

SECURITY DATA GOVERNANCE SYSTEM

SZABOLCS MAGYAR

The dissertation was submitted in part fulfilment of requirements for the degree of MSc Information
Management

DEPT. OF COMPUTER AND INFORMATION SCIENCES

UNIVERSITY OF STRATHCLYDE

AUGUST 2019

DECLARATION

This dissertation is submitted in part fulfilment of the requirements for the degree of MSc of the University of Strathclyde.

I declare that this dissertation embodies the results of my own work and that it has been composed by myself. Following normal academic conventions, I have made due acknowledgement to the work of others.

I declare that I have sought, and received, ethics approval via the Departmental Ethics Committee as appropriate to my research.

I give permission to the University of Strathclyde, Department of Computer and Information Sciences, to provide copies of the dissertation, at cost, to those who may in the future request a copy of the dissertation for private study or research.

I give permission to the University of Strathclyde, Department of Computer and Information Sciences, to place a copy of the dissertation in a publicly available archive.

(please tick) Yes [☒] No [☐]

I declare that the word count for this dissertation (excluding title page, declaration, abstract, acknowledgements, table of contents, list of illustrations, references and appendices) is . 21660

I confirm that I wish this to be assessed as a Type 1 ☒ 3 ☐ 4 ☐ 5 Dissertation (please circle)

Signature:



Date:

14/08/2019

ABSTRACT

As the Internet has become increasingly ubiquitous, the number of IT incidents and data breaches increased, consequently leading to substantial revenue losses for companies. These losses could be limited by Information Security Management Systems (ISMS). This research examines best practices from the literature and software solutions regarding the asset, risk and policy management, and the way these best practices can be implemented.

The aim of this study is to design an ISMS that is based on best practices from the literature and other ISMS software solution. The study also examines how the proposed system can be built, presents the core functionalities that the system needs to support, proposes the best methodology that can be used for the implementation, and explores the languages and tools that could be used to make the system secure and ensure its usability.

It was discovered that asset, risk, and policy management are connected to each other and that current software solutions are either too complex to use or do not provide a good graphical interface. The system was developed using Agile. 3 tier architecture has been used where the back-end is a MySQL database. Due to the proposed architecture of the system, SQL injections are the highest threats, however, the system is secure against them. The system is simple, informative and has a dashboard page where diagrams, that show the Key Performance Indicators (KPIs) of an organisation, can be seen.

Contents

1. Introduction.....	1
1.1. Motivation	1
1.2. Research problem, questions and objectives.....	2
1.3. Outcome	3
1.4. Dissertation outline	3
2. Background and literature review	4
2.1. Background.....	4
2.1.1. Asset management.....	4
2.1.2. Risk management	4
2.1.3. Policy management	5
2.1.4. The connection between asset, risk and policy management	6
2.2. ISO 27001 and 27002	6
2.3. Information Security Management Software (ISMS) solutions	10
2.3.1. Risk & Audit	10
2.3.2. StandardFusion.....	12
2.3.3. eramba	13
2.4. Selecting the right product.....	15
3. Requirements specification	17
3.1. Methodology	17
3.1.1. Requirements elicitation	17
3.1.2. MoSCoW	18
3.1.3. Use case diagrams	18
3.2. Requirements gathering.....	19
3.3. Use case diagram.....	21
4. Implementation	23
4.1. Methodology	23
4.1.1. Agile	23
4.1.2. Entity Relationship Diagram	25
4.2. High-level design	25
4.2.1. The architecture of the system	25
4.2.2. User interface design.....	25
4.2.3. Database design	28
4.3. Low-level design	32
4.3.1. Programming languages and tools used	32
4.3.2. Structure of the code	34

4.3.3. Security of the code.....	34
4.3.4. Login page.....	35
4.3.5. My account functionalities	36
4.3.6. Admin pages	37
4.3.7. Functionalities available for every user.....	39
5. Testing	44
5.1. Testing types.....	44
5.1.1. Unit testing	44
5.1.2. Use case testing.....	44
5.1.3. Load testing	45
5.1.4. User interface testing	45
5.1.5. Vulnerability testing	45
5.2. Findings.....	46
5.2.1. Nessus scan	47
6. Results and evaluation	48
6.1. Results	48
6.2. Usability testing.....	52
6.3. Requirement-based evaluations	54
6.4. Security evaluations	55
6.5. Reflective evaluation of development process	56
7. Conclusion	57
7.1. Summary of thesis contributions	57
7.2. Further development perspectives	59
7.2.1. Crucial improvements	59
7.2.2. Cosmetic improvements.....	59
7.2.3. Improvements for future projects.....	59
References.....	60
The Appendix.....	65
Use cases	65
Usability test.....	84

Table of Figures

Table 1. Functional requirements	19
Table 2. Non-functional requirements	21
Table 3. Users	29
Table 4. Assets	30
Table 5. Policies	31
Table 6. Risks	31
Table 7. Event log	32
Figure 1. ISO 27001 framework (https://advisera.com/27001academy/what-is-iso-27001/)	7
Figure 2. Number of ISO 27001 certificates worldwide (https://www.iso27001security.com/html/27001.html)	9
Figure 3. ISMS functionalities (www.capterra.com)	10
Figure 4. Resolver's graphical interface (www.resolver.com)	11
Figure 5. Resolver control effectiveness (www.resolver.com)	11
Figure 6. StandardFusion compliance	12
Figure 7. StandardFusion risk impact	13
Figure 8. Eramba Demo	14
Figure 9. Eramba dashboard	15
Figure 10. Use case diagram	22
Figure 11. Asset page design	27
Figure 12. Create/edit new assets	28
Figure 13. Entity-relationship diagram	29
Figure 14. Example of populating a drop-down list	33
Figure 15. Object-oriented mysqli example	35
Figure 16. Password encryption	36
Figure 17. Admin pages in the Navigation bar	37
Figure 18. Editing users	38
Figure 19. Implementing DataTables	38
Figure 20. Next review date	39
Figure 21. Asset database constraint	40
Figure 22. Download script for policies	41
Figure 23. Google Charts with Ajax	42
Figure 24. Nessus scan result	47
Figure 25. Login page	48
Figure 26. Dashboard	49
Figure 27. Asset management	49
Figure 28. New asset creation	50
Figure 29. Risk Management	50
Figure 30. Policy Management	50
Figure 31. User management	51
Figure 32. Event log	51

