat\*

In this case, the actual architecture of Sorting Hat is a correct choice for this project. The architecture used is a N-layered model using the Model-View-Controller approach.

This approach is a very popular way of structuring Web Applications. Consequently, there is a great amount of documentation about the typical issues which could arise when this approach is followed.

Furthermore, the MCV and the 3 layered model provides a great level of abstraction of the different layers from which the Web Application consists. These layers are *User Interface*, *Business Logic* and *Data access*. Thanks to this level of abstraction, the process of developing and debugging the application are more straightforward and the dependencies between the different layers are better defined.

### 4.2. Description of the improvements for Sorting Hat\*

In this section of the chapter several improvements will be proposed in order to improve Sorting Hat\*'s efficiency and enhance the User Experience while using it. Furthermore, a detailed description of their implementation plan will be provided in order to simplify its further implementation process.

The different layer to which the improvement suggestions will be focus are the *Database*, the *Business Logic* and the *User Interface*.

## 4.2.1. Proposed Database improvements

In this section the proposed improvements for the Database will be described.

### 4.2.1.1 Database Project Structure

The difference of productivity between a project with a proper Database project structure and a project with a deficient Database project structure is immense. Moreover, this gap increases as the number of developers in the project increases too.

Consequently, ensuring an appropriate project structure is a key feature which all project should accomplish as soon as possible. However, usually once that the project has started and it evolves over time, maintaining a good Database project structure usually is mistakenly considered as a waste of time. Because of this, managers are responsible of inculcating its importance as well as establishing quality controls so as to avoid its degeneration.

#### File Structure

According to the file structure of the Database, the first proposal is to rename the root folder from "*Base de datos*" to "*Database*".

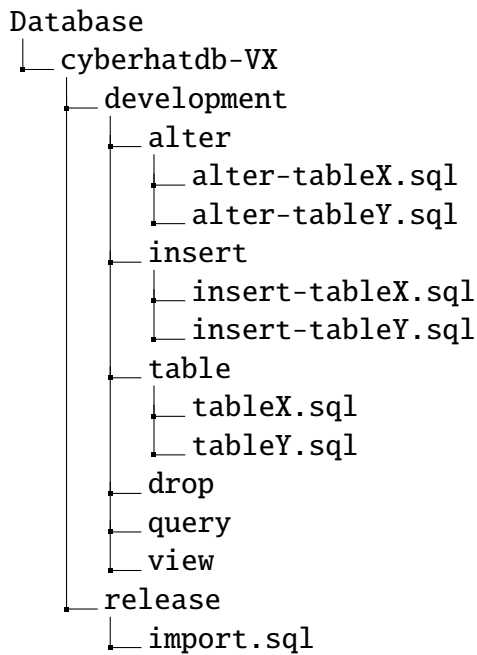
Inside this root folder, it is recommended to create a folder for each different Database which is been used in Sorting Hat\*. As in this case only "*cyberhatdb*" Database is been used, only the folder "*cyberhatdb-VX*" will be created. According to this new folder naming, "*V*" stands for version and "*X*" for the version's name, for example "*Database3-V4*" would be the forth version of the Database called "*Database3*".

Furthermore, inside "*cyberhatdb*" two folders are created. One is "*development*" where all the changes are performed and where different SQL files are located. The second folder is called "*release*". At this folder only one SQL file is stored, which is the last stable version of the Database without the new changes applied at the "*development*" folder.

Inside "*development*" there are different folders which corresponds to the different types of SQL statements. There is a folder for the "*ALTER*" statements, another one for "*INSERT*", "*CREATE TABLE*", "*DROP*", "*QUERIES*" and "*VIEWS*". Inside them the different SQL files are stored.

If more folders are needed, they could be created as long as it is well documented its creation purpose and the files which they are supposed to contain.

The proposed folder structure for the actual stage of the Database is the following:



Thanks to this file structure, the usage of a control version software such as Git is much more accessible letting the developers of the project to work in a more efficient way.

## Code Structure

The code structure of the original SQL file is just all the SQL code needed for the Database together in one file. Following the reasoning of the previous section, once that the file structure improvements have been implemented, the code structure improvements proposals can be accomplished.

The core of the code structure improvement is the concept of refactoring. Refactoring is the process of transforming the code implementation of a code file without modifying its original functionalities. The main reason for performing a refactoring of a program which currently is working properly is to ensure that if in the future any modification is needed it can be done efficiently without any consequence to the proper execution of it and its functionalities [78].

There are many different indicators which reveal the need to refactor code, such as dirty code or just bad design patterns followed while the development process [79]. In this case, the cause is the fact that a file with 7000 lines of code is not the optimal way to organize a file of code.

Consequently, the proposed improvement is refactoring the original into different files, the ones described in the previous section. That is to say to separate all the creations of tables, all the alter processes and all the insertions.

## Summary

In order to outline the main proposed improvements of the previous section, the following table is going to be used as a reference.

Database Project Structure improvements		
Section	Improvement	Effect
File Structure	Rename root folder from " <i>Base de datos</i> " to " <i>Database</i> "	
	Create a new folder for each Database	Facilitate the access to the project to non Spanish developers
	Add to the Database's folder the current version of it	Simplify the implementation of new Databases to the project
	Inside the Database's folder create one folder for <i>development</i> and other folder for <i>release</i>	Speed the navigation through the project
	Inside <i>release</i> divide the files regarding to the different SQL statements or elements	
Code Structure		Speed the navigation through the project
	Refactor the original SQL file to fulfill the requirements of the proposed File Structure	Improve the productivity of the implementation of upgrades to the project

Table 4.1: Database Project Structure improvements

### 4.2.1.2 Code style

#### SQL code style

As MySQL is the query language used in this project, its reference manual and statement syntax is going to be the used as reference to ensure the correctness of the proposed improvements [80].

The main objective of the code style's improvements which are going to be proposed in this section are related to ensure the consistency of *cyberhatdb*'s SQL code. There are

two main issues according to the code style: table naming and field naming.

1. Table naming: In order to maintain the consistency when the tables are created, it is suggested to follow the subsequent guidelines:

- Meaningfulness: The names chosen for the created tables should clearly indicate what information the table stores. Furthermore, the names should be brief and meaningful being both of them equally important [81].
- Collectives: Table's name should use a collective name in the case that it exists.
- Singular: The singular form of the table's name should be used when it is possible.
- Concatenation: Avoid to concatenate two table's names in order to refer to the relationship between them. In the case that there is no English word to refer to that relationship, there is no issue concatenating them.
- Prefixes and suffixes: It is not recommended to use prefixes or suffixes in the table's name, except the case in which there is no other choice to do so.
- Casing: In order to the casing of the table's names the Snake Case is going to be used as the standard for them.

2. Column naming: The suggested improvements for enhancing column naming are the following:

- Singular: All the columns should be written in singular in order to improve the efficiency while managing the Database.
- Prefixes and suffixes: So as to be consistence across the column's names only suffixes will be used.
- Casing: In order to improve the consistency in all the SQL code, the same casing as table naming will be proposed. Snake Case is the suggested case convention for column's name [82].
- Identifier field: As it was stated in the analysis section of the Database, naming the identifier column as "*id*" is not recommended due to the fact that it can lead to mistakes while referencing several identifiers from different tables.

Consequently, it is recommended to use the name's table adding the suffix

"\_id", such as *"client\_id"*.

- Identifier suffix: Just to make it clear, the identification suffix should be the last suffix of the column's name and it should be in lower case, such as *"please\_be\_consistent\_while\_coding\_id"*.

In conclusion, it is not so important what guidelines a developer is using while coding as well as a guideline is followed and the consistency along the code is guaranteed. It is highly recommended that in the case that any modification to this suggested code style is desired to be done to document the new code style. If this is not done the project could lead again to a complete code inconsistency again.

## **Summary**

In order to outline the main proposed improvements of the previous section, the following table is going to be used as a reference.



SQL Code Style improvements		
Section	Improvement	Effect
Table Naming	Meaningful and brief names	
	Use of collective names when possible	
	Singular form for table's names	Code consistency
	Avoid the concatenation of tables' names	Improve development and debugging efficiency
	Avoid using prefixes and suffixes	
	Use Snake Case as the naming standard	
Column Naming	Singular form for column's names	
	Only suffixes will be used for column naming	
	Add the column's name as suffix to the <i>id</i> column	Code consistency
	<i>id</i> is written in lowercase	Improve development and debugging efficiency
	<i>id</i> is the last word concatenated in its column's name	
	Use Snake Case as the naming standard	

Table 4.2: SQL Code Style improvements

#### 4.2.1.3 Relational Data Model

In this section of the improvements proposed for Sorting Hat\* Database, all the different aspect assessed during the analysis of Sorting Hat's Database will be taking into account in order to be as accurate as possible.

## Trivial tables

According to the tables from Sorting Hat\*'s Database which are trivial, "proficiencylevel" is the most important of all of them. This table contains the five different proficiency levels. The issue is the fact that all the tables which reference to this table contains the same data.

Consequently, if all the tables contain the same data regarding to "proficiencylevel", this table is useless and a great amount of storage is being wasted. In order to solve this problem, the table "proficiencylevel" and the columns referencing to it should be deleted.

Similarly, as "profileset", "user\_has\_profile", "user\_has\_targetprofile", "competency", "tag", "profile\_with\_proficiency\_level", "user\_selected" and "profileset\_has\_profile" has no data stored in them, they also should be deleted in order to enhance the efficiency of the Database.

## Trivial columns

Regarding to the columns which have no clear purpose in the Database, several approaches are going to be discussed for dealing with this situation.

To start with, as "LowerProficiencyLevel\_id" and "UpperProfficiencyLevel\_id" have the same value for all the instances they have no purpose in Sorting Hat\*. As a consequence, these two columns should be deleted.

In addition to this, "Type" belonging to "profile" is in the same position as the previous columns. The recommended improvement for the column "Type" is the same as the former one, deleting the column "Type" due to its uselessness.

The issue regarding to the column "remember\_token" is the fact that no data is stored into it. Consequently, in order to enhance the efficiency of the database the column "remember\_token" should be deleted.

In regards with the column called "NCWF\_SpecialityArea\_NCWF\_Category\_id" belonging to the table "ncwf\_workrole", could be deleted with no complication. This is based on the fact that "ncwf\_workrole" already contains "NCWF\_SpecialityArea\_id" column which references to the table "ncwf\_specialityarea" where the "NCWF\_Category\_id" is already stored. Consequently, this data is redundant.

If this improvement is accomplished, the relation between the three different NCWF's tables will be more intuitive and no redundant data will be stored in then.

## Similar columns

The most relevant issue according to similar columns is the equality between the columns "SourceReference" and "Source" belonging to the tables "knowledge", "task", "skill" and

"ability". Both columns have the exactly the same value in all of the aforementioned tables.

The solution to this problem is straightforward. The column *"SourceReference"* need to be deleted from all the tables previously mentioned.

Furthermore, according to the columns *"created\_at"* and *"updated\_at"* belonging to "user\_has\_certification", which is the junction table between "user" and "certification" tables, are already stored at table "certification". Consequently, in order to avoid redundant data, both columns need to be removed from the junction table "user\_has\_certification".

### **Inappropriate column format**

To start with, according to the inappropriate column format of the tables of Sorting Hat\*'s Database, the first issue is the length of the column *"Profile\_id"*. This column appears in several tables, such as "certification", "profilesset" or "profile\_with\_proficiencylevel". This column references to the working profile which belongs to the table "profile".

The problem arises when the number of profiles are calculated. There are 52 different profiles, obtained from the NCWF framework. Nevertheless, in the tables this column is set to have up to 11 digits, which could store more than 90.000 millions of profiles.

As it can be perceived, this is not the optimal format to the column *"Profile\_id"*. It is suggested to changed its length to 4 digits. With this measure, the storage space used for saving this column will decrease in more than 50% and it could store up to 9.999, which taking into account that these profiles are defined by the framework and that currently only 53 are defined it should be more than enough for supporting future updates.

Furthermore, the same issue occur in the columns *"NCWF\_SpecialityArea\_id"* and *"NCWF\_Category\_id"* which are the identification columns of the tables "ncwf\_specialityarea" and "ncwf\_category". These tables have 33 and 7 different elements stored respectively.

Consequently, in order to identify them using the identification columns, only 2 digits could be used. However, as in the previous case, it is recommended to use 4 digits in order to be able of supporting large updates in the future.

Finally, the last issue regarding to the inappropriate column format is according to the table "profile". This table contains a column called *"type"* inside which the type of the workrole is indicated.

Nevertheless, instead of referencing each of this profiles to the corresponding type, which could be stored in a separate table and reference through some digits, such as using a foreign key with the identification, it is stored as a enumeration of varchars containing up to 45 letters. Consequently, the same varchar is copied in all the instances of the table "profile".

In order to solve this situation, it is recommended to create a new table which will be called "profile\_type" with only two columns: its identifier and the name of the type of profile. As a result, in the table "profile" the column "type" will be changed to store up to 2 digits, which will correspond to the identifier of the "profile\_type" being this column the foreign key to "profile\_type".

In the case that this approach is taken, the Relational Data Model of the Database will be more intuitive and the space reserved for storing the profile's type will greatly decrease.

### **Trivial Primary Keys**

In this section the issues according to trivial primary keys will be dealt with. The main issue is the fact that the tables "knowledge", "task" and "skill" incorporate as part of their primary key a column called "Source" as well as the identifier.

In order to improve the efficiency and clarity of the Relational Data Model, this column needs to be removed as part of the primary key of the aforementioned tables.

Furthermore, another issue which is related with the former one described is related to the tables "profile\_has\_knowledge", "profile\_has\_task", "profile\_has\_skill" and "profile\_has\_ability". These junction tables have as primary keys all of their columns. These columns are: the identification of both tables, "Source", "LowerProficiencyLevel\_id" and "UpperProficiencyLevel\_id".

As it can be seen, these tables only need as primary key the identification of both tables, because they are junction tables. Consequently, "Source", "LowerProficiencyLevel\_id" and "UpperProficiencyLevel\_id" do not have to be part of the primary key.