1. Introduction

1.1. Motivation

The increase in computers and their utilization brought the era of Information Society (Ohki, et al., 2009). With the help of computers and Internet companies could reach heights they have never imagined before. On the other hand, misuse of computers increases as well. These incidents can cause a lot of harm to a company since businesses and the companies' competitiveness builds upon it. Once an IT accident happens to a company it could reduce the company's value. The main cause is that when people use the companies' websites they create usernames with a password and in many cases these websites require them to give confidential information (date of birth, place of birth, bank card number, etc.). These people entrust the companies to manage their data carefully and if a security incident happens trust will be lost which results in revenue loss as well, especially if the incident affects many users. This will lead to more attention from the media and many people will know about it. As the Internet became increasingly widespread, the amplitude of these incidents and data breaches became bigger and bigger. The biggest data breaches in history are linked to Yahoo (Armerding, 2018). In 2013 one billion accounts have been compromised. Names, email addresses, passwords, date of births, security questions and answers have been stolen. Later, in 2014 500 million user data has been leaked, including real names, email addresses, telephone numbers and dates of birth. In 2017 Yahoo estimated that all three billion user accounts had been compromised. These breaches decreased the company's sales price by \$350 million and have been sold for \$4.48 billion

Although these huge data breaches have happened mostly in the 2010s, there were security incidents before as well. That is the reason IT security and IT security data governance have been born. According to the Information Security Handbook (Bowen, et al., 2006, p. 2), information security governance is:

"... the process of establishing and maintaining a framework and supporting management structure and processes to provide assurance that information security strategies are aligned with and support business objectives, are consistent with applicable laws and regulations through adherence to policies and internal controls, and provide assignment of responsibility, all in an effort to manage risk."

The international standard for information security is ISO 27001, it describes the requirements for implementing an information security management systems (ISMS) and it is supported by ISO 27002 which is a code of practice and recommends best practices (Calder, 2018). ISMS is defined as "part of

the overall management system, based on a business risk approach, to establish, implement, operate, monitor, review, maintain and improve information security.” (Calder & Watkins, 2008, p. 40).

1.2. Research problem, questions and objectives

Because of the increasing importance of IT security and its management are growing there is a need for ISMS. Most of this software is available for commercial use so the companies have to buy them if they want to use them, however, there are some open-source software as well for example eramba (eramba, 2019a). Not only the main goal and the target customer size of this software may differ but the features too. When one compares some of them it is easy to see that the main difference amongst the ISMSs is the functionality. Functionalities can be for example whether the software solution is able to do auditing, incident management, environmental compliance, IT and operational risk management, policy management, reporting and prepare activity dashboards. Moreover, it is important to keep in mind that risk management, policies, and controls can mean different things for different people even within the same profession. There is no standard about what the minimum requirements of these software’s functionalities are and what they should necessarily contain. On the other hand, some software contains too many functionalities, they became extremely complicated to work with and thus they lose one of their most important goal which is to help the work of the management in governance, risk management, and compliance. This goal helps the management to make valid and well-established decisions fast and so preserving the companies’ competitiveness and avoiding to pay huge fines and lose valuable customers. Although ISO 27001 is the standard there is no best practice approach for asset, risk and policy management.

The main questions of this research are:

- What are the best practices for asset, risk and policy management?
- How a simple system should be built that manages information security governance data?
- What are the functionalities this system should support?
- What is the best methodology that should be used when developing this system?
- What are the best languages, libraries that could be used for developing the system?
- How system security can be ensured?
- How system usability can be ensured?

The main objective of this research is to design and implement an Information Security Management System which will answer the questions above. This ISMS aims to provide the basics of asset, risk and policy management according to the best practices from the literature.

1.3. Outcome

An ISMS system has been developed. It is a 3 tier architecture where the back-end is a MySQL database. Users can register and log in to the system where they can view entities they own like assets, their associated risks, and corresponding policies. They can see a dashboard on a separate page where three diagrams are shown based on their own entities. Users can also change their password, delete their account and log out. The administrators of the system can view, edit and delete not only their own entities but the entities of everybody. They are able to see and change the ownership of other entities as well. In the dashboard page, they see diagrams based on every entity. Moreover, they have access to two more pages as well: they can send emails to users and edit user rights on the user management page and in the event log page they can see the basic actions that have taken place in the website including actions connected to entities and users. The software uses only sessions (and not cookies) to identify users. The level of security is high, it is fully protected against SQL injections and cross-site scripting (XSS) as well.

1.4. Dissertation outline

The rest of the chapters describe steps that were taken in the process of developing the software. Chapter 2 contains the description of asset, risk and policy management, it is shown how these connect to each other and to ISO27001 and 27002. Moreover, different ISMS solutions are presented and the best practices of choosing the best product are explained. In chapter 3 the requirements gathering process is shown with the description of the methodology, how the requirements have been collected, what these are, how these are classified and a use case diagram shows how the system's functionalities work. In the first part of chapter 4, the high-level design is demonstrated with the methodology used, the architecture of the system, database design and user interface design. In the second part of chapter 4 programming languages used are demonstrated through coding examples. The solutions are described and decisions made are explained. In testing, techniques are described and the findings of testing are outlined. In chapter 7 the methodology and findings of evaluation are presented. Lastly, in the Conclusion, the research is summarized and further development perspectives are specified.

2. Background and literature review

In this chapter firstly the need for IT governance is described, focusing on asset, risk and policy management. After that IT security standards are defined: ISO 27001 and ISO 27002. It is shown why these standards are important, what kind of content they have and how organisations can comply with these standards. In the second part of the chapter different Information Security Management Software solutions are demonstrated: Risk & Audit, StandardFusion, and eramba. All of them are products with an easily accessible demo version and all of them have a proper number of reviews as well. Lastly, best practices are collected about what kind of features a software should support based on the literature.

2.1. Background

2.1.1. Asset management

According to Biswas (Biswas, 2019), an asset is “Any item of economic value owned by an individual or corporation.” The main concept of asset management is that the most important assets should be identified, tracked, classified and the owner should be assigned to them to ensure they are sufficiently protected. If a company would like to be effective, the asset management strategy should include information assets (assets that contain information), software assets and information technology equipment as well. In the IT asset management process these assets are tracked, verified that they are up-to-date (reviewed) and that they are protected. Every asset has an owner and a responsible person, a classification which shows how important the asset is for the company, a create date and protection as well. Asset management is the coordinated and systemized practices and activities through an organization manage its assets optimally with their associated performance, risks and expenditures. There are a huge amount of reasons asset management is important (Dosal, 2018): improperly managed assets can become cybersecurity vulnerabilities and negatively impact workflows and not knowing these assets may result in not knowing the single points of failure. Moreover, IT asset management is crucial for regulatory compliance (e.g. GDPR), the incoming assets need to be properly tracked and the outgoing assets properly disposed of. The most important benefit is that through asset management the risks a company may face can be mitigated.

2.1.2. Risk management

IT risk is any threat to business data, critical systems, and business processes. These risk can be associated with the use, operation, ownership, involvement, influence, and adoption of IT. (NI Business Info, 2019a). These risks have the potential to damage business value. There are several types of risks: physical threats (e.g. theft or unauthorised access), electronic threats (hacker gets accessed to a

system, computer virus or fraudulent email), technical failures (software bugs or computer failures), infrastructure failures (loss of internet connection) or human error (someone might delete a file) (NI Business Info, 2019b). Risk management is to make a decision and take actions to address uncertain outcomes, to control how risks might impact the business. According to Mark Darby (Darby, 2019), “Information security risk management (ISRM) is the process of identifying, evaluating and treating risks around the organisation’s valuable information”.

There are five steps of ISRM (Rapid7, 2019). The first one is identification where assets, vulnerabilities, threats, and controls are identified. After that, the assessment happens, where the information gathered in the first step is combined. The most basic formula to calculate risk is $\text{likelihood} \times \text{impact}$, however, this can be $\text{threat} \times \text{vulnerability} \times \text{likelihood} \times \text{impact} \times \text{asset value} - \text{security controls}$ as well. After the assessment, the treatment is the next step where the organization should select treatment options (remediation, mitigation, transference, risk acceptance, and risk avoidance). The decision must be communicated within the organization as a fourth step, it is crucial that the responsibility and accountability need to be defined and that the stakeholders understand the cost of treating risk. As the last step, it is a continuous process, the controls need to be monitored.

There are several benefits of risk management. It will help deal with the uncertainties around assets (Darby, 2019). Implementing a risk management system can help the formulation of realistic plans in terms of time and cost estimate (Rahman Ahlan & Arshad, 2012). If the risk has been understood then it can lead to the minimization of risks and by building up statistical data of historical risks may help facilitate greater but more rational risk-taking. Risk management also helps the company avoid additional costs and disruptions to and identify the risks that are worth pursuing.

2.1.3. Policy management

Policies are critical to the organization since they establish boundaries for individuals, relationships, processes and transactions (Kothari, 2019). If policies are properly managed, communicated and enforced, they provide a framework for governance, they identify and treat risks and they define compliance. Compliance is documented by policies meaning how the organization meets requirements and obligations from contracts, regulators and other commitments. Without policies, there are no written standards for acceptable and unacceptable conducts and since policies attach legal duties of care to the organization, mismanagement of policies may introduce liability and exposure, and noncompliant policies may be used against the organization in legal and regulatory proceedings.

If an organization has many departments, each of them may have its own policies which may result in wasted resources through redundancy and overlap, poor visibility and reporting and lack of accountability. Policy management is “the process of creating, communicating, and maintaining

policies within an organization” (VComply Editorial, 2018). There are five steps in policy management. In the first one, in Creation, a policy is written and goes through an approval process. After that in Communication, the policy is communicated to the staff (after approval) which includes publication and training. These policies are then enforced and the exceptions are managed (Management) and the policies are reviewed regularly, updated and archived when needed. This is a continuous process as well. There are best practices to policy management (McLachlan, 2011): ownership should be created, the policies should be centralised and communicated early on, policies, processes and procedures should be refined as needed and they should be enforced as well. There are numerous advantages of policy management: employees understand the constraint of their job, policies enable the workforce to clearly understand individual and team responsibilities which saves time and resources, it allows managers to exercise control by exception and it provides legal protection as well.

2.1.4. The connection between asset, risk and policy management

The process that connects asset, risk and policy management is the following: since only those assets are listed in the system who have risk, the risk is assigned to these assets. The control (mitigate) of these risk is ensured through policies because these policies contain actions i.e. what needs to be done in order to prevent risk. These policies are assigned to risks (and indirect to assets as well) to make sure the risks are mitigated and it is on the lowest possible level.

2.2. ISO 27001 and 27002

Standards represent a consensus on characteristics that should remain applicable for a longer period of time so these should be documented and published (Disterer, 2013). These standards should support companies and individuals when procuring services and products. International Organization for Standardization (ISO) is the leading issuing body for international standards. ISO 27001 was adopted from BS 7799 part two (British Standards Institute) in 2005 and the current version is ISO 27001:2013 (ISO 27001 Security, 2019a).

The main goal of ISO 27001 is to protect the confidentiality, integrity, and availability of the information in an organization (Advisera, 2019). The method for that is first, potential problems that could happen to the company should be found out (risk assessment) and after that preventive measures should be defined (risk mitigation). ISO 27001 is based on managing risk: the first one needs to find out where the risks are and after that these risks can be treated systematically.

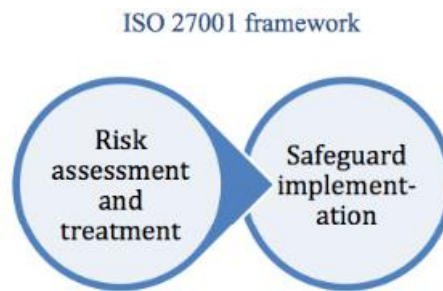


Figure 1. ISO 27001 framework (<https://advisera.com/27001academy/what-is-iso-27001/>)

The controls usually are policies, technical implementations or procedures. Since a company certainly possesses these controls, ISO 27001 has described a way to fit all these elements into a system, an Information Security Management System (ISMS) that can be seen in Figure 1.

Since ISO 27001 is a framework, it does not define how an ISMS solution should work and what kind of functionalities it should consist of. That is the scope which can be found in the Statement of Applicability (SoA) part of the standard. It depends on the person who does the implementation what the appropriate protection is for his/her organisation, for the specific needs e.g. one company needs a backup every 24 hours while the others do not, there are not two companies that are the same. ISO 27001 requires the companies to perform risk assessments and risk treatments and the company needs to implement control mechanisms to have the protection from all the risks that are described in the risk assessment part (Kosutic, 2019). These controls are described in the Annex part of ISO 27001 (described later). Moreover, ISO 27001 gives a checklist of what top managers must do (besides the safeguards). This checklist contains the following elements: setting the business expectations for information security, publishing a control how these expectations are met, designating main responsibilities for information security, providing enough human resources and money and reviewing regularly whether all the expectations are met.

In Annex A of ISO27001 reference control objectives and controls are described. These controls are taken over from ISO 27002. It is a normative list meaning that the companies are expected to use it but they can decide which controls they would like to use and they can even supply this list with their own controls in order to be able to address their particular risks. This is where ISO 27001 and 27002 connect (ISO 27001 Security, 2019b): ISO 27001 defines the mandatory requirements for an ISMS and it uses ISO 27002 to indicate information security controls within the ISMS. ISO 27002 is a code of practice (and not a formal specification). Organisations that adopt this standard must assess their own information risks, define their controls objectives and apply controls using the standard. 35 control objectives are specified. These control objectives are at a high level and contain functional requirements specification for a company's information security management architecture. There are 114 controls and each of these control objectives is supported by at least one control. The control

objectives and controls are divided into 14 categories: information security policies, organization of information security, human resource security, asset management, access control, cryptography, physical and environmental security, operations security, communications security; system acquisition, development and maintenance, supplier relationships, information security incident management, information security aspects of business continuity management, compliance.