### **Missing Foreign Keys**

As it is described in the previous sections, the column "type" from the table "profile" is modified. Previously, it stored the type of the profile as a varchar. In the proposed improvement this column is modified and it is set to be an integer of 2 digits, which is the foreign key to the table "profile\_type".

### **New Tables**

As a result of the modification of the table "profile", the new table "profile\_type" is created. It contains two columns: an identifier, which is an integer of 2 digits, and a name, which is a varchar of 45 letters. It is referenced by "profile" through the column "type" as a foreign key.

## Relational Data Model diagram

### Summary

In order to outline the main proposed improvements of the previous section, the following table is going to be used as a reference.

Relational Data Model improvements		
Section	Improvement	Effect
Trivial tables	Remove the table <u>"proficiencylevel"</u> and all the columns referencing to this table	Optimization of the Database storage
	Remove <u>"profileset"</u> , <u>"user_has_profile"</u> , <u>"user_has_targetprofile"</u> , <u>"tag"</u> , <u>"competency"</u> , <u>"profile_with_proficiency_level"</u> , <u>"user_selected"</u> and <u>"profileset_has_profile"</u>	Remove useless elements from the Database  Enhance the simplicity and intuitiveness of the Relational Data Model
Trivial columns	Remove <u>"LowerProficiencyLevel_id"</u> and <u>"UpperProfficiencyLevel_id"</u>	Optimization of the Database storage
	Remove <u>"remember_token"</u> belonging to the table <u>"user"</u>	Remove useless elements from the Database
	<u>"NCWF_SpecialityArea_NCWF_Category_id"</u> is removed from table <u>"ncwf_workrole"</u>	Enhance the simplicity and intuitiveness of the Relational Data Model
	Remove <u>"Type"</u> column belonging to <u>"profile"</u>	

Relational Data Model improvements		
Section	Analysis	Effect
Similar columns	<i>"SourceReference"</i> column from <i>"knowledge"</i> , <i>"task"</i> , <i>"skill"</i> , <i>"ability"</i> are deleted	Optimization of the Database storage
	<i>"created_at"</i> and <i>"updated_at"</i> belonging to <i>"user_has_certification"</i> are deleted	Remove useless elements from the Database  Enhance the simplicity and intuitiveness of the Relational Data Model
Inappropriated column format	The column <i>"Profile_id"</i> is changed from containing 11 digits to 4 digits	
	The columns <i>"NCWF_SpecialityArea_id"</i> and <i>"NCWF_Category_id"</i> are changed from containing 11 digits to 4 digits	Optimization of the Database storage
	The column <i>"type"</i> from the table <i>"profile"</i> is changed from containing a varchar to an integer of 2 digits	
Trivial Primary Keys	The tables <i>"knowledge"</i> , <i>"task"</i> and <i>"skill"</i> get removed the column <i>"Source"</i> as part of their primary key	
	<i>"Source"</i> , <i>"LowerProficiencyLevel_id"</i> and <i>"UpperProficiencyLevel_id"</i> do not need to be part of the primary key of the junction tables <i>"profile_has_knowledge"</i> , <i>"profile_has_task"</i> , <i>"profile_has_skill"</i> , <i>"profile_has_ability"</i>	Enhance the simplicity and intuitiveness of the Relational Data Model

Relational Data Model improvements		
Section	Analysis	Effect
Missing Foreign Keys	The column <i>"type"</i> from <i>"profile"</i> is set to be a foreign key to the table <i>"profile_type"</i>	Enhance the simplicity and intuitiveness of the Relational Data Model
New Tables	The table <i>"profile_type"</i> is created. It contains an identifier of 3 digits and a varchar name of 45 letters	Enhance the simplicity and intuitiveness of the Relational Data Model

Table 4.3: Relational Data Model improvements

#### 4.2.1.4 Security

##### MySQL accounts

Firstly, the most important issue to solve is establishing a password to the root user of the MySQL Database. If this is performed, the vulnerability of accessing to the Database with root privileges will greatly decrease.

In order to solve this issue, a password needs to be set. This can be done using the following command [83].

```
SET PASSWORD FOR 'root'@'localhost' = PASSWORD('MyNewPass') (4.1)
```

Furthermore, as there are different root users for different hosts, the former command has to be done defining the host as *localhost* and *127.0.0.1*.

Another approach for solving this problem is *mysql\_config\_editor*. The following command needs to be executed in order to change the root's password [84];

```
./mysql_config_editor set --user = root --password (4.2)
```

## Backups

A backup is a duplicate of the data and state of a Database. It is crucial to have a proper backup policy in order to avoid the situation where the Database is compromised and these data is not saved in a secondary storage. This situation could lead to the inability for the users to access to Sorting Hat\*.

In order to define a backup policy, the first thing which needs to be determined is the Database Backup Schedule. It is recommended that the period of time between each backup is less than 7 days. In the case that by any cause it is needed to performed another backup earlier than the established time, it could be performed with no issues. Nevertheless, the backup should never be performed later that the defined period in the backup policy, which currently is 7 days [85].

According to the implementation of the backup policy, there are two main approaches for its implementation.

- *Automatic Backup*: If this approach is followed, it is only needed to determine some parameters, such as the user name and the password, in order to the software to perform automatically the backup. Several software are available at Internet for this purpose. One of them is *AutoMySQLBackup* which among other features it includes email notification and backup compression and encryption [86].
- *Manual Backup*: In the case that this approach is chosen, the administrator of Sorting Hat\* would have to perform manually the backup. Several approaches could be followed for this case. Furthermore, it is recommended to use *phpMyAdmin* in order to accomplish a manual backup. This is based on the fact that *phpMyAdmin* is already implemented in Sorting Hat\* [87].

The approach which is recommended to be followed is the *Automatic Backup* due to its clearly advantages according to usability and reliability.

## Summary

In order to outline the main proposed improvements of the previous section, the following table is going to be used as a reference.



Database Security improvements		
Section	Improvement	Effect
MySQL Accounts	Establishing a password for the root user	Protecting the access to root privileges
Database Backup	Define the backup schedule	Increase of the reliability in the Database
	Implement an automatic backup solution	Ensuring a backup Database in the case that the Database is corrupted

Table 4.4: Database Security improvements

## 4.2.2. Proposed JavaScript improvements

In this section the proposed improvements for the JavaScript implementation will be described.

### 4.2.2.1 JavaScript Library

The JavaScript Library used in Sorting Hat\* is jQuery. According to the analysis performed in the previous chapter, one main issue according to the implementation of jQuery in Sorting Hat\* was noticed.

This issue was the fact that jQuery was imported twice. It is only needed to import a JavaScript file once, in this case the JavaScript file is jQuery Library. Furthermore, it is widely recommended to load the external JavaScript files at the end of the body of the HTML files [55]. As a consequence, the loading of the User Interface is not hindered by the fetching and loading of the JavaScript files improving the efficiency of Sorting Hat\*.

To sum up, jQuery should be imported only once at the bottom of the body of the HTML documents.

### Summary

In order to outline the main proposed improvements of the previous section, the following table is going to be used as a reference.

JavaScript Library improvements		
Section	Improvement	Effect
jQuery	Load JQuery Library only once at the end of the HTML body	Increase the efficiency of the loading of Sorting Hat* User Interface

Table 4.5: JavaScript Library improvements

#### 4.2.2.2 JavaScript Project Structure

Defining a proper JavaScript Project Structure is a key feature which is needed in order to ensure that the developers are as efficient as possible managing the JavaScript files.

##### File Structure

The main issue regarding to the File Structure of the JavaScript files is the fact that there are two folders with similar names. These folders are: *"js"* and *"javascript"*. Both of these files contains JavaScript files.

In order to enhance clarity and intuitiveness the folder *"javascript"* is deleted and the files which were stored inside it are copied to the folder *"js"*.

Furthermore, another problem which was found was the fact that there was a duplicated file. The name of this file was *"jquery.js"*. To solve this situation the copy of this file which was stored in the folder *"js"* is deleted.

Once that this improvements are implemented the JavaScript Project Structure should be similar to the following one.

```

SortingHat*
├── public
│   ├── css
│   ├── icons
│   └── js
│       ├── app.js
│       ├── jquery.js
│       └── testPage.js

```

##### Code Structure

To start with, one of the main issues according to the JavaScript Code Structure is the fact that the main JavaScript file is named *"arboles.js"*. It is known that the files' name need to be as descriptive as possible and, if it is possible, in English.

As a consequence, *"arboles.js"* should be renamed to *"testPage.js"* where *"test"* is the web page in which this file is used.

Furthermore, another issue according the code structure of the JavaScript file is the fact that all the methods have no JavaDoc comments in order to explain their behaviour. In order to solve this situation, all the methods should be forced to have a descriptive JavaDoc explaining its behaviour and execution flow.

Another problem is the fact that are several logs along the implementation of the methods aforementioned. The resolution of this situation is straightforward, removing all the logs of the file *"testPage.js"*.

Finally, the last issue is regarding the imports of the JavaScript file from the HTML documents. Currently, this file is imported from all the pages. However, this JavaScript file is only needed at the page called *"test"*. Consequently, the imports of *"testPage.js"* that are not in *"test"* should be removed.

## Summary

In order to outline the main proposed improvements of the previous section, the following table is going to be used as a reference.

JavaScript Project Structure improvements		
Section	Improvement	Effect
File Structure	Delete the folder called <i>"javascript"</i>	Simplify the implementation of new JavaScript files to the project
	Delete the duplicated file called <i>"jquery.js"</i>	Speed the navigation through the project
Code Structure	Rename the file called <i>"arboles.js"</i> to <i>"testPage.js"</i>	Speed the navigation through the project
	Implement JavaDoc comments to all the methods	Improve the productivity of the implementation of upgrades to the project
	Remove all the logs	Facilitate the access to the project to non Spanish developers
	Remove unnecessary imports	

Table 4.6: JavaScript Project Structure improvements

### 4.2.2.3 JavaScript Code Style

According to the improvements for Sorting Hat\* regarding the code style, the main issue which needs to be solved is the inconsistency about variable's naming.

The solution determined in order to fix this problem is establishing a naming standard for the variables. In order to decide the naming standard, the JavaScript Style Guide of *Airbnb* is going to be used as a reference. At *Airbnb* Style Guide the variable naming follows *camelCase* naming. Consequently, *camelCase* is going to be the naming convention followed for variable's naming [88].

Furthermore, the previous analysis accomplished to the JavaScript code of Sorting Hat\* disclosed that all the comments were written in Spanish. In this case, the solution is straightforward. All the comments should be translated into English. As a consequence, developers who do not speak Spanish could join to the project leading to a more intercultural team.

Additionally, one of the issues which could have the worst impact in the project is the great amount of hardcoded variable comparisons. Through the lines of JavaScript code of Sorting Hat\*, several Strings are compared with hardcoded values defined by the developers. The result of this practise is that in the case that any of these values needs to be changed errors probably will arise.

In order to solve this situation, these hardcoded values need to be declared in the top of the JavaScript in order to accelerate its modification in the case that it is needed.

Finally, the last improvement which is recommended according the JavaScript Code Style is regarding to the declaration of variables. The current problem is the fact that between the different methods declared along the JavaScript file there are several variables declared. This practise is not optimal due to the fact that it increases the time searching for this variable and it could lead to issues according to the scope of the variable.

As a consequence, all the variables declared between the methods should be moved to the top of the file, just below the declaration of the hardcoded Strings constants.

If any doubt arises during the future development of the project according to the JavaScript Code Style, *Airbnb* Style Guide is recommended to be followed as a standard [88].

### Summary

In order to outline the main proposed improvements of the previous section, the following table is going to be used as a reference.

JavaScript Code Style improvements		
Section	Improvement	Effect
Variable Naming	Consistency of variables' name is enforced following the camelCase naming standard	Reduce of the possibility of having misspelling errors
		Efficient development
		Efficient debugging
Variable Declaring	Global variables declared between methods are moved to the top of the file	Efficient development
		Efficient debugging
Hardcoded String Variables	The hardcoded Strings used in the comparisons should be declare at the top of file	Efficient development
		Efficient modification of the code
Comments Language	All the comments should be translated into English	Eases the incorporation of non Spanish developers to the team

Table 4.7: JavaScript Code Style improvements

### 4.2.3. Proposed PHP improvements

In this section all the proposed improvements for the PHP implementation will be described.

#### 4.2.3.1 PHP Framework

The PHP Framework used for the implementation of the PHP code of Sorting Hat\* is Laravel. Laravel is one of the most popular and spread PHP Frameworks. A more detailed explanation about this Framework was accomplished in the section regarding to the Analysis of the PHP Framework.

According to the issues related to the Laravel implementation in Sorting Hat\*, there is only one main issue. The main problem is the fact that the feature which Laravel provides to the developers in order to develop Responsive User Interfaces is not used. This feature is the implementation of *Vue* components.

Vue is a a Progressive JavaScript Framework focuses only in the front-end of a Web

Application. It is characterized by its great performance while manipulating the DOM and its versatility for adapting to other Frameworks [89].

As a consequence of its versatility and performance, Laravel allows the use of Vue components facilitating the modification of the User Interface as well as the implementation of a Reactive User Interface [90].

Therefore, Vue components should be used in order to upgrade the User Interface into a Reactive version so as to enhance the User Experience of Sorting Hat\*.

### Summary

In order to outline the main proposed improvements of the previous section, the following table is going to be used as a reference.

PHP Framework improvements		
Section	Improvement	Effect
Laravel	Implementation of Vue components	Facilitate the upgrading of the User Interface into a Reactive version

Table 4.8: PHP Framework improvements

#### 4.2.3.2 PHP Project Structure

PHP is known to be one the most controversial programming languages over the world. One of the reasons of this feeling is the fact that PHP projects, if they are not managed correctly, are very probable to become an organizational chaos [91][92].

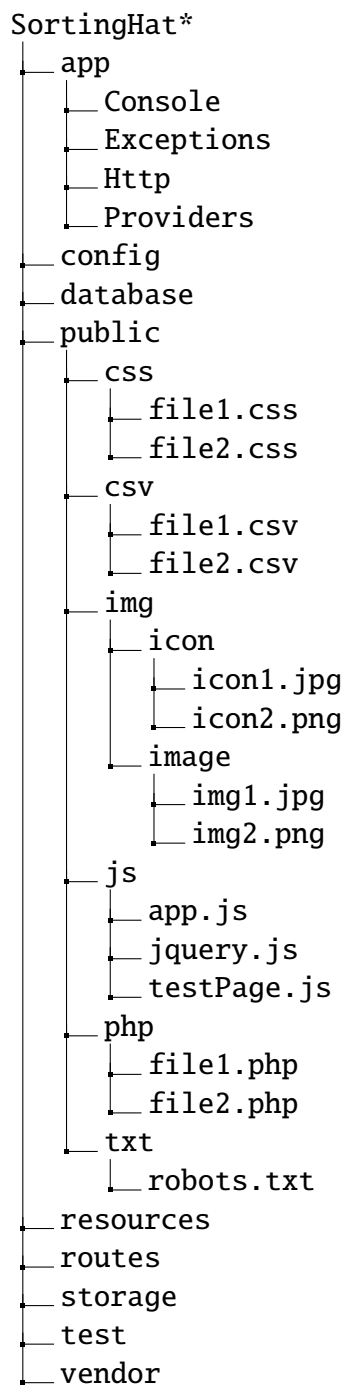
As a consequence, maintaining a proper PHP Project Structure is a key feature of any high quality Web Application based on PHP.

#### File Structure

The main issue according to the File Structure of the PHP Project Structure, or Laravel Project Structure, is the fact that inside the folder *public* there are files of every type with no organization. There are *php*, *css*, *png*, *jpg*, *pptx* and *txt* files.

In order to solve this situation, the solution is straightforward. A folder for each type of file need to be created inside *public*. In the case of the images, in which there are several types and extensions, an additional organizational division could be made in order to separate the common images and the icons.

Finally, when all these organizational improvements are accomplished, the PHP Project Structure should be similar to the following diagram.



Furthermore, the ppx file should be removed due to the fact that it is not used in Sorting Hat\*.

### Code Structure

According to the PHP files, they are located in two different folders. These folders are *public* and *resource/views*. The files located in the first folder are normal PHP files. How-

ever, the files located inside *resource/views* are of the type *.blade.php*, which is a special type of file for Laravel Framework.

Both of these types of files have different issues according to the Code Structure. Consequently, each type of file needs a different suggested improvement in order to enhance the Code Structure of the whole PHP code of Sorting Hat\*.

### **Common PHP files**

The main issue regarding to the PHP files located in *public* is the fact that the code inside the files is no refactored. This situation gets to the point that there is no method inside the files, even inside *querystats.php* which has 450 lines of code.

As a consequence of this situation, each of the files has to be divided in several methods, each of them being a different core part of the execution of the program. As a result of this measure, the complexity of the PHP files will greatly decrease and its readability will increase too.

### **Blade files**

Regarding to the *.blade.php* files, their main issue is the fact that there is *HTML*, *PHP* and the whole *CSS* file of each page placed together in the same file. As a consequence of this situation, it is almost impossible to understand the *blade* files due to its unnecessary complexity.

In order to solve this situation and improve the readability of the *blade* files, first of all the *CSS* code should be refactored from the *blade* files into new *CSS* files located in the *CSS* folder. These *CSS* files should be located in the folder which has been created in the previous section in order to enhance the File Structure.

As well as the *CSS* files, the *PHP* code located inside the *blade* files should be refactored into new files in order to improve readability and promote the reuse of code.

Finally, in order to enhance the understanding of the code and the execution flow of Sorting Hat, descriptive comments in English should be included into the new *CSS* and *PHP* refactored files as well as in the *HTML* code located in the *blade* files.

### **Summary**

In order to outline the main proposed improvements of the previous section, the following table is going to be used as a reference.



PHP Project Structure improvements		
Section	Improvement	Effect
File Structure		Efficient development
	A different folder is created for each type of file in order to group the files by their type	Efficient debugging
		Speed the navigation through the project
	The ppx is removed because it is useless	Remove useless elements from the project
Code Structure	PHP files are divided into different methods	Increase of the readability of the code
	Refactor the <i>CSS</i> code from the <i>blade</i> files	Efficient development
	Refactor the <i>PHP</i> code from the <i>blade</i> files	Efficient debugging
	Include descriptive comments in the refactored files as well as in the <i>blade</i> files	Promote the reuse of code

Table 4.9: PHP Project Structure improvements

#### 4.2.3.3 PHP Code Style

According to the PHP Code Style, the standard PSR-2 is going to be used as a reference. If in future development any issue arises according to the Code Style of PHP, this standard should be used in order to deal with it [93].

To start with, there are two issues according to the inconsistency of the line breaks. One of them is the inconsistency according to the amount of line breaks used for separating blocks of code.

In order to enhance the coherence with the line breaks, the amount of line breaks used for separating blocks of code should be two if they separate totally different blocks of code and one if they are related lines of code. However, no more than two break lines should be used.

Furthermore, the other issue related to the line breaks is the amount of line breaks used for separating the comments to the code. No line break should be placed between

the comment and the code to which it refers.

Another issue of the Code Style which was noticed while the analysis was the inconsistency of the left indenting used. It should only be used four spaces as a left indent and it should be followed the hierarchy of the previous code. That is to say, if the line of code is in the same scope as the one below both of them should use the same left indentation.

Regarding to the use of spaces in the control structures, the issue was similar to the previous described. In this case, the inconsistency was the number of spaces used between the different elements which construct a control structure.

The improvement proposed is using one space after the control structure keyword and between the closing parenthesis and the opening brace. However, no space should be use after the opening parenthesis nor after the closing of the brackets. This suggestion is the same as the one proposed in the standard previously mentioned.

As a consequence of the implementation of these improvement, the readability of the code will greatly increase as well as the productivity of the developers.

Finally, as in the improvements proposed in the previous sections, all the comments written in Spanish should be translated in order to ease the incorporation of a non Spanish speaker into the project.

## **Summary**

In order to outline the main proposed improvements of the previous section, the following table is going to be used as a reference.

PHP Code Style improvements		
Section	Improvement	Effect
Line Breaks	Only one line break between lines of code if they are related	Efficient development
	Only two line breaks between block of codes if they are not related	Efficient debugging
	No line break between a comment and the code to which it refers	Enhance readability
Indenting	Use only four spaces as left indenting	Efficient development
	Be consistent with the left indenting with the previous code	Efficient debugging Enhance readability
Control Structures	One space after the keyword and between the closing parenthesis and the opening brace	Efficient development Efficient debugging
	No space after the opening parenthesis nor after the closing of the brackets	Enhance readability
Comments Language	All the comments should be translated into English	Eases the incorporation of non Spanish developers to the team

Table 4.10: PHP Code Style improvements

#### 4.2.4. Proposed User Interface improvements

##### 4.2.4.1 User Interface Framework

The Framework used for developing the User Interface is Bootstrap. The main purpose of Bootstrap in Sorting Hat, which is one of the main features of this Framework, is implementing a responsiveness User Interface [67].

However, even though it is tried to implement a responsive User Interface in Sorting Hat\*, it is not correctly implemented. This can be clearly seen in the analysis of the Responsiveness of the User Interface.

Consequently, the solution of this problem is straight forward but not easy to implement. The implementation of Bootstrap to Sorting Hat\* should be modified in order to accomplish the correctly implementation of a responsiveness User Interface.

This improvement is crucial for the increase of the overall quality of Sorting Hat\*. If the User Interface of Sorting Hat\* is upgraded to a full responsive User Interface, the User Experience will be of much higher quality.

### Summary

In order to outline the main proposed improvements of the previous section, the following table is going to be used as a reference.

User Interface Framework improvements		
Section	Improvement	Effect
Bootstrap	Upgrade the User Interface to a fully responsive version using the Framework Bootstrap	Enhance of the User Experience

Table 4.11: User Interface Framework improvements

#### 4.2.4.2 CSS

##### CSS Project Structure

According to the improvements related to the Project Structure of the CSS files of Sorting Hat\*, they are going to be divided into the improvements for the File Structure and for the Code Structure.

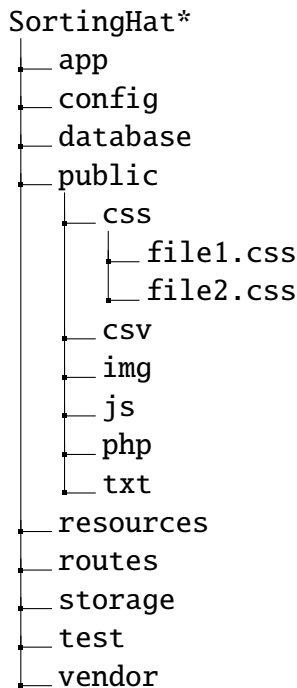
##### File Structure

The main issue with respect to the File Structure is the fact that there is a great amount of different types of files stored together in the folder *public*. This is the same issue as the one described in the previous section.

In order to solve this problem, the improvement proposed in the preceding section is going to be used. It consist on creating a specific folder for the CSS files inside the folder *public* in order to store there the CSS files.

Furthermore, all the CSS files which were located in *public* has to be moved to *public/css* and inside this folder only CSS files should be stored.

The structure which this structure could have is similar to the following diagram.



### Code Structure

In the case of the Code Structure more issues were discovered during the analysis. The main issue was regarding to the absence of refactored code.

To start with, the CSS code inside the HTML code should be refactored into new CSS files with descriptive names. Additionally, the CSS code from the independent CSS file should also be refactored with the same rules.

Furthermore, this refactoring should be done in a way in which it promotes the reuse of CSS code. The refactoring of the CSS code could follow the approach *SMACSS*, Scalable and Modular Architecture for CSS. The main objective of this CSS architecture is reducing the amount of code needed as well as facilitating code support. *SMACSS* promotes the division of the CSS files into five different categories: *Base Rules*, *Layout rules*, *Modules Rules*, *State rules* and *Theme rules* [94].

This is the most important measure regarding the CSS Code Structure. If this measure is correctly implemented the overall quality of the code will greatly increases.

When this refactoring is been accomplished the duplicated CSS code and the unnecessary lines of code commented should be removed.

Moreover, descriptive comments in English should be included in the CSS code in order to enhance the readability and understandability of the CSS code.

## Summary

In order to outline the main proposed improvements of the previous section, the following table is going to be used as a reference.

CSS Project Structure improvements		
Section	Improvement	Effect
File Structure	Inside <i>public</i> a folder called <i>css</i> is created to store the CSS files	Efficient development Efficient debugging
	Move the CSS files from <i>public</i> to <i>public/css</i>	Speed the navigation through the project
Code Structure	Refactor the CSS code from the HTML code	
	Refactor the code from the independent CSS file	
	The refactoring should promote the reusing of CSS	Efficient development Efficient debugging
	Delete duplicate code	
	Delete unnecessary code commented	
	Add descriptive comments	

Table 4.12: CSS Project Structure improvements

## CSS Code style

Regarding to the CSS Code Style, several issues were described during the analysis performed in the previous section. In order to design the improvements for these issues, the Google CSS Style Guide is going to be used as reference [72].

To start with, one of the main problems is the inconsistency of the amount of line breaks used for separate blocks of CSS code. The solution to this situation is forthright, it is using just one line break to separate different blocks of CSS code.

Another issue which was found during the analysis is the fact that the indenting is

inconsistent. In some cases spaces are used while in other cases tabs are the ones used. In order to solve this issue only two spaces should be used as left indenting. If this measure is implemented the overall readability of the file will greatly increase.

Furthermore, it was noticed that in some cases Shorthand Properties were used while in other cases that was not the case. In order to reduce the amount of code of the CSS files and to be consistent along the project, Shorthand Properties should be used in all the rules in which this approach can be applied.

According to the Capitalization, one problem was detected during the analysis. This problem was the fact that in most of the cases lower case was used, however in some rules upper case was the one used. The solution to this problem is to use only lower case. As a consequence, the consistency of the code along all the files will greatly increase enhancing the readability of them.

Finally, the last issue found in the analysis regarding to the CSS Code Style was the inconsistency of the separation between the last selector and the opening brace. In most of the cases only one space is used. Nonetheless, in other cases no space is used or just a line break is used. In order to solve this situation and maintain a consistency, only one space should be used to separate the selector from the opening brace.

In the case that in future developments more issues arise related to the CSS Code Style, the previously mentioned Style Guide of Google should be used as a reference in order to solve them.

## **Summary**

In order to outline the main proposed improvements of the previous section, the following table is going to be used as a reference.

CSS Code Style improvements		
Section	Improvement	Effect
Line Breaks	Use only one Line Break to separate CSS code blocks	Enhance readability
		Efficient development
		Efficient debugging
Indenting	Use only two spaces as left indenting	Enhance readability
		Ease the understanding of the scope of the CSS rules
		Efficient development
Shorthand Properties	Use Shorthand Properties in all the rules which accept this approach	Efficient debugging
		Reduce the amount of code needed
		Ease the understanding of the scope of the CSS rules
Capitalization	Use only lower case for all the CSS files	Efficient development
		Efficient debugging
		Enhance readability
Declaration Block Separation	Only one space used for separating the last selector and the opening brace	Efficient development
		Reduce the amount of code needed
		Efficient debugging

Table 4.13: CSS Code Style improvements



### 4.2.4.3 HTML

In this section all the proposed improvements for the HTML of Sorting Hat\* are going to be described.

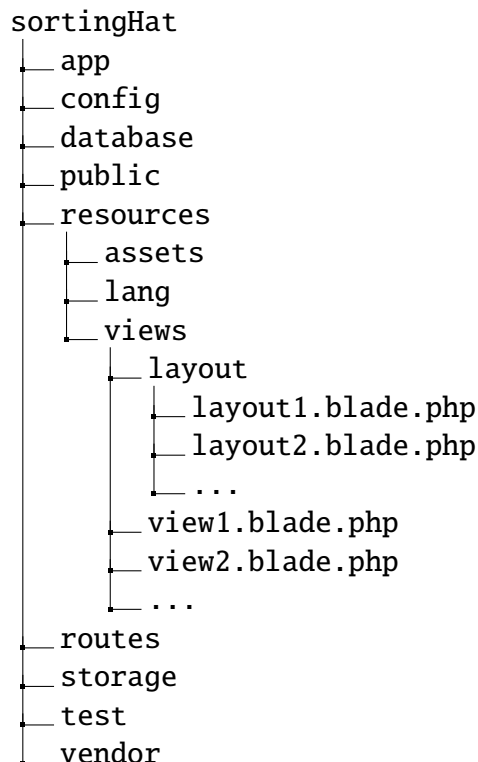
#### HTML Project Structure

According to the improvements proposed for the HTML code belonging to Sorting Hat\*, they are going to be divided into two sections: the improvements for the File Structure and the improvements for the Code Structure.