Traditionally ISO 27001 has used the Plan-Do-Check-Act (PDCA) cycle for implementing an ISMS (Calder, 2013). It is not mandatory to use the PDCA cycle for that purpose after the introduction of the 2013 version of the standard, however, it is still worth considering. The PDCA cycle states that business processes need to be treated as they are in a continuous feedback loop because then managers can identify and change those parts of the process that needs improvement. In this case, when implementing ISMS, the steps of the cycle mean the following actions: in the 'Plan' stage, the ISMS is established. Information assets and their associated security requirements are identified, information security risks are assessed and relevant controls are selected to manage unacceptable risks (Disterer, 2013). In the 'Do' stage the ISMS is implemented and operated by implementing controls and managing operations. The 'Check' stage is about monitoring and reviewing the ISMS. Here the tasks are monitoring and assessing performance. Lastly, in the 'Act' step the ISMS is maintained and improved by corrective and preventive actions. By making sure this ISMS implementation cycle is continuous it is certain that the system and the organization's information security management is maintained and improving.

If the organization wants to verify that its ISMS complies with ISO 27001 it needs to pass a certification procedure conducted by a certification organization, Registered Certification Bodies (RCB) (Disterer, 2013). The ISO has a list of RCBs and the organization needs to choose an RCB. The certification process might take a few months and it has a validity of three years. After that, a re-classification can be applied which takes less time than the first certification process. By 2017, there have been 40.000 certifications worldwide and it increases by 20% annually that can be seen in Figure 2 (ISO 27001 Security, 2019a).

There are many business benefits a company can achieve when implementing ISO 27001. First of all, it complies with legal requirements (Advisera, 2019). There are more and more regulations and laws regarding IT security, by implementing ISO 27001 the organization can resolve this problem and comply with all the laws. Since the main philosophy of ISO 27001 is to prevent security incidents from happening (which would cost money regardless of how huge they are), by preventing them the organization saves more money than the actual investment in ISO 27001.

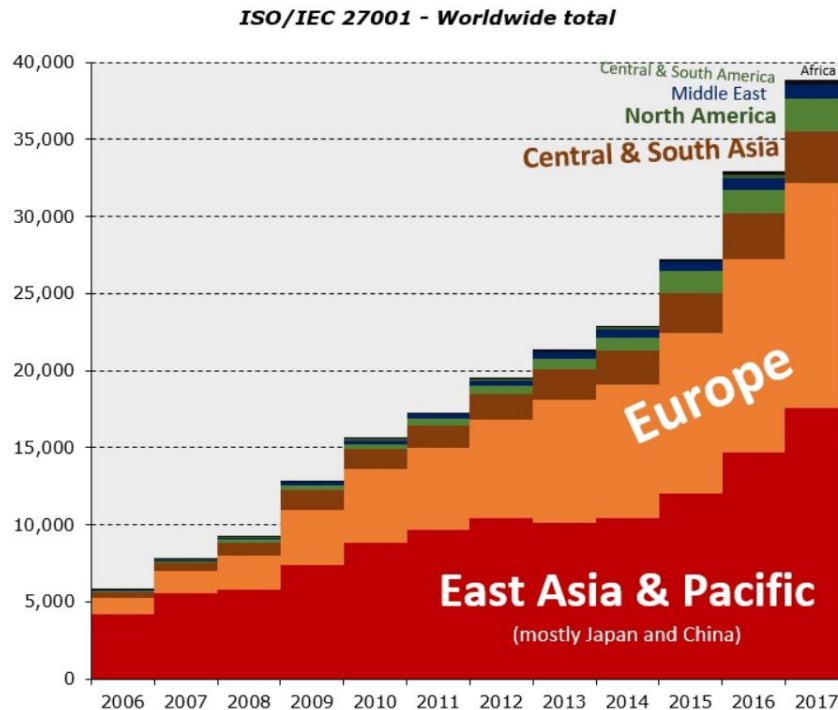


Figure 2. Number of ISO 27001 certificates worldwide (<https://www.iso27001security.com/html/27001.html>)

ISO 27001 encourages companies to write down their processes. The reason that it is a huge benefit is it reduces the cost of lost time of employees since if these processes are not defined, the employees probably do not know what, how and by whom needs to be done. Moreover, it manages information in all its form, helps the company self-defend from threats and technology-based risks, it adapts to changes in the environment and in the organisation (Dutton, 2017). Since it makes sure that information security is present and established in the business, its presence improves organisational culture and makes processes efficient. It ensures business continuity through protecting data availability, confidentiality, and integrity (CIA) and critical business processes from a major disaster. Due to ISMS, companies are more resilient to cyber-attacks and because of monitoring, improvement, internal audits and corrective actions make sure that the controls work properly and they are up to date.

On the other hand, the value of the certificate is dependent on the scope of the ISMS and on the SoA e.g. if the scope is only a department and/or the company accepts that there are malware risks but it does not implement controls the certification body would not refuse to certify the company since antivirus controls are not mandatory (ISO 27001 Security, 2019a). That is why being certified does not guarantee that the organization is secure, but that it has a compliant ISMS. Furthermore, a study of 25 firms of U.S. and Europe (Hsu, et al., 2016) found out that having an ISO 27001 certificate does not result in advantages in terms of financial and stock market performance. The authors found two main reasons for that. The first is that the main goal of ISO 27001 is to prevent loss through mitigating risks

and having the certification is seen that the organization meets the requirements and not as a competitive advantage. The other reason is that many firms hold the certificate that covers only part of the organization although the best is to manage risk at the organizational level.

2.3. Information Security Management Software (ISMS) solutions

Many companies do not/cannot track risk assets and policies and controls together: it may track only the risk assets and the policies valid for these etc. There is a huge amount of software solutions in the market that has been created in order to be able to assist the high demand from various companies all around the world. In order to be able to compare this software and their functionality, many websites have been created. According to [capterra.com](https://www.capterra.com), these are the functionalities an ISMS can possess (see Figure 3):

✓	Alerts/Notifications
✓	Auditing
✓	Business Process Control
✓	Compliance Management
✓	Corrective Actions (CAPA)
✓	Exceptions Management
✓	Internal Controls Management
✗	IT Risk Management
✓	Legal Risk Management
✗	Mobile Access
✓	Operational Risk Management
✗	Predictive Analytics
✓	Reputational Risk Management
✗	Response Management
✓	Risk Assessment

Figure 3. ISMS functionalities (www.capterra.com)

It is easy to understand that a small company needs a different solution than a multi-national company, not only because of the functionalities used, but there are other factors, for example, price, which platforms the software supports and what kind of training opportunities are, too. It is important to state that most of this software, even the demo version as well is set-up to work with business entities (B2B). That is why only software with free demo version is shown.