##### File Structure

With regards the File Structure of the HTML, the main issue is the fact that there is no folder for storing the Laravel Layouts. The solution to this situation is very simple. A new folder with the name *layouts* should be created inside the folder *resources/views*.

Once that the implementation of this improvement is accomplished, the File Structure should be similar to the following diagram.



As a consequence of the implementation of this improvement, the efficiency while implementing Laravel Layouts will increase.

## **Code Structure**

Regarding to the Code Structure of HTML, the main issue is the absence of the reuse of HTML code. The solution to this issue is the use of the Laravel Layouts.

Laravel Layouts are HTML templates which could be used by views with no need to rewrite all the code again. Because of this, the use of Laravel Layouts is the improvement proposed for solving the absence of the reuse of code. As a consequence, the readability of the code will greatly enhance as well as the quality of the code.

Furthermore, another important problem with regards to the Code Structure is the fact that there is PHP, CSS and JavaScript write directly inside the HTML. This is a known very bad practise.

The solution to this situation is refactoring all the PHP, CSS and JavaScript code into new files which will be imported to the HTML, but not written inside them. These new files refactored have to follow the improvements suggested previously.

In addition to this, in the analysis was noticed that the JavaScripts files were loaded in the header of the HTML files. This is also known to be a bad practise due to the fact that it could lead to the increasing of the time needed to load the User Interface.

In order to solve this problem, the JavaScript files should be imported in the bottom of the body.

Another issue regarding to the import of JavaScript files is that is some HTML files JavaScript are imported but they are not used. In order to settle this problem the imports of the files which are not used should be removed.

Finally, as in the previous sections, it is highly recommended to add descriptive comments in English in order to enhance the readability of the whole code.

## **Summary**

In order to outline the main proposed improvements of the previous section, the following table is going to be used as a reference.

HTML Project Structure improvements		
Section	Improvement	Effect
File Structure		Efficient development
	Create the folder <i>layout</i> inside <i>resources/views</i> to store the Laravel layouts	Efficient debugging
		Speed the navigation through the project
Code Structure	Implement Laravel Layouts for reusing HTML code	
	Refactor all the PHP, JavaScript and CSS code inside the HTML code into new files	Efficient development
	Import the JavaScript files in the bottom of the body	Efficient debugging
	Remove the import of files which are not used	
	Add descriptive comments	

Table 4.14: HTML Project Structure improvements

## HTML Code Style

In order to be as precise as possible with the improvements proposed for the HTML Style Guide, the Google Code Style for HTML is going to be used as reference [72].

With respect to the HTML Code Style, several issues were discovered during the analysis. The first issue which was noticed is the inconsistency of the amount of line breaks used for separate the different blocks of HTML code.

In order to solve this problem, it is recommended to use only one line break to separate code blocks. In this is done, the consistency of the HTML will raise.

In the analysis performed in the previous section it was noticed that the indenting of the HTML code was inconsistent. The inconsistency was related to the amount of indent used and the use of spaces or tabs.

The solution to this inconsistency is the used of only two spaces in order to indent.

Once that this is done, the consistency of the code will enhance as well as the readability of it. Furthermore, as this is the same suggestion as to the Code Style of the CSS code, the overall consistency of the project will increase too.

In addition to this, it was also noticed that some images had no meaningful *"alt"* attribute. This attribute is essential in the case that the images are not correctly loaded into the User Interface.

In order to solve this problem, a meaningful *"alt"* attribute should be added to all the images added in the HTML. Additionally, the images which have already an acceptable *"alt"* attribute should be checked and if it is possible be updated to a more meaningful version.

Another issue, which is less important but if it is fixed the redundancy in the code will decrease, is related to the *"type"* attribute of some files. Since HTML5, if the files imported are JavaScripts or CSS files its type is not needed to be specify.

Consequently, the *"type"* attribute should be removed from the imports of CSS and JavaScript files.

Finally, the last issue regarding the Code Style of the HTML code is related to the quotation marks. It is recommend to use double quotation marks, `" "`, when quoting attribute values. However, in the analyzed code it was discovered that single quotation marks, `' '`, were the ones used for this purpose. The solution is straight forward, all the single quotation used for this purpose need to be changed to double quotation marks.

In the case that in future developments more issues arises related to the HTML Code Style, the previous mentioned Style Guide of Google should be used as a reference in order to solve them.

## **Summary**

In order to outline the main proposed improvements of the previous section, the following table is going to be used as a reference.

HTML Code Style improvements		
Section	Improvement	Effect
Line Breaks	Use only one line break to separate blocks of HTML code	Decrease the size of the file
		Efficient development
		Efficient debugging
Indenting	Use only two spaces as left indenting	Increase of the consistency of the whole project
		Efficient development
		Efficient debugging
Images	Add a meaningful <i>"alt"</i> attribute to the images	Enhance the User Experience in the case that the images are not correctly loaded
"Type" Attribute	Delete the <i>"type"</i> attribute from the imports of CSS and JavaScript files	Decrease the size of the file
		Enhance the readability of the code
Quotation Marks	Change the single quotation marks to double quotation marks when quoting attribute values	Enhance the readability of the code

Table 4.15: HTML Code Style improvements

#### 4.2.4.4 Responsiveness

According to the Responsiveness of Sorting Hat\*, during the analysis several issues were noticed. Their improvements are going to be divided into two different sections: computer Responsiveness and mobile Responsiveness. Inside each of them, the different issues regarding to the Responsiveness are going to be assessed.

The improvements proposed which are going to be described in the following sections are a high level improvements. No description about the code is going to be given because that is part of the implementation of the improvements of Sorting Hat\*.

## **Computer Responsiveness**

The improvements for implementing Computer Responsiveness in Sorting Hat\* are going to be divided into the different pages which have the most important issues regarding to Computer Responsiveness.

### **Log in**

Regarding to the main inconsistency in the Log in page is the background size. It is not resized nor adapted to the full screen for larger screens. As a consequence, there is an area of the screen in which instead of the image there is a solid gray background.

The obvious solution to this situation is resizing the background image or specifying the size of the background image to fit all the screens. The objective of this improvement is that no background is left without the background image and avoiding that some area of the background is filled with the default gray color background.

### **Home**

Regarding to the Home page, the inconsistency related to Computer Responsiveness is the fact that the images in small screens are not fully displayed, they are cut. This problem should be fixed in order to offer to the user the best User Experience possible, and cutting the images by half is not the best practice to accomplish it.

In order to solve this situation, the improvement solution is similar to the one proposed in the previous section. The images should be resized or moved in order to fit in the screen, no matter of the size of the screen. In this improvement is performed, the User Experience of the user will greatly increase.

### **Account**

The problem in the Account page is different from the ones of the previous sections. In this case the problem is related to the bottom bar which appears in many pages, such as at the Account page.

The problem is that the lower bar is not fixed to the bottom of the page, where it should be. Instead, depending of the size of the screen it is displayed in one place or another.

In order to solve this situation, the bottom bar should be fixed to the bottom of the page no matter the size of the screen. If this is accomplished, the overall consistency of the page will greatly increase and with it the quality of the User Interface of Sorting Hat\*.

## **Mobile Responsiveness**

The structure of the proposed improvements for the mobile Responsiveness is similar to the structure of the previous section. The proposed improvements for Sorting Hat\* are

going to be grouped by the page at which they should to be implemented.

### **Log in**

The issue regarding to the Log in page is the same as the one explained in the computer Responsiveness improvements. The issue is that the background image does not cover the totality of the screen.

As a consequence, in order to solve this problem the size of the background should be resized to fit all the screen sizes or to be fixed to a size which fit all the screens.

### **Home**

In the case of the Home page, the issue related to computer Responsiveness is also similar to the one explained in the computer Reactiveness improvements. The problem is the fact that the images are not resized correctly to the different screen sizes so they are moved in the screen inconsistently to fit the screen size.

This situation needs to be solved in order to enhance the User Experience of Sorting Hat\*. In order to accomplish it, the images need to be resized to fit the screen size maintaining a consistent layout structure.

### **Account**

Finally, the last problem which needs to be solved according to the mobile Reactiveness is related to the Account page. The issue in this page is related to the layout used in the User Interface.

In this case, the User Interface is just an adaptation of the computer User Interface. The result is that only one quarter of the screen is used.

In order to solve this situation, the layout used in this page should be changed in order to optimize the space available providing a more appealing and useful User Interface.

### **Summary**

In order to outline the main proposed improvements of the previous section, the following table is going to be used as a reference.

Responsiveness improvements		
Section	Improvement	Effect
Computer Responsiveness	At the Log in page resize the size of the background image or fix it to fit all the screen sizes	Enhance the User Experience
	At the Home page resize or move the images in order to fit the images in all the screen sizes	
	At the Account page fix the down bar to the bottom of the screen no matter the size of the screen	
Mobile Responsiveness	At the Log in page resize the size of the background image or fix it to fit all the screen sizes	Enhance the User Experience
	At the Home page resize or move the images in order to fit the images in all the mobile screen sizes	
	At the Account page change the layout used for displaying the elements to use more of the available screen space	

Table 4.16: Responsiveness improvements

#### 4.2.4.5 User Interface Design

In this section all the proposed improvements for the issues regarding the User Interface Design are going to be explained.

These improvements are going to be grouped in different sections which are related to the design principle of the problem.



## **Clarity**

Related to the clarity of the User Interface, several issues were noticed while the analysis performed in the previous section. One of this issues is related to the figures displayed in the Test page.

The problem of the Test page is the fact that the amount of decimal digits which are displayed are excessive to what the user needs. The amount of decimal digits should be reduced to just two digits in order to be as clear as possible.

The other problems are related to the Dashboard page. The main problem of this page is that in the graphs next to the "%" symbol there is no amount. As a consequence, the user gets no information from the graphs. The solution is straight forward. The amount to which the graphs refer to should be displayed.

Furthermore, there is another issue at the Dashboard page which needs to be solve. If the cursor is placed over the last graph of the page the number of decimal digits which are displayed is excessive. In order to be consistent with the improvement proposed previously, only two decimals digits should be displayed.

## **Color Palette**

According to the Color Palette, there is only one mayor issue in Sorting Hat\*. This issue is related to the diagram used at the bottom of the Home page.

The issue is the fact that the Color Palette used for this diagram is not the same nor similar to the Color Palette used in the rest of Sorting Hat\*. In order to solve this situation, the colors used in this diagram should be changed in order to be similar to the ones used in the rest of Sorting Hat\*.

## **Consistency**

According to the Consistency of Sorting Hat\*, several issues were discovered during the analysis. The main issue is regarding to the width of the text paragraphs of the Home page. Each of the paragraphs have different width and they are not placed symmetrically respect to the center, creating the sense of inconsistency.

In order to solve this problem the width of the paragraphs and its location in the screen should be modified. The width is recommended to be the same for both paragraphs as well as placing them in a symmetric way so as to enhance the sense of consistency.

Furthermore, another issue which was noticed was related to the navigation bar. The font and size of the text of the navigation bar depends on the page in which the user is. This needs to me solved using the same font style and size for the toolbar of all the pages. If this is done, the overall quality of the User Interface will greatly increases.

## **Effectiveness**

The Effectiveness issues are one of the most important issues to solve because they are the ones which harm the most to the quality of the User Experience. The first problem is related to the Home page. The problem is that one of the texts which appears over the images cannot be properly read, the color of the font is white as the image. The solution is easy, the image needs to be changed to no which allows the text to be read with no problem.

Another important problem is the fact that the page selected in the navigation toolbar does not change, no matter in which page the user is. This can be easily solved. The page in which the user is should be marked in the navigation toolbar.

## **Error Messages**

According to the Error Messages, the problem is very similar in two different pages of Sorting Hat\*. These pages are the Log in and the Create Certificate page.

Regarding to the Log in page, if any of the input parameters to register or to log in into Sorting Hat\* is introduced incorrectly, the error message displayed is meaningless and general. In order to solve this problem, a customized error message for each of the input fields should be created. As a consequence, in the case that the user gets an error message from Sorting Hat\* he will know what to do in order to correct the error.

The other problem is at the Create Certificate page. The problem is the same, in any of the fields are introduced incorrectly a general error message is showed. To change this situation the same improvement as the one previously proposed should be implemented. A customized error message for each of the input fields should be created.

## **Symmetry**

Relate to the Symmetry inconsistencies, the first inconsistency is located at the Register page. In this page, the form boxes used to register or log in into Sorting Hat\* are placed asymmetrically in the screen. The solution is straight forward, move them in the screen in order to place them symmetrically to the center of the screen.

Furthermore, there are two more issues at the Home page. The first of these issues is the placement of the diagram located at the bottom of the page. This diagram is closer to one side of the screen. To solve this issue, the diagram should be placed in the center, with equal free space respect to each of the sides of the screen.

Finally, the last issue has been previously described. This issue is the inconsistency of the placement of the text paragraphs at the Home page. The solution is just placing each of the paragraphs symmetrically to each other having as a reference the center of the screen. If these improvements are implemented the overall consistency of Sorting Hat\* will be highly increased as well as the overall quality of the User Experience.

## Summary

In order to outline the main proposed improvements of the previous section, the following table is going to be used as a reference.

User Interface Design improvements		
Section	Analysis	Effect
Clarity	At the Test page use only two decimal digits when displaying the results	
	At the Dashboard page display the numbers next to the "%" symbol	Enhance the User Experience
	At the Dashboard page use only two decimal digits when displaying the results of the graphs	
Color Palette	At the Home page change the color palette of the diagram located at the bottom of the page to be consistent to Sorting Hat* color palette	Improve consistency across Sorting Hat* Enhance of the User Experience
	At the Home page set the width of all the text paragraphs to the same value	Improve consistency across Sorting Hat*
Consistency	At the Home page set the text paragraphs in a symetryc layout to improve	Enhance of the User Experience
	Use the same style and size for the font of the Navigation Toolbar for all the pages	
Effectiveness	At the Home page change the image which does not let the text to be read	Enhance of the User Experience
	Mark in the Navigation Toolbar the page in which the user is	

User Interface Design improvements		
Section	Analysis	Effect
Error Messages	Create a customized error messages for each of the input fields of the Log in and Create Certificate	Faster user error solving Enhance of the User Experience
	At the Log in page place the log in and register forms symmetrically each other from the center of the screen	
Symmetry	At the Home page place the diagram at the bottom of the page in the center of the screen	Enhance of the User Experience
	At the Home page place the text paragraphs symmetrically the one from each other	

Table 4.17: User Interface Design improvements

## 5. PLANNING AND BUDGET

In this chapter the planning used in order to accomplish this Bachelor Thesis will be explained. In addition to the planning, a budget related to the planning will be described too.

### 5.1. Planning

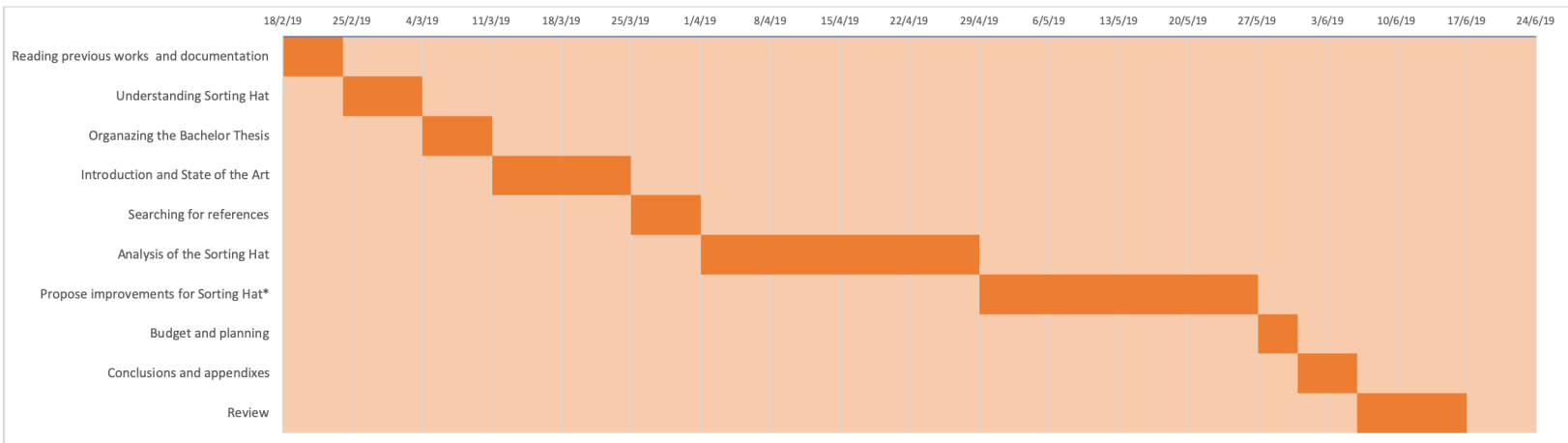
The duration of the Bachelor Thesis is around four months of work. Therefore, the starting point of the Bachelor Thesis is the day **18 of February of 2019** and the last day of the review of the Bachelor Thesis is the **17 of June of 2019**.

Once that the whole duration of the Bachelor Thesis has been outlined, a general description of the different parts in which the whole planning has been divided is going to be given.

- Reading previous works and documentation: during this period several Bachelor Thesis were read in order to have an idea about the desired structure of a Bachelor Thesis.
- Understanding Sorting Hat: in this period the code of Sorting Hat was studied as well as its functionalities.
- Organizing the Bachelor Thesis: the structure of the Bachelor Thesis is determined as well as the approach which is going to be taken in order to write this Bachelor Thesis.
- Introduction and State of the Art: the sections of the Introduction and State of the Art are written.
- Searching for references: references and standards are searched in order to have a first impression of what the analysis and improvements could look like.
- Analysis of the Sorting Hat: the analysis of Sorting Hat is accomplished.
- Propose improvements for Sorting Hat\*: the improvements proposed for Sorting Hat\* are described.

- Budget and planning: the section related to the budget and planning is written.
- Conclusions and appendixes: the sections regarding to regulatory framework, conclusions and appendixes are written.
- Review: a review process of the Bachelor Thesis in order to find errors is performed.

Fig. 5.1. Planning of the Bachelor Thesis



## 5.2. Budget of the project

In this section the budget related to the material is going to be outlined as well as the human resource costs. Furthermore, at the end of this section both costs will be sum in order to display a table with the total costs of this Bachelor Thesis.

### 5.2.1. Material Cost

All the costs related to the software applications or the hardware devices which are used during the development of a project should be included into the section of *Material Cost*. Consequently, the following table outlined the Material Cost related to this Bachelor Thesis.

Material Cost				
Type of material	Name of material	Unitary price	Lifetime for material (months)	Attributable cost
Software	MySQL	0	N/A	0
	MySQL Workbench	0	N/A	0
	PHPStorm	0	N/A	0
	XAMPP	0	N/A	0
	Overleaf	0	N/A	0
Hardware	MacBook Pro Intel Core i5 8 GB	960 €	48	80 €
	Samsung Monitor	120 €	36	13.14 €
	Xiaomi Mi 5s	220 €	24	36.4 €
Perishable Materials	Perishable Materials	40 €	4	40 €
<b>Total Cost</b>				<b>169.53 €</b>

Table 5.1: Material Cost



### 5.2.2. Human Resource Cost

The Human Resources costs are related to the costs of the employees. In this case, as all the Bachelor Thesis has been related to the analysis of Sorting Hat, only the salary of an analyst is going to take into account in order to calculate the Human Resource cost.

Human Resource Cost			
Employee Role	Cost (Eur/Hour)	Number of hours	Overall cost
Analyst	20 €	510	10200 €
<b>Total Cost</b>			<b>10200 €</b>

Table 5.2: Human Resource Cost

### 5.2.3. Total Cost

Finally, the Total Cost of the project is the sum of the both the Material and the Human Resource costs, taking into account that a profit of 20% is going to be obtained. After that, the Total Cost of the project is going to be calculated taking into account the 21% of IVA.

Total Cost	
Material Cost	169.53 €
Human Resource Cost	10,200 €
Direct Cost	10,369.53 €
Indirect Cost (5% of Direct Cost)	518.47 €
Total Cost without IVA	10,888 €
IVA Cost	2,286.48 €
<b>Total Cost</b>	<b>13,174.48 €</b>

Table 5.3: Total Cost

## 6. REGULATORY FRAMEWORK

As it has been described throughout this Bachelor Thesis, cybersecurity is one of the most important issues in the recent years in the technology area and its importance will continue raising.

As a consequence of the expansion of Internet and all the technologies around it, such as smart-phones or computers; new threats which could not be imagined before this situation has appeared forcing to the companies and state agencies to take measures in order to solve this threats in an efficient way.

In the beginning of this situation, there was a lack of legislation on cybersecurity in most of the countries. However, as the importance and relevance of cybersecurity grew new regulation was created in order to assessed the new challenges which were arising.

In this chapter, the most important regulations on cybersecurity at a country and European level will be explained in order to gain a further understanding of the regulatory framework of cybersecurity.

### 6.1. Network and Information Security Directive

The Network and Information Security Directive, known as NIS, is part of the European's cybersecurity strategy. The objective of this directive is to improve the quality of cybersecurity throughout the European Union.

The Network and Information Security Directive was approved in 2016. Subsequently, all the members of the European Union begun to adopt national legislation which were aligned with the European directive. Furthermore, the European nations have some degree of flexibility from the European Union in order to design the legislation taking into account special national situations or the possibility of reusing national organizational infrastructures.

The NIS Directive is composed by three main parts which are going to be briefly described.

- National capabilities: the countries which are part of the European Union must have certain cybersecurity capabilities, such as a national CSIRT.
- Cross-border collaboration: a collaboration between the countries related to cross-

border is crucial feature of this directive.

- *National supervision of critical sectors*: the supervision of the critical sectors of each country must be made by the countries themselves in order to fulfill the security standards.

Moreover, the European Commission has a map showing the status of the transposition of the NIS Directive in all European nations in order to gain a fast insight about the current situation of its implementation[95].

## **6.2. Ley orgánica de Protección de Datos**

In November 2017 the new law called "Ley Orgánica de Protección de Datos" was approved in Spain. This new law was approved in November 2017 but it was not applied until May 2018. It was the replacement of the old former law regarding to the use of digital data.

One of the main changes is the removing of the concept of tacit consent in order to reinforce a more expressed action. In addition to this, the principle of transparency is one of the main concepts which this new law brings. This new concept refers to the fact that the users must be informed about the treatment of their data whenever it affects to them in a direct way or not.

Furthermore, it should be noted that "Ley Orgánica de Protección de Datos" also gives the possibility of the existence of self-regulation mechanisms in the public and private sectors[96].



Fig. 6.1. Summary of Ley orgánica de Protección de Datos

### 6.3. Ley de Conservación de Datos

The last law which is going to be described in this section is "Ley de Conservación de Datos". This law is from the Spanish government, as the former one, and its main objective is to regulate the obligation of the operators to keep the data obtained from the users.

Furthermore, another important concept of this law is the fact that the duty to transfer the obtained data to competent bodies whenever necessary prior judicial authorization for the purpose of investigation or prosecution of crimes is contemplated in this law.

This Law is applied to the location and data traffic on natural and legal persons as well as related data to them which is necessary in order to identify the registered user.

## 7. CONCLUSION

In this section the conclusions which have been obtained through the development of this Bachelor Thesis will be explained. Furthermore, the fulfilling of the initial objectives and the future approaches for Sorting Hat will be described.

### 7.1. Full-filling of the initial objectives

In the first chapter the initial goals for the development of this Bachelor Thesis were described. In this section the fulfilling of these objectives is going to be determined.

- Database analysis: A full analysis of the Database used for managing Sorting Hat was correctly accomplished.
- Business Logic analysis: An analysis was performed to the PHP and JavaScript code, which are the components of the Business Logic.
- User Interface analysis: The User Interface was subject of an analysis taking into account features such as the Responsiveness and Design inconsistencies among others.
- Database proposed improvements: Several proposals for improving the Database as well as fixing the issues found in the analysis will be described. Furthermore, a diagram of the resulting database for further understanding of the updated system.
- Business Logic proposed improvements: Proposals for improving and enhancing the efficiency of Sorting Hat Business Logic have been given taking into the account the issues found during the analysis.
- User Interface proposed improvements: Proposed improvements as well as guidelines for solving future problems regarding the User Interface of Sorting Hat has been successfully given.

## 7.2. Summary of the results

In order to summarize the result of the analysis and the proposal of the improvements, the following tables with all the results of the analysis as well as the proposed improvements are going to be displayed.

### 7.2.1. Database Results

In this section the analysis' results and the proposal of the Database improvements are going to be displayed in the following table.

Database Results			
Module	Section	Analysis	Improvement
Project Structure	File Structure	Only one file in the project folder	Rename root folder from "Base de datos" to "Database"
			Create a new folder for each Database
Project Structure	Code Structure	Only one file of 7173 lines of code	Add to the Database's folder the current version of it
			Inside the Database's folder create one folder for <i>development</i> and other folder for <i>release</i>
Code Style	Table Naming	Inconsistency of tables' names	Inside <i>release</i> divide the files regarding to the different SQL statements or elements
			Refactor the original SQL file to fulfill the requirements of the proposed File Structure
Code Style	Table Naming	Inconsistency of tables' names	Meaningful and brief names
			Use of collective names when possible
			Singular form for table's names
			Avoid the concatenation of tables' names
Code Style	Table Naming	Inconsistency of tables' names	Avoid using prefixes and suffixes
			Use Snake Case as the naming standard

Database Results			
Module	Section	Analysis	Improvement
			Meaningful and brief names
			Use of collective names when possible
			Singular form for table's names
Code Style	Column Naming	Inconsistency in columns' names	Avoid the concatenation of tables' names
			Avoid using prefixes and suffixes
			Use Snake Case as the naming standard
		All the columns referencing to <i>"proficiencylevel"</i> table have the same values	Remove the table <i>"proficiencylevel"</i> and all the columns referencing to this table
	Trivial tables	<i>"profilesset"</i> , <i>"user_has_profile"</i> , <i>"user_has_targetprofile"</i> , <i>"tag"</i> , <i>"competency"</i> , <i>"profile_with_proficiencylevel"</i> , <i>"user_selected"</i> and <i>"profilesset_has_profile"</i> has no data stored in them	Remove <i>"profilesset"</i> , <i>"user_has_profile"</i> , <i>"user_has_targetprofile"</i> , <i>"tag"</i> , <i>"competency"</i> , <i>"profile_with_proficiencylevel"</i> , <i>"user_selected"</i> and <i>"profilesset_has_profile"</i>
		<i>"LowerProficiencyLevel_id"</i> and <i>"UpperProficiencyLevel_id"</i> have the same values in all the tables	Remove <i>"LowerProficiencyLevel_id"</i>
Relational Data Model		<i>"remember_token"</i> belonging to the table <i>"user"</i> has no purpose nor data	Remove <i>"remember_token"</i> belonging to the table <i>"user"</i>
	Trivial columns	<i>"NCWF_SpecialityArea_NCWF_Category_id"</i> is not needed in table <i>"ncwf_workrole"</i> . With <i>"NCWF_SpecialityArea_id"</i> the category can be accessed	<i>"NCWF_SpecialityArea_NCWF_Category_id"</i> is removed from table <i>"ncwf_workrole"</i>
		<i>"Type"</i> column belonging to <i>"profile"</i> contains the same data for all the instances	<i>"Type"</i> column belonging to <i>"profile"</i>
	Similar columns	<i>"SourceReference"</i> and <i>"Source"</i> columns from <i>"knowledge"</i> , <i>"task"</i> , <i>"skill"</i> , <i>"ability"</i> , have the same values	<i>"SourceReference"</i> column from <i>"knowledge"</i> , <i>"task"</i> , <i>"skill"</i> , <i>"ability"</i> are deleted
		<i>"created_at"</i> and <i>"updated_at"</i> belonging to <i>"user_has_certification"</i> are already stored at <i>"certification"</i>	<i>"created_at"</i> and <i>"updated_at"</i> belonging to <i>"user_has_certification"</i> are deleted
	Inappropriated column format	The columns <i>"Profile_id"</i> could contain up to 11 digits when only 2-3 are needed	The column <i>"Profile_id"</i> is changed from containing 11 digits to 4 digits