2.3.1. Risk & Audit

It identifies itself as “A solution for the entire organisation.” (Resolver, 2019). It is an audit software mainly meant for internal audits to automate processes and to improve the ability to deliver value to the organisation. Risk & Audit has many pros and a small number of cons. First of all, the dashboards

are great, the graphical user interface is not only beautiful but useful too. There are tables that clearly show the trends and thus making decisions is easier and faster (see Figure 4):

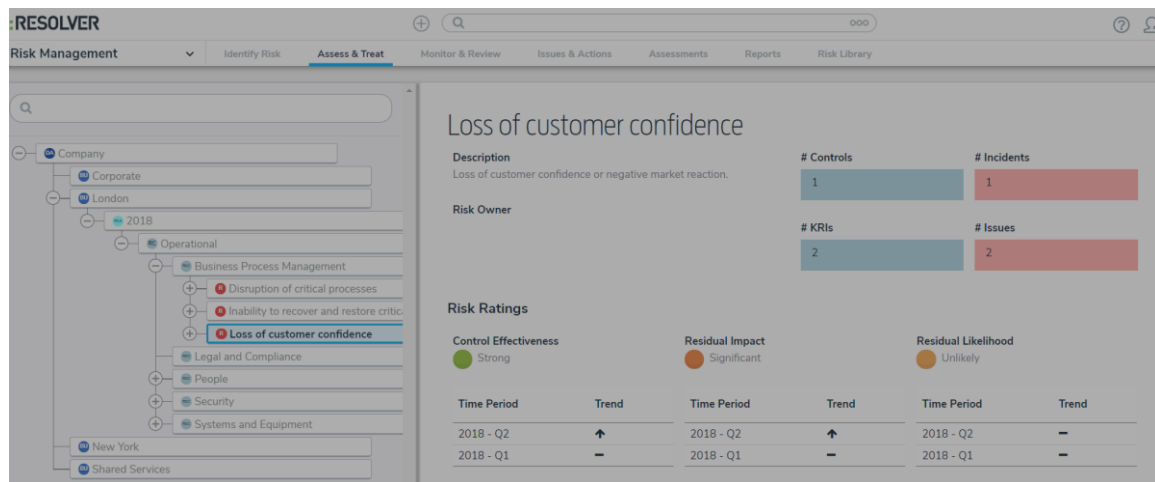


Figure 4. Resolver's graphical interface (www.resolver.com)

The reason is that once management is involved in interpreting the outcomes from a software, especially in the case of an audit and KPIs, it is extremely important to have a clean and meaningful interface, and that is what Resolver's software does (see Figure 5):

Step 2: Determine Control Effectiveness

Review and document any related controls and assess the overall control effectiveness

Control Effectiveness			Guidance	
Controls				
Name	Created On	Key Control	Operating Effectiveness	# Design Effective
Formal customer feedback process	2018-06-13		Not Tested	1
Control Effectiveness		Proposed Changes		
<div><div></div>Strong</div>				

Figure 5. Resolver control effectiveness (www.resolver.com)

This is a really nice example of a clean interface. Not only because the colours help perceive and understand how good/bad (in this case) a control perform, but if one would not know what a colour mean, he could go to the Guidance (next to Control Effectiveness) and the explanation would be there. Another advantage is that the product can be customized easily. According to the reviews on Capterra (Capterra, 2019), it is the software that works for the customer and for its need and not vice versa. Moreover, because the software is simple and user-friendly, it is easy to use and training does not take long. Lastly, the most important advantage is that Resolver handles everything that is needed for auditing. On the other hand, there are some features that could make this product even better. One is that there is no auto-save option which can be troublesome in case of power or internet shortage so

hours of work could be lost easily. Lack of reminders for task management can be disturbing as well, especially when there are requirements that should not be overlooked. Moreover, not having a policy management module means that understanding how risks are handled could be problematic, however, since there is no public data about the pricing it is difficult to know the value for money.

2.3.2. StandardFusion

StandardFusion is another software solution for IT GRC. Its slogan is “Say goodbye to spreadsheets” (StandardFusion, 2019a). It is an integrated Risk Management GRC software. As its motto says the main goal is to replace spreadsheets and MS Excel-based standard, control and risk management into a system that can handle all of them at the same time, without making it complicated. It has four main parts: Risk, Audit, Compliance and Vendor Assessment. The controls and their linkage with standards and risks are in the centre. On one hand, according to the reviews (Capterra, 2019), the software is really simple, informative and helps understand compliance as well (see in Figure 6):

	Name	Title	Workflow State	Status	Owner
✓	04. CONTEXT OF THE ORGANIZATION				
^	4.1	Understanding the organization and its context The organization shall determine external and internal issues that are relevant to its purpose and that affect its ability to achieve the intended out...	NEW	100%	jordan@dropbag.net
^	4.2 (a)	Understanding the needs and expectations of interested parties The organization shall determine interested parties that are relevant to the information security management system;	NEW	100%	(None)
^	4.2 (b)	Understanding the needs and expectations of interested parties The organization shall determine the requirements of these interested parties relevant to information security.	NEW	100%	(None)
^	4.3	Determining the scope of the information security management system The organization shall determine the boundaries and applicability of the information security management system to establish its scope.	NEW	50%	(None)
^	4.3 (a)	Determining the scope of the information security management system When determining this scope, the organization shall consider the external and internal issues referred to in 4.1;	NEW	0%	(None)
^	4.3 (b)	Determining the scope of the information security management system When determining this scope, the organization shall consider the requirements referred to in 4.2;	NEW	0%	(None)
^	4.3 (c)	Determining the scope of the information security management system When determining this scope, the organization shall consider interfaces and dependencies between activities performed by the organization, an...	NEW	0%	(None)
^	4.3	Determining the scope of the information security management system The scope shall be available as documented information.	NEW	0%	(None)
^	4.4	Information security management system The organization shall establish, implement, maintain and continually improve an information security management system, in accordance with t...	NEW	0%	(None)
✓	05. LEADERSHIP				
^	5.1 (a)	Leadership and commitment Top management shall demonstrate leadership and commitment with respect to the information security management system by: (a) defining the L...	NEW	0%	(None)
^	5.1 (b)	Leadership and commitment Top management shall demonstrate leadership and commitment with respect to the information security management system by: ensuring the L...	NEW	0%	(None)
^	5.1 (c)	Leadership and commitment Top management shall demonstrate leadership and commitment with respect to the information security management system by: ensuring that...	NEW	0%	(None)
^	5.1 (d)	Leadership and commitment Top management shall demonstrate leadership and commitment with respect to the information security management system by: communication...	NEW	0%	(None)
^	5.1 (e)	Leadership and commitment Top management shall demonstrate leadership and commitment with respect to the information security management system by: ensuring that...	NEW	0%	(None)
^	5.1 (f)	Leadership and commitment Top management shall demonstrate leadership and commitment with respect to the information security management system by: directing and...	NEW	0%	(None)
^	5.1 (g)	Leadership and commitment Top management shall demonstrate leadership and commitment with respect to the information security management system by: promoting co...	NEW	0%	(None)
^	5.1 (h)	Leadership and commitment Top management shall demonstrate leadership and commitment with respect to the information security management system by: supporting ot...	NEW	0%	(None)
^	5.2 (a)	Policy Top management shall establish an information security policy that is appropriate to the purpose of the organization;	NEW	0%	(None)
^	5.2 (b)	Policy Top management shall establish an information security policy that includes information security objectives (see 6.2) or promotes the framework f...	NEW	0%	(None)
^	5.2 (c)	Policy Top management shall establish an information security policy that includes a commitment to satisfy applicable requirements related to informat...	NEW	0%	(None)

Figure 6. StandardFusion compliance

In Figure 6 different actions and completion can be seen. By using a simple diagram for the status one can decide fast which actions need to be focused on and if there is a question, who is the responsible person people can turn to. To make it more practical, at the owner column, not the actual name but the email address is shown so people can directly write an email to that person. There is no unnecessary information on this page that is why it is so valuable.