Database Results			
Module	Section	Analysis	Improvement
Relational Data Model	Inappropriated column format	"NCWF_SpecialityArea_id", "NCWF_SpecialityArea_NCWF_Category_id" and "NCWF_Category_id" could contain up to 11 digits when only 2-3 are needed.	The columns "NCWF_SpecialityArea_id" and "NCWF_Category_id" are changed from containing 11 digits to 4 digits
		"Type" column belonging to "profile" replicates the data of the chosen type instead of referencing it	The column "type" from the table "profile" is changed from containing a varchar to an integer of 2 digits
	Trivial Primary Keys	The tables "knowledge", "task" and "skill" contain as primary key its identifier and its source. Only the identifier is needed because by definition it is unique	The tables "knowledge", "task" and "skill" get removed the column "Source" as part of their primary key
	Missing Foreign Keys	The tables "profile_has_knowledge", "profile_has_task", "profile_has_skill", "profile_has_ability", does not need the source and the proficiency's level as primary keys	"Source", "LowerProficiencyLevel_id" and "UpperProficiencyLevel_id" do not need to be part of the primary key of the junction tables "profile_has_knowledge", "profile_has_task", "profile_has_skill", "profile_has_ability"
		"user_has_workrole" is not linked with "user" nor "ncwf_workrole"	The column "type" from "profile" is set to be a foreign key to the table "profile_type"
	MySQL accounts	No password is defined for the root user	Establishing a password for the root user
Database Security		There is no backup to restore the actual Database if an adversity happens	
	Database Backup	There is no procedure to automatize backups  It is not clearly stated the period of time while a backup is valid until it is needed to create a new one	Protecting the access to root privileges

Table 7.1: Result of the analysis and the proposed improvements for the Database

### 7.2.2. JavaScript Results

In this section the analysis' results and the proposal of the JavaScript improvements are going to be displayed in the following table.



JavaScript Results			
Module	Section	Analysis	Improvement
Library	jQuery	JQuery Library is loaded two times in each HTML file	Load JQuery Library only once at the end of the HTML body
		JQuery is loaded in the header of the HTML	
Project Structure	File Structure	Two equal folders for storing the JavaScript files	Delete the folder called "javascript"
		Duplicated files	Delete the duplicated file called "jquery.js"
	Code Structure	Non descriptive file's name	Rename the file called "arboles.js" to "testPage.js"
		Unnecessary importation of the JavaScript files in several pages	Remove unnecessary imports
		Absence of JavaDoc comments	Implement JavaDoc comments to all the methods
Appearance of logs	Remove all the logs		
Code Style	Variable Naming	Inconsistency of variables' name	Consistency of variables' name is enforced following the camelCase naming standard
	Variable Declaring	Global variables declared between methods	Global variables declared between methods are moved to the top of the file
	Hardcoded String Variables	Several String comparisons are performed using hardcoded Strings in the middle of the method	The hardcoded Strings used in the comparisons should be declare at the top of file
	Comments Language	All the comments are written in Spanish	All the comments should be translated into English

Table 7.2: Result of the analysis and the proposed improvements for the JavaScript

### 7.2.3. PHP Results

In this section the analysis' results and the proposal of the PHP improvements are going to be displayed in the following table.

PHP Results			
Module	Section	Analysis	Improvement
Framework	Laravel	Laravel feature for the correct implementation of Responsible User Interfaces is not fully used	Implementation of Vue components
Project Structure	File Structure	Mixing of several types of files inside the folder <i>public</i> with no organization	A different folder is created for each type of file in order to group the files by their type
	Code Structure	No refactoring of the files	The pptx is removed because it is useless
			PHP files are divided into different methods

PHP Results			
Module	Section	Analysis	Improvement
Project Structure	Code Structure	Absence of descriptive comments	Include descriptive comments in the refactored files as well as in the <i>blade</i> files
		No organization inside <i>.blade.php</i> files	Refactor the <i>CSS</i> code from the <i>blade</i> files Refactor the <i>PHP</i> code from the <i>blade</i> files
Code Style	Line Breaks	Inconsistency of the amount of line breaks used for separate blocks of code	Only one line break between lines of code if they are related Only two line breaks between block of codes if they are not related
		Inconsistency of the amount of line breaks used for separate the comments	No line break between a comment and the code to which it refers
	Indenting	Inconsistency in the amount of left indenting used	Use only four spaces as left indenting
			Be consistent with the left indenting with the previous code
	Control Structures	No consistency in the use of spaces in the control structures	One space after the keyword and between the closing parenthesis and the opening brace
No space after the opening parenthesis nor after the closing of the bracket			
Comments Language	All the comments are written in Spanish	All the comments should be translated into English	

Table 7.3: Results of the analysis and the proposed improvements for the PHP

## 7.2.4. User Interface Results

In this section the different features of the User Interface which has been assessed during the analysis and the proposal of improvements are going to be summarized.

### 7.2.4.1 CSS Results

In this section the analysis' results and the proposal of the CSS improvements are going to be displayed in the following table.

CSS Results			
Module	Section	Analysis	Improvement
Framework	Bootstrap	Incorrect implementation of a responsive User Interface	Upgrade the User Interface to a fully responsive version using the Framework Bootstrap
Project Structure	File Structure	Mixing of CSS files with other type of files inside the folder <i>public</i> with no organization	Inside <i>public</i> a folder called <i>css</i> is created to store the CSS files
		CSS files outside the " <i>css</i> " folder where it is their expected place to be located	Move the CSS files from <i>public</i> to <i>public/css</i>
	Code Structure	Unnecessary lines of code commented	Delete unnecessary code commented
		Duplicated code	Delete duplicate code
		No refactoring of the files	Refactor the code from the independent CSS file
		Absence of descriptive comments	Add descriptive comments
CSS code written inside the HTML code	Refactor the CSS code from the HTML code The refactoring should promote the reusing of CSS		
Code Style	Line Breaks	Inconsistency of the amount of line breaks used for separate blocks of CSS code	Use only one Line Break to separate CSS code blocks
	Indenting	Inconsistency in the amount of left indenting used	Use only two spaces as left indenting
	Shorthand Properties	Absence of Shorthand Properties for some rules	Use Shorthand Properties in all the rules which accept this approach
	Capitalization	Inconsistency of the use of lower case	Use only lower case for all the CSS files
	Declaration Block Separation	Inconsistency of the separation between the last selector and the opening brace	Only one space used for separating the last selector and the opening brace

Table 7.4: Results of the analysis and the proposed improvements for the CSS of the User Interface

### 7.2.4.2 HTML Results

In this section the analysis' results and the proposal of the HTML improvements are going to be displayed in the following table.

HTML Results			
Module	Section	Analysis	Improvement
Project Structure	File Structure	There is no folder for storing Laravel Layouts	Create the folder <i>layout</i> inside <i>resources/views</i> to store the Laravel layouts

HTML Results			
Module	Section	Analysis	Improvement
Project Structure	Code Structure	Layouts are not created	Implement Laravel Layouts for reusing HTML code
		Existence of PHP, JavaScript and CSS code inside the HTML code	Refactor all the PHP, JavaScript and CSS code inside the HTML code into new files
		The scripts are loaded on the header of the HTML	Import the JavaScript files in the bottom of the body
		Scripts loaded in pages where they are not beign used	Remove the import of files which are not used
		Absence of descriptive comments	Add descriptive comments
Code Style	Line Breaks	Inconsistency of the amount of line breaks used for separate blocks of HTML code	Use only one line break to separate blocks of HTML code
	Indenting	Inconsistency in the amount of left indenting used	Use only two spaces as left indenting
		Inconsistency in the use of spaces or tabs for the indenting	
	Images	No meaningful <i>alt</i> attribute is added to the images	Add a meaningful " <i>alt</i> " attribute to the images
	"Type" Attribute	Specify the " <i>type</i> " attribute when importing a CSS or JavaScript file	Delete the " <i>type</i> " attribute from the imports of CSS and JavaScript file
Quotation Marks	Single quotation marks used for quoting attribute values	Change the single quotation marks to single quotation marks when quoting attribute values	

Table 7.5: Result of the analysis and the proposed improvements for the HTML of the User Interface

### 7.2.4.3 Responsiveness Results

In this section the analysis' results and the proposal of the Responsiveness improvements are going to be displayed in the following table.

Responsiveness Results			
Module	Section	Analysis	Improvement
Responsiveness	Computer Responsiveness	Log in page background does not fill the whole screen for large screens	At the Log in page resize the size of the background image or fix it to fit all the screen sizes
		Home images get cut if the screen is smaller, instead of been resized	At the Home page resize or move the images in order to fit the images in all the screen sizes

Responsiveness Results			
Module	Section	Analysis	Improvement
Responsiveness	Computer Responsiveness	The bottom bar of the Account page is not fix to the bottom of the page. Its position change depending on the size of the screen	At the Account page fix the down bar to the bottom of the screen no matter the size of the screen
	Mobile Responsiveness	Log in page background does not fit the whole screen	At the Log in page resize the size of the background image or fix it to fit all the screen sizes
		Log in page boxes to register and log in are placed one next to the other leving half of the screen empty	
		Home images are not resized correctly forcing to the text next to them to be moved below or over them	At the Home page resize or move the images in order to fit the images in all the mobile screen sizes
		At the Account page only one quarter of the whole screen is used	At the Account page change the layout used for displaying the elements to use more of the available screen space

Table 7.6: Results of the analysis and the proposed improvements for the Responsiveness of the User Interface

#### 7.2.4.4 User Interface Design Results

In this section the analysis' results and the proposal of the User Interface Design improvements are going to be displayed in the following table.

User Interface Design Results			
Module	Section	Analysis	Improvement
User Interface Design	Clarity	The figures displayed in the Test page are not clear	At the Test page use only two decimal digits when displaying the results
		No figures are displayed next to "%" at the DashBoard page	At the Dashboard page display the numbers next to the "%" symbol
		The figures displayed over the graphs are not clear	At the Dashboard page use only two decimal digits when displaying the results of the graphs
	Color Palette	Diagram at the Home page without any consistency to the color palette used	At the Home page change the color palette of the diagram located at the bottom of the page to be consistent to Sorting Hat* color palette
	Consistency	Inconsistency of the text paragraphs at the Home page	At the Home page set the width of all the text paragraphs to the same value  At the Home page set the text paragraphs in a symetryc layout to improve

User Interface Design Results			
Module	Section	Analysis	Improvement
User Interface Design	Consistency	Inconsistency in the font used for the Navigation Bar	Use the same style and size for the font of the Navigation Toolbar for all the pages
		Effectiveness	The text in the Home picture cannot be correctly read
	The Navigation Toolbar always shows the same tab as the one selected		Mark in the Navigation Toolbar the page in which the user is
	Error Messages	Meaningless error message at the Log in page	Create a customized error messages for each of the input fields of the Log in and Create Certificate
		Symmetry	Asymmetry in the log in and in the register form
	Asymmetry of the diagram at the Home page		At the Home page place the diagram at the bottom of the page in the center of the screen
	Asymmetry of the text paragraphs at the Home page		At the Home page place the text paragraphs symmetrically the one from each other

Table 7.7: Results of the analysis and the proposed improvements for the Design of the User Interface

### 7.3. Future work

Taking this Bachelor Thesis as cue point for future work, there are several approaches which could be taken in order to further improve Sorting Hat\*. The three most important among them are the ones which are going to be described in the following bullet points.

- *Implementation of the proposed improvements:* The main future approach for Sorting Hat is the implementation of the proposed improvements. If this approach is taken, the quality of the service provided by Sorting Hat will greatly increased.
- *Development of further functionalities:* Another interesting approach which could be taking is the development of new functionalities which could be integrated into the ones which Sorting Hat already have. This approach could lead to the widen of the scope of the people to which Sorting Hat is focus on.
- *Host Sorting Hat on the Internet:* A distinct approach would be hosting Sorting Hat on the Internet. This approach could be very interesting from a technical point of view because of the challenges to which the developers would have to confront.

## BIBLIOGRAPHY

- [1] I. S. A. Suliman S. Aljomaa Mohammad F. Al.Qudah and A. S. Abduljabba, “Smartphone addiction among university students in the light of some variables”, *Computers in Human Behavior*, vol. 61, no. 2, pp. 155–164, Jul. 2016.
- [2] I. T. U. (ITU), *Global ict developments, 2001-2018\**, Available at <https://www.itu.int/en/ITU-D/Statistics/Pages/stat/default.aspx>, Dec. 2018.
- [3] FBI, “Ransomware prevention and response for cisos”,
- [4] McAfee, “Economic impact of cybercrime report”, 2018.
- [5] S. Morgan, *Cybersecurity jobs report 2018-2021*, Available at <https://cybersecurityventures.com/jobs/>, May 2017.
- [6] NIST, *National initiative for cybersecurity education*, Available at <https://www.nist.gov/itl/applied-cybersecurity/nice/about>.
- [7] B. S. William Newhouse Stephanie Keith and G. Witte, *National initiative for cybersecurity education (nice) cybersecurity workforce framework*, Available at <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-181.pdf>, Aug. 2017.
- [8] *What is a website audit and why is it necessary in 2019?*, Available at <https://pedestalsearch.com/what-is-website-audit-why-necessary/>.
- [9] C. S. C. UK, *Cyber security challenge uk*, Available at <https://www.cybersecuritychallenge.org.uk>.
- [10] *Education*, Available at <https://www.cybersecuritychallenge.org.uk/education>.
- [11] *Careers*, Available at <https://www.cybersecuritychallenge.org.uk/careers>.
- [12] *Typical roles*, Available at <https://www.cybersecuritychallenge.org.uk/careers/typical-roles>.

- [13] *National initiative for cybersecurity careers and studies*, Available at <https://niccs.us-cert.gov>.
- [14] *Niccs education and training catalog*, Available at <https://niccs.us-cert.gov/training/search>.
- [15] *Workforce development*, Available at <https://niccs.us-cert.gov/workforce-development>.
- [16] *Nice cybersecurity workforce framework*, Available at <https://niccs.us-cert.gov/workforce-development/cyber-security-workforce-framework>.
- [17] E. Comission, *Cybersecurity act*, Available at [https://ec.europa.eu/commission/news/cybersecurity-act-2018-dec-11\\_en](https://ec.europa.eu/commission/news/cybersecurity-act-2018-dec-11_en).
- [18] *Enisa*, Available at <https://www.enisa.europa.eu>, 2019.
- [19] B. S. William Newhouse Stephanie Keith and G. Witte, “National initiative for cybersecurity education (nice) cybersecurity workforce framework”, in. Aug. 2017, ch. Appendix A – Listing of NICE Framework Elements, p. 11.
- [20] —, “National initiative for cybersecurity education (nice) cybersecurity workforce framework”, in. Aug. 2017, ch. Appendix A – Listing of NICE Framework Elements, p. 12.
- [21] —, “National initiative for cybersecurity education (nice) cybersecurity workforce framework”, in. Aug. 2017, ch. Appendix A – Listing of NICE Framework Elements, p. 16.
- [22] —, “National initiative for cybersecurity education (nice) cybersecurity workforce framework”, in. Aug. 2017, ch. Appendix A – Listing of NICE Framework Elements, p. 24.
- [23] —, “National initiative for cybersecurity education (nice) cybersecurity workforce framework”, in. Aug. 2017, ch. Appendix A – Listing of NICE Framework Elements, p. 77.



- [24] —, “National initiative for cybersecurity education (nice) cybersecurity work-force framework”, in. Aug. 2017, ch. Appendix A – Listing of NICE Framework Elements, p. 59.
- [25] —, “National initiative for cybersecurity education (nice) cybersecurity work-force framework”, in. Aug. 2017, ch. Appendix A – Listing of NICE Framework Elements, p. 88.
- [26] —, “National initiative for cybersecurity education (nice) cybersecurity work-force framework”, in. Aug. 2017, ch. Appendix B – Work Role Detail Listing, p. 95.
- [27] *Institute of information security professionals*, Available at <https://www.iisp.org>.
- [28] *Institute of information security professionals*, Available at [https://www.iisp.org/iisp/About\\_Us/Our\\_Frameworks/Our\\_Skills\\_Framework/iispv2/Accreditation/Our\\_Skills\\_Framework.aspx](https://www.iisp.org/iisp/About_Us/Our_Frameworks/Our_Skills_Framework/iispv2/Accreditation/Our_Skills_Framework.aspx).
- [29] L. Stanley, *10 reasons why users leave your website in 10 seconds*, Available at <https://www.resourcetechniques.co.uk/news/seo/10-reasons-why-users-leave-your-website-in-10-seconds-101189>, Sep. 2018.
- [30] *Monolithic applications*, Available at <https://docs.microsoft.com/en-us/dotnet/standard/containerized-lifecycle-architecture/design-develop-containerized-apps/monolithic-applications>, Feb. 2019.
- [31] *Common web application architectures*, Available at <https://docs.microsoft.com/en-us/dotnet/standard/modern-web-apps-azure-architecture/common-web-application-architectures#traditional-n-layer-architecture-applications>, Jan. 2019.
- [32] *Mvc architecture*, Available at <https://www.tutorialsteacher.com/mvc/mvc-architecture>.
- [33] E. Markou, *Organizing sql queries*, Available at <https://www.blendo.co/blog/organizing-sql-queries/>, Dec. 2017.
- [34] *Git*, Available at <https://git-scm.com>.

- [35] C. Rylan, *Why enforcing code style is important*, Available at <https://coryrylan.com/blog/why-enforcing-code-style-is-important>, Aug. 2015.
- [36] P. Divine, *Case styles: Camel, pascal, snake, and kebab case*, Available at <https://medium.com/@pddivine/string-case-styles-camel-pascal-snake-and-kebab-case-981407998841>, Nov. 2018.
- [37] *Relational data model in dbms: Concepts, constraints, example*, Available at <https://www.guru99.com/relational-data-model-dbms.html>.
- [38] C. Singh, *Relational model in dbms*, Available at <https://beginnersbook.com/2015/04/relational-model-in-dbms/>.
- [39] *Er diagrams vs. eer diagrams: What's the difference?*, Available at <https://cacoo.com/blog/er-diagrams-vs-eer-diagrams-whats-the-difference/>, May 2018.
- [40] D. L. Soltesz, *Relational data model in dbms: Concepts, constraints, example*, Available at <https://www.techwalla.com/articles/what-are-the-advantages-of-a-relational-database-model>.
- [41] M. Oak, *Advantages of relational databases*, Available at <https://techspirited.com/advantages-of-relational-databases>, Feb. 2018.
- [42] *Mysql workbench*, Available at <https://www.mysql.com/products/workbench/>.
- [43] *What is an entity relationship diagram*, Available at <https://www.lucidchart.com/pages/er-diagrams>.
- [44] *What is database security?*, Available at <https://www.techopedia.com/definition/29841/database-security>.
- [45] *What is data governance?*, Available at <https://dama.org/content/what-data-governance>, 2018.
- [46] E. Parliament and Council, *Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016 on the protection of natural persons with regard*

*to the processing of personal data and on the free movement of such data, and repealing directive 95/46/ec*, Available at <http://data.europa.eu/eli/reg/2016/679/oj>, Apr. 2016.

- [47] D. Tynan, *Facebook says 14m accounts had personal data stolen in recent breach*, Available at <https://www.theguardian.com/technology/2018/oct/12/facebook-data-breach-personal-information-hackers>, Oct. 2018.
- [48] D. O'Sullivan, *Facebook could face billion dollar fine for data breaches*, Available at <https://edition.cnn.com/2018/12/14/tech/facebook-billion-dollar-fine/index.html>, Dec. 2018.
- [49] D. A. Kaj Arno and M. Widenius, *General security issues and the mysql access privilege system*, Available at <https://www.oreilly.com/library/view/mysql-reference-manual/0596002653/ch04s02.html>.
- [50] M. Anderson, *How to move a mysql data directory to a new location on ubuntu 16.04*, Available at <https://www.digitalocean.com/community/tutorials/how-to-move-a-mysql-data-directory-to-a-new-location-on-ubuntu-16-04>, Jul. 2016.
- [51] M. Rouse, *What is backup?*, Available at <https://searchdatabackup.techtarget.com/definition/backup>, Oct. 2018.
- [52] S. Kumar, *Difference between library and framework*, Available at <https://www.c-sharpcorner.com/UploadFile/a85b23/framework-vs-library/>, May 2015.
- [53] S. Greif, *When evaluating any new javascript library*, Available at <https://medium.freecodecamp.org/the-12-things-you-need-to-consider-when-evaluating-any-new-javascript-library-3908c4ed3f49>, Sep. 2018.
- [54] *Companies that use jquery and jquery integrations*, Available at <https://stackshare.io/jquery>.
- [55] *Javascript where to*, Available at [https://www.w3schools.com/js/js\\_where.asp](https://www.w3schools.com/js/js_where.asp).

- [56] K. Ahmed, *How to structure your javascript*, Available at <https://kamranahmed.info/blog/2014/08/07/how-to-structure-your-javascript/>, Aug. 2014.
- [57] S. Hiemstra, *5 reasons why you need a javascript style guide*, Available at <https://www.themarketingtechnologist.co/5-reasons-why-you-need-a-javascript-style-guide/>, May 2015.
- [58] O. Editorial, *An overview of php framework guides for developers*, Available at <https://onextrapixel.com/an-overview-of-php-framework-guides-for-developers/>, May 2010.
- [59] W. Morris, *8 best php frameworks for web developers*, Available at <https://www.hostinger.com/tutorials/best-php-framework>, Feb. 2019.
- [60] *Laravel - the php framework for web artisans*, Available at <https://laravel.com>.
- [61] *Why laravel is the best php framework in 2019?*, Available at <https://www.valuecoders.com/blog/technology-and-apps/laravel-best-php-framework-2017/>, Mar. 2019.
- [62] J. Shah, *Why laravel is consider as one of the best php framework in 2018?*, Available at <https://medium.com/techcompose/why-laravel-is-consider-as-one-of-the-best-php-framework-in-2018-4f40def9c69c>, Jan. 2019.
- [63] *Laravel application directory structure*, Available at <https://www.w3schools.in/laravel-tutorial/application-directory-structure/>,
- [64] *Php | coding standards - geeksforgeeks*, Available at <https://www.geeksforgeeks.org/php-coding-standards/>.
- [65] *Php coding standard*, Available at [https://www.tutorialspoint.com/php/php\\_coding\\_standard.htm](https://www.tutorialspoint.com/php/php_coding_standard.htm).
- [66] *User interface framework*, Available at [https://docs.oracle.com/cd/E12517\\_01/back\\_office/pdf/141/html/pos\\_imp2/uiframework.htm](https://docs.oracle.com/cd/E12517_01/back_office/pdf/141/html/pos_imp2/uiframework.htm).
- [67] *Bootstrap · the most popular html, css, and js library in the world*. Available at <https://getbootstrap.com>.

- [68] *Cascade and inheritance - learn web development | mdn*, Available at [https://developer.mozilla.org/en-US/docs/Learn/CSS/Introduction\\_to\\_CSS/Cascade\\_and\\_inheritance](https://developer.mozilla.org/en-US/docs/Learn/CSS/Introduction_to_CSS/Cascade_and_inheritance), Mar. 2019.
- [69] A. D. L. Cuadra, *Methods to organize css | css-tricks*, Available at <https://webdesign.tutsplus.com/articles/css-documentation-best-practices--cms-30139>, Jan. 2018.
- [70] *Blade templates - laravel - the php framework for web artisans*, Available at <https://laravel.com/docs/5.8/blade>.
- [71] *Html5 style guide*, Available at [https://www.w3schools.com/html/html5\\_syntax.asp](https://www.w3schools.com/html/html5_syntax.asp).
- [72] *Google html/css style guide*, Available at <https://google.github.io/styleguide/htmlcssguide.html>.
- [73] *What is a responsive user interface?*, Available at <https://www.webfuel.com/what-is-a-responsive-user-interface>, Jan. 2015.
- [74] *Why responsive design is important and google approved*, Available at <https://freshsparks.com/why-responsive-design-is-important/>, Jan. 2015.
- [75] A. Dai, *6 bad ui design examples & common errors of ui designers*, Available at <https://hackernoon.com/6-bad-ui-design-examples-common-errors-of-ui-designers-e498e657b0c4>, Jan. 2018.
- [76] J. Porter, *Principles of user interface design*, Available at <http://bokardo.com/principles-of-user-interface-design/>.
- [77] *10 basic interaction design principles to boost the ux design*, Available at <https://www.mockplus.com/blog/post/interaction-design-principles>, Dec. 2017.
- [78] M. Fowler, *Refactoring*, Available at <https://refactoring.com>.
- [79] *Refactoring: Clean your code*, Available at <https://refactoring.guru/refactoring>.
- [80] I. Hellström, *Chapter 13 sql statement syntax*, Available at <https://dev.mysql.com/doc/refman/8.0/en/sql-syntax.html>.

- [81] P. Factor, *Sql naming conventions*, Available at <https://www.red-gate.com/simple-talk/blogs/sql-naming-conventions/>, Jan. 2019.
- [82] *The power of a good sql naming convention*, Available at <https://www.xaprb.com/blog/2008/10/26/the-power-of-a-good-sql-naming-convention/>, Oct. 2008.
- [83] *How to reset the root password*, Available at <https://dev.mysql.com/doc/refman/8.0/en/resetting-permissions.html>.
- [84] J. Janssen, *(more) secure local passwords in mysql 5.6 and up*, Available at <https://www.percona.com/blog/2014/11/25/more-secure-local-passwords-in-mysql-5-6-and-up/>, Nov. 2014.
- [85] J. Hughes, *How to determine your database backup schedule*, Available at <http://www.manageforce.com/blog/how-to-determine-your-database-backup-schedule>, Feb. 2016.
- [86] *Automysqlbackup*, Available at <https://sourceforge.net/projects/automysqlbackup/>, Nov. 2016.
- [87] *How to backup and or restore your mysql database using phpmyadmin*, Available at <https://support.managed.com/kb/a2034/how-to-backup-and-or-restore-your-mysql-database-using-phpmyadmin.aspx>.
- [88] Airbnb, *Javascript style guide*, Available at <https://github.com/airbnb/javascript#naming-conventions>, Apr. 2019.
- [89] *Vue.js*, Available at <https://vuejs.org>, 2019.
- [90] *Javascript & css scaffolding - laravel - the php framework for web artisans*, Available at <https://laravel.com/docs/5.8/frontend>.
- [91] J. Shah, *Why does php suck? | why does it suck?*, Available at <https://whydoesitsuck.com/why-does-php-suck/>, Jul. 2014.
- [92] ———, *Php: A fractal of bad design / fuzzy notepad*, Available at <https://eev.ee/blog/2012/04/09/php-a-fractal-of-bad-design/>, Apr. 2012.
- [93] *Psr-2: Coding style guide - php-fig*, Available at <https://www.php-fig.org/psr/psr-2/>.