At the risks, the risk impact can be seen which then recalculates the risk level. To make it more understandable, the software uses a graphical interface to highlight it better (see Figure 7). Although this table format may contain too much information, the emphasis is on risks and how the controls mitigate these risks. This is the reason the graphical interface is useful in this case. For every control, a task can be created which can be assigned to people all around the organisation. That is why this software's main targeted user group is managers who can track anything in the system. To sum up, StandardFusion's advantage is based on its power to link connections, its interface is simple and informative and it is a framework tool i.e. any IT security standard can be implemented through this product.

Name	Treatment	Probability	Impact	Vulnerability	Value	Risk Level	Probability	Impact	Vulnerability	Value	Risk Level
Production Datacenter - Failure of communication link	Mitigate Risk	Very High	Very High	High	Low	52.84	Immensely	Medium	Low	Low	35.29
Production Datacenter - Internet outage	Mitigate Risk	Low	High	Low	Immensely	41.18	Very High	Medium	Medium	Medium	41.18
Production Datacenter - Unauthorized physical access	Mitigate Risk	High	Medium	Very High	Low	41.18	Medium	Low	Medium	Medium	23.53
Production Datacenter - Air conditioning failure	Mitigate Risk	Low	Very High	Low	High	35.29	Medium	Medium	Low	Medium	23.53
Production Datacenter - Power outage	Mitigate Risk	Low	Immensely	Low	Immensely	52.84	Medium	Low	Low	Low	11.76
Production Servers - Network	Mitigate Risk	Immensely	High	Medium	Immensely	70.59	Low	Medium	Low	Low	11.76
Production Servers - Corruption of data	Mitigate Risk	Immensely	High	Medium	Immensely	70.59	Low	Medium	High	Medium	23.53
Production Servers - Degradation in system performance	Mitigate Risk	Medium	High	Medium	Very High	47.06	Very High	Low	Medium	Medium	35.29
Production Servers - Inadequate system capacity	Mitigate Risk	Medium	High	High	Very High	52.84	Medium	Medium	Medium	Low	23.53
Production Servers - Malfunction of equipment	Accept Risk	High	Immensely	High	Immensely	76.47	Medium	High	High	Medium	41.18
Production Servers - Power outage	Mitigate Risk	Low	Very High	Low	Immensely	47.06	Medium	High	High	Medium	41.18
Production Network - Failure of communication links	Mitigate Risk	Low	Immensely	Low	Immensely	52.84	Low	Low	Medium	Low	11.76
Production Network - Inadequate system capacity	Mitigate Risk	Medium	High	Medium	Very High	47.06	High	Low	Low	Low	17.85
Production Network - Internet outage	Mitigate Risk	Low	Immensely	Low	Immensely	52.84	Low	Medium	Low	Medium	17.85
Production Network - Malfunction of equipment	Mitigate Risk	Low	High	High	Immensely	52.84	Medium	Medium	Low	Low	17.85
Production Network - Connection of unauthorized devices	Accept Risk	Low	High	Low	Medium	23.53	Medium	Medium	Low	Medium	23.53
Customer Data - Backups unavailable	Mitigate Risk	Medium	Very High	High	Immensely	64.71	Medium	Medium	Low	Very High	35.29
Customer Data - Corruption of data	Mitigate Risk	Medium	Immensely	Medium	Immensely	64.71	Medium	Medium	Low	Low	17.85
Customer Data - Lack of audit trail	Accept Risk	Low	High	Low	Medium	23.53	Very High	Low	Low	Low	23.53
Network and System Administrators - Death or injury	Mitigate Risk	Low	Very High	Low	Very High	41.18	Low	Medium	Low	Low	17.85
Network and System Administrators - Human error	Mitigate Risk	Medium	High	Low	Very High	41.18	Low	Medium	Medium	Low	17.85
Network and System Administrators - Social engineering	Mitigate Risk	Medium	High	Medium	High	41.18	Low	Medium	Low	Medium	17.85
Software Developers / Architects - Death or injury	Accept Risk	Low	High	Low	High	28.41	Medium	Medium	Medium	Medium	28.41
Software Developers / Architects - Malicious employee	Mitigate Risk	Low	Very High	Medium	Immensely	52.84	Medium	Low	Low	Low	11.76
Software Developers / Architects - Social engineering	Mitigate Risk	Medium	Medium	Medium	High	35.29	Low	Low	Medium	Low	11.76
Dropshipping Source Code - Disclosure of confidential information	Mitigate Risk	Low	Immensely	Medium	Immensely	58.82	Medium	Medium	Immensely	47.06	47.06
Dropshipping Source Code - SQL Injection	Mitigate Risk	Medium	Very High	Medium	Immensely	52.84	Low	Medium	Medium	Medium	17.85
Dropshipping Source Code - Backdoor through code	Mitigate Risk	Medium	Very High	High	Immensely	64.71	Medium	Medium	High	High	35.29
Dropshipping Source Code - Inadequate version history	Accept Risk	Low	Medium	Low	Medium	17.85	Low	Low	Medium	Medium	17.85
Dropshipping Source Code - Lack of audit trail	Accept Risk	Low	Medium	Low	Medium	17.85	Low	Medium	Low	Medium	17.85
Backup Media - Backup data corruption	Mitigate Risk	Medium	Very High	High	Very High	58.82	Medium	Medium	Low	Medium	23.53
Backup Media - Backups unavailable	Mitigate Risk	Medium	Very High	High	Very High	58.82	Immensely	Very High	Low	Low	47.06
Backup Media - Deterioration of storage media	Mitigate Risk	Medium	High	High	High	47.06	Medium	Medium	Low	Low	17.85
Customer Contracts - Corruption of data	Accept Risk	Low	Medium	Low	Medium	17.85	Medium	Low	Medium	Low	17.85
Customer Contracts - Deterioration of paper documents	Accept Risk	Medium	Low	Low	Low	11.76	Low	Medium	Low	Low	11.76

Figure 7. StandardFusion risk impact

On the other hand, the software lacks two functions (Capterra, 2019), the first one is more important than the second: there is no policy management that would help to mitigate the risks and enforce the standard while the introduction of approval stages would help following the task and audit flow more clearly and more transparent. Regarding the pricing StandardFusion charges from 500 USD/month up to 3000 USD/month plus on-boarding fees (StandardFusion, 2019b).

2.3.3. eramba

Eramba is an open IT GRC software. It defines itself as “eramba is the leading, open enterprise-class IT Governance, Risk & Compliance application” (eramba, 2019a). It is public information that the software has been downloaded 4867 times in 2018. Open means that anyone can download and use the community version which is meant to be used on Linux. It has an enterprise version as well, which comes in a package and has more support than the community version (eramba, 2019b): enterprise release, install assistance, online Q&A, support and software updates. These extra services cost 2500

EUR/3000 USD a year. There are additional services as well: starters' assistance (80 EUR/hour) and onsite workshops (starting from 1800 EUR). Although StandardFusion offers more, this area is where Eramba has an advantage above all other ISMS: it is free, you do not have to pay to use the software. Moreover, it is accessible through Virtual Machines, so Linux is not a requirement. It has public documentation (both video and documents) on the website that helps people make understood how the software works. It has nine core functionalities: policy management, controls, and audit, exception management, compliance management, risk management, data flow analysis – GDPR, incident and project management. Figure 8 shows how the demo looks like (Eramba, 2019c):

DASHBOARD

PROGRAM

ORGANIZATION

ASSET MGT

CONTROLS CATALOGUE

RISK MGT

COMPLIANCE MGT

SECURITY OPERATIONS

SYSTEM

Figure 8. Eramba Demo

The graphical interface looks simple and plain, especially the colours and the information it shows. However, in this case, this level of information is preferable since every one of them that are shown is needed for the daily work without having to enter other menus or submodules. Eramba's documentation is thorough and proves that eramba meets the requirements for a good ISMS: every module is linked with each other, this linkage is explained in the documentation so it is understandable, there are subsections for audits, and reports can be prepared not only for sections but for items too. Next review date always needs to be input and if this date has passed without the review actually happening, that risk/policy/control has a yellow text that the planned review date has passed. Moreover, notifications can be created, the calculations of risks make sense, and so overall it has every functionality an ISMS should possess. On the other hand, some of the advantages are disadvantages at the same time: because the software is complex, training takes a huge amount of time: to familiarize with the functions, to understand the relationships between them and although the videos and documentation are detailed, there might be some questions that would make the usage clearer,

however, the user would need to pay money in order to get the answers. Furthermore, in spite of reports being prepared are informative, the main dashboard is not (Eramba, 2019c):

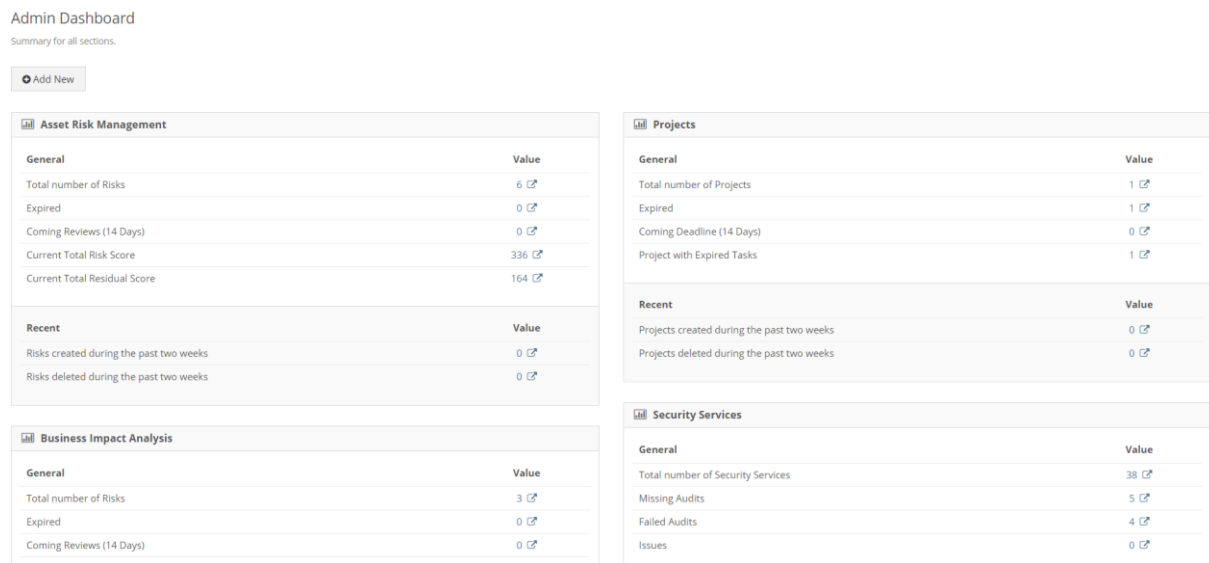


Figure 9. Eramba dashboard

In Figure 9, there are basically just numbers without indications, colours and comparisons. Even for a person who works in the industry for 10+ years, it would be hard to solve the meaning of these number and what trends they show. Only by showing a trend i.e. there was an increase/decrease in the number of e.g. risks in the last month would be more useful, however, dashboards are better if they are shown on diagrams (possibly on a weekly/monthly basis) so that it would help the interpretation of these plain numbers and their consequences and fast and easy decision-making process. If people use eramba for reporting to up to senior manager level, the software is a really good choice. However, above that, where people need to decide fast and they need to receive high-level information i.e. rather organisational level reports than item-specific ones, the software lacks visualization and creativity. All in all, eramba is a good choice if one wants a free and thorough IT GRC software, however, he needs to keep in mind that the training takes a huge amount of time and some of the dashboards are not as informative as they could be.

2.4. Selecting the right product

In the last sections, three Information Security Management Software is shown and there is an almost infinite number other ones on the market a company could choose from. Although the strengths and weaknesses of these three products are described, before starting to design the product, it is worth considering what features an ISMS should possess. First of all, good ISMS supports standard and regulatory compliance (Klassen, 2018). Since most of the companies implement ISMS to comply with the industry-specific guidelines and standards, it is important that the software should support

compliance with regulations and rules that apply to that specific industry where the organisation is present.

Another feature that the software should support is audits (Klassen, 2018). Internal audits need to be carried if the organisation wants to comply with ISO 27001:2013 and external audits need to be supported in case of HIPAA/PCI etc. Moreover, by using a software that supports audits not only time and money are saved but it establishes confidence in compliance as well.

In the case of security incidents, the software should have an intuitive dashboarding system so as executive managers have more control and oversight of security incidents (Klassen, 2018). It is important that this system should use real-time data. For security analysts, an ISMS should offer a tool that shows what security incidents happened in the past and are happening currently so that these can be analysed.