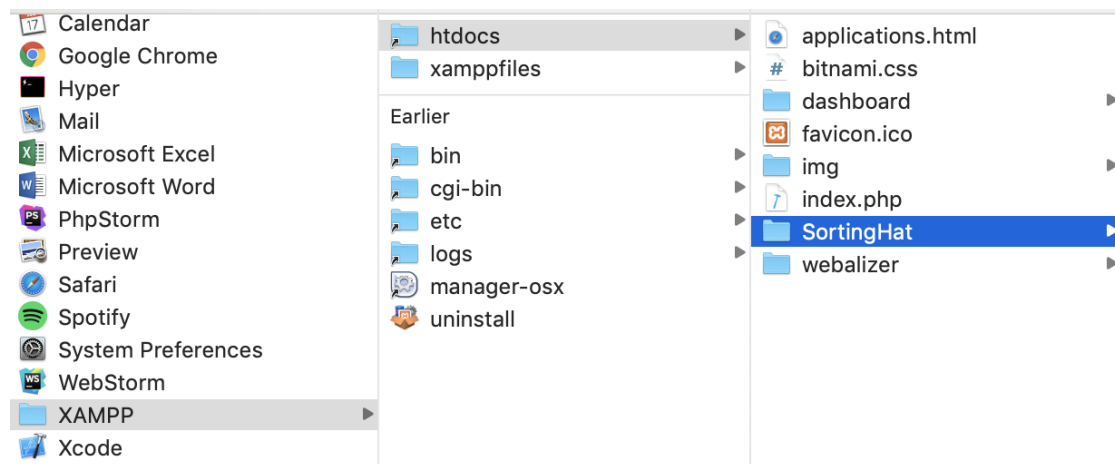
- [94] I. Brown, *Methods to organize css* | *css-tricks*, Available at <https://css-tricks.com/methods-organize-css/>, Jul. 2017.
- [95] *Nis directive* — *enisa*, Available at <https://www.enisa.europa.eu/topics/nis-directive>.
- [96] J. del Estado, *Ley orgánica 3/2018, de 5 de diciembre, de protección de datos personales y garantía de los derechos digitales*, Available at <https://www.boe.es/buscar/doc.php?id=BOE-A-2018-16673>, Dec. 2018.

# APPENDIX A: USER MANUAL FOR DEPLOYING SORTING HAT

## HAT

In this appendix the steps to the correctly deployment of Sorting Hat are going to be described. In addition to this, the open source and free web server stack XAMPP is the chosen solution for this project. Because of this, its use is highly recommended. In the case that any other web server is desired to be used instead of XAMPP, it is considerably recommendable to read its documentation in order to deploy the application without any inconvenience.

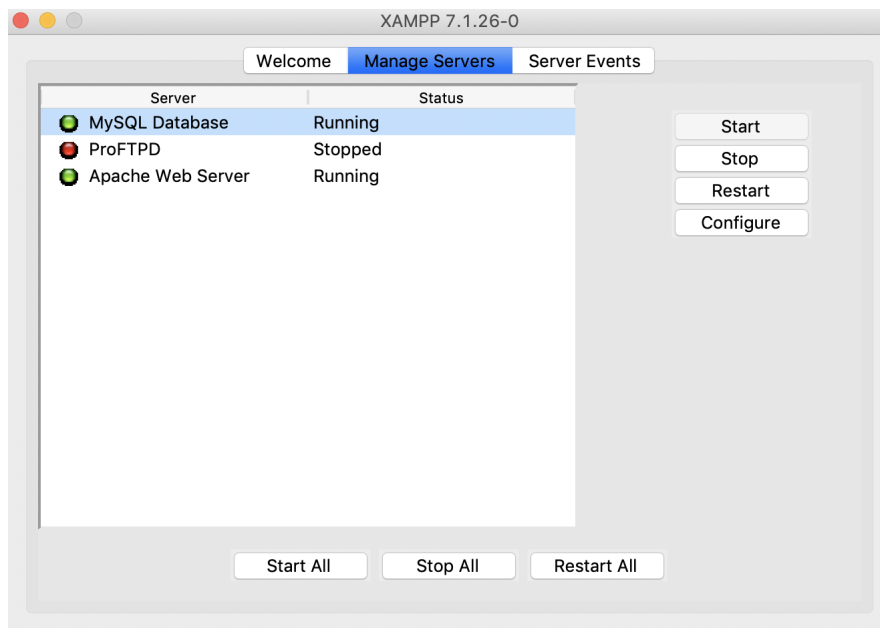
Firstly, XAMPP has to be download. It is important to check that the downloaded version is compatible with the OS and version used. This user manual is focus in *mac-OS*. Once it is downloaded and installed, the compress file with the source code of Sorting Hat should be decompressed and copied to the folder *htdocs*. This folder is located inside XAMPP application folder.





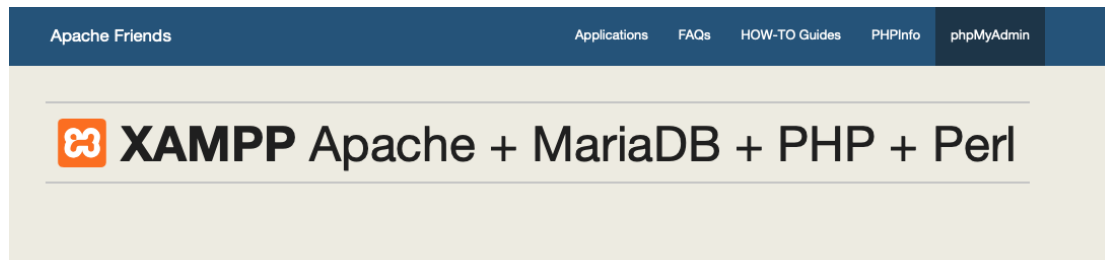
## Deployment of MySQL Database and Apache server

The next step to perform is starting the server and setting up the database. In order to accomplish it, XAMPP should be opened. Once it is open, the options *Apache Web Server* and *MySQL Database* have to be run by clicking the "Start" option.



## Migrating the data to the database

When the database is correctly deployed, the following step to be performed is migrating the data from the SQL script to the database. In order to perform this phase the "Go to Application" option need to be clicked and then the *phpMyAdmin* section too.



### Welcome to XAMPP for OS X 7.1.26

You have successfully installed XAMPP on this system! Now you can start using Apache, MariaDB, PHP and other components. You can find more info in the [FAQs](#) section or check the [HOW-TO Guides](#) for getting started with PHP applications.

XAMPP is meant only for development purposes. It has certain configuration settings that make it easy to develop locally but that are insecure if you want to have your installation accessible to others. If you want have your XAMPP accessible from the internet, make sure you understand the implications and you checked the [FAQs](#) to learn how to protect your site. Alternatively you can use [WAMP](#), [MAMP](#) or [LAMP](#) which are similar packages which are more suitable for production.

Start the XAMPP Control Panel to check the server status.

### Community

XAMPP has been around for more than 10 years – there is a huge community behind it. You can get involved by joining our [Forums](#), adding yourself to the [Mailing List](#), and liking us on [Facebook](#), following our exploits on [Twitter](#), or adding us to your [Google+](#) circles.

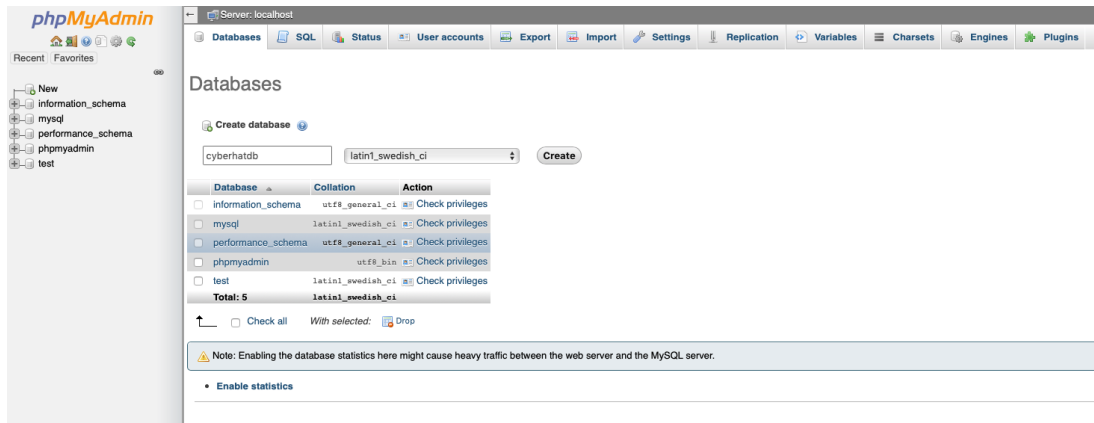
### Contribute to XAMPP translation at [translate.apachefriends.org](http://translate.apachefriends.org).

Can you help translate XAMPP for other community members? We need your help to translate XAMPP into different languages. We have set up a site, [translate.apachefriends.org](http://translate.apachefriends.org), where users can contribute translations.

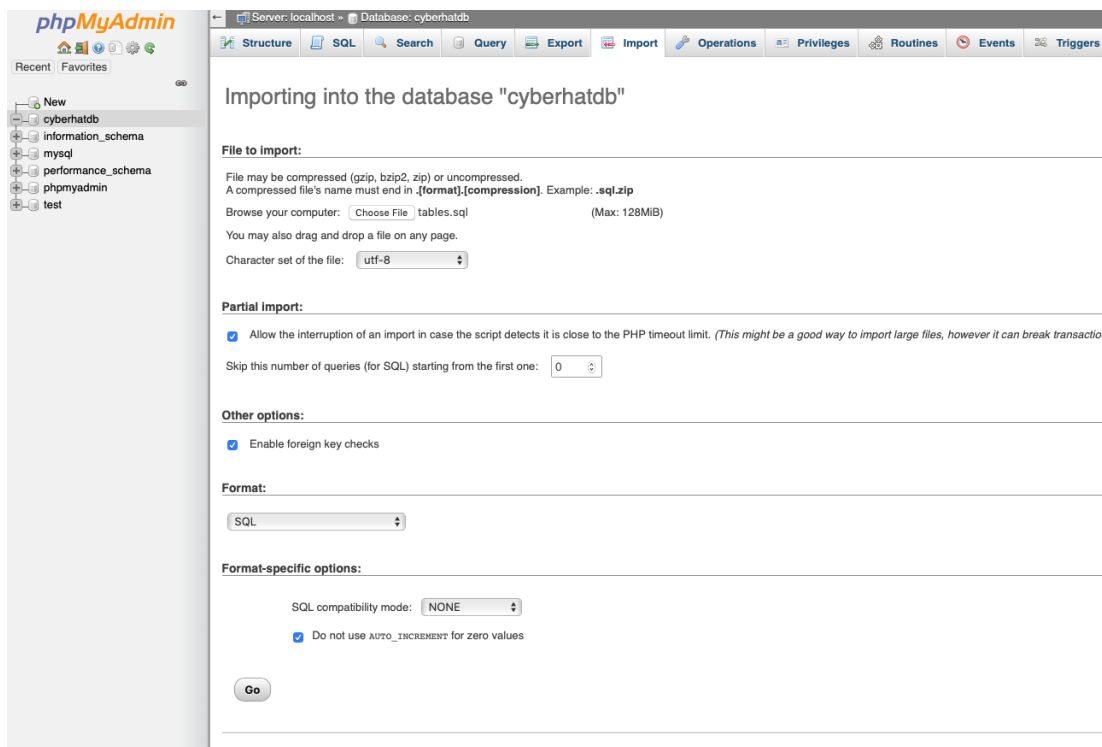
### Install applications on XAMPP using Bitnami

Apache Friends and Bitnami are cooperating to make dozens of open source applications available on XAMPP, for free. Bitnami-packaged applications include Wordpress, Drupal, Joomla! and dozens of others and can be deployed with one-click installers. Visit the [Bitnami XAMPP page](#) for details on the currently available apps.

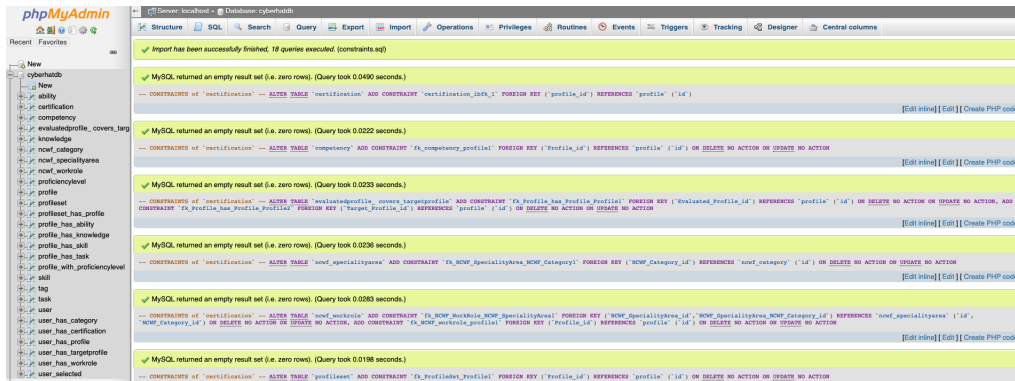
Once in *phpMyAdmin*, a new database is to be created. The name of the new database has to be "cyberhatdb" for the correct execution of Sorting Hat.



Furthermore, when "cyberhatdb" is created, the next step is importing the data from the SQL file into it. In order to perform this step, the section "Import" and "Choose File" should be clicked. The file chosen is the SQL files located in the folder "Base de datos". These files is *cyberhatdb.sql*.



Finally, a similar screen to the following picture should be displayed notifying the correct import of the SQL files.



## Changing file's permissions

In order to Sorting Hat to run correctly, the files regarding to Sorting Hat need to have permissions for executing, reading and writing. If this is not performed, Sorting Hat will go through an error regarding to this situation when Sorting Hat is accessed.

Consequently, the following command need to be executed in order to grant the needed permissions to the files.

$$\textit{sudo chmod -R 0777 htdocs/sortinghat/} \quad (7.1)$$

Thanks to this command, all the files inside the folder called "*sortinghat*", which contains all the files of Sorting Hat, will have the needed permissions for the correct execution of Sorting Hat.

## Browsing in Sorting Hat

The last step in this user manual is just opening Sorting Hat in the browser. In order to accomplish it, the URL *http://localhost/sortinghat/public/* should be ac-

cessed. Once this URL is browsed, the user is redirected to the main page of Sorting Hat. A similar web page to the following page showed should be displayed in the browser.