One of the most important features an ISMS should offer is risk assessment (Klassen, 2018). Without understanding risks, an organisation cannot budget its workload and information security resource. It is also crucial that risk should be calculated (quantified) and classified. After that, the company can have a proper understanding of the highest-vulnerability vectors of a data breach. By implementing risk assessment, controls and policies these risks can be mitigated. Moreover, it is really important that these risks should be connected to assets the company possesses, however, only those assets should be listed who have information security risks.

Lastly, the software should be able to support GDPR. This part affects only those organisations that collect personal information from residents of the EU (Klassen, 2018). Organisations only have 30 days to answer data requests and because of the fact that anyone has 'the right to be forgotten' and the right to access any information about them, if a high volume of request would be received, the system should be able to handle these requests as well. Moreover, if a company is not compliant with GDPR but it should be then it can receive a fine of up to four % of annual global turnover or €20 million, whichever is greater. That means that the companies are interested in complying with GDPR not only because of the ISMS but financially too.

3. Requirements specification

In the previous chapters the relevance of the thesis topic, research questions, ISMS software solutions from the literature were described with the types of software solutions present on the market including their pros and cons. In this chapter firstly the methodologies used are defined: requirements elicitation, MoSCoW and use case diagrams. After that details of these methodologies' actions are described, some of them only in the Appendix.

3.1. Methodology

3.1.1. Requirements elicitation

According to Capiro (Capiro, 2019), there are eight steps of the requirements elicitation. At first, an approach to requirements is established. This includes setting unique identifiers and traceability measures for requirements artefacts and deciding on how version control, change, and configuration are managed. The second step is to establish the system context. Business problems, objectives and opportunities need to be determined, the actuality of the project is described, the research questions are asked. As a next step, stakeholder analysis and management need to happen. This includes identifying, analysing and managing stakeholders, furthermore, stakeholders' goals and perspectives need to be understood as well. The fourth step is to determine a strategy for requirements specification. Specification style may vary between formal and informal, sparse and comprehensive, mainly spoken and mainly written. The choice of style depends on a number of factors e.g. project stage, risk (the higher the risk, the more formal the approach should be), non-functional requirements, project approach (waterfall or agile), preferred requirements style (use cases or agile user stories or only describe user stories), use of models, prototypes, simulations, and link with testing (early involvement of test team). After that requirements can be elicited, analysed and modelled. Based on the strategy defined in step four, relevant approaches to requirements elicitation can be selected so that these requirements can be analysed for relevance, correctness, conflict, ambiguity, feasibility, etc. Based on the analysis, requirements can be modelled (for example with prototypes). As a sixth step, these requirements need to be validated by stakeholders which ensures that all of them agree that these requirements lead to the solution of the problem. After that, it needs to be ensured that requirements procedures are integrated with testing so that testers assist in the process of requirements analysis. The last step is to assess the achievement of benefits (when these benefits will be realized).

3.1.2. MoSCoW

Without ranking the requirements, the company may focus on those requirements the customer does not consider important so in these cases the customer's needs might be misunderstood (Haughey, 2019).

Thanks to this prioritization technique the developing order of the requirements can be easily set, moreover, it can be easily decided what not to develop in case of pressure on resources. MoSCoW stands for must-have, should-have, could-have and would-have. Must have requirements are non-negotiable ones and without delivering them the project would not be a success. An example is the security functionalities that help maintain compliance. Since these are mandatory requirements, the easiest way if somebody is unsure about a specific requirement's importance is to determine what happens without that feature. If the product is not working without it or the release is useless then it is a must-have requirement. The project team should deliver as many should-have requirements as it can. The reason is that although they are not vital to the final product, it will still function, they add significant value to the product if they are included (ProductPlan, 2019). Examples for should have requirements are performance improvements, minor bug fixes or even new functionality. Could-have requirements are nice to have although the delivery of these does not affect the success of the whole project since they have a smaller impact on the final product. There are the first requirements that are deprioritized i.e. left out if the first two categories turn out bigger than expected. Would-have requirements (or "Wish" and "Would not have this time") helps manage expectation about features that might be included in the release but this time they are not the priority. This category helps prevent scope creep which happens when a project grows uncontrollably. Some of these features are prioritized later while others are not likely to happen at all. The technique not only captures a broader perspective by including participants from various departments but it allows the team to determine the energy amount that goes to each category too. Thanks to this, a good variety of initiatives can be delivered.

3.1.3. Use case diagrams

Use case diagrams are the part of the UML (Unified Modelling Language) and describe a sequence of interactions between external factors (system, person or a device) and the system and they are used to analyse high-level requirements (Ceta, 2019). The requirements are expressed through use cases. There are three main components of a use-case diagram: functional requirements, actors who interact with the system and relationships between use cases and actors, represented by straight arrows. It is essential that these diagrams represent a high-level overview of the relations between use cases, systems and actors (Lucidchart, 2019b). These diagrams are optimal in the following scenarios: basic flow of events are modelled in a use case, when the context and requirements of the system are

needed to be specified, when the main objective is to represent the goal of a system-user interaction and when functional requirements are needed to defined and organized in a system.

3.2. Requirements gathering

The first step before gathering the requirements is to define the stakeholders. This project idea came from a person who works at the University. This person was both the project sponsor and the key system user as well. The potential users of the system are a small group of technical people who were represented by the project sponsor, all of them are stakeholders. Requirements gathering took place through a semi-formal interview with the project sponsor. The precondition of this interview was that knowledge of literature about ISMS systems, best practices and specific software solutions needed to be present so that the sponsor's needs could be understood and if needed, advised as well. In this interview, the sponsor described the functionalities the system should possess. It was agreed that the system would be developed using Agile, use case diagrams would display the different functionalities and that before the start of the implementation, user interface design would be shown to the project sponsor to receive opinion and comments about it. The design would be implemented through low-level prototyping. After the interview, the requirements catalogue was set up. This catalogue uses the MoSCoW method, described earlier in this chapter, to prioritise these requirements of the stakeholders and it contains not only functional but non-functional requirements as well. The prioritization of functionalities was an input that came from the key system user. The functional requirements specify something the system should do, it can be behaviour or function, for example, displaying a name or adding an item (Eriksson, 2012). Two user types should be present in the system: admin, who can view/edit/delete every entity (assets, risks, policies and users) and the user who can view/edit/delete only those entities where he is the owner. The functional requirements can be seen in Table 1:

Table 1. Functional requirements

<u>Name of the requirement</u>	<u>Description (if needed)</u>	<u>Priority (based on MoSCoW)</u>
1. Register		Must-Have
2. Login		Must-Have
3. Password reset		Must-Have
4. Sign out		Must-Have
5. Delete account		Must-Have
6. Dashboard page		Must-Have
6.1. Asset classification diagram	A diagram that shows the number of assets based on their classification (high, medium, low)	Must-Have

6.2. Chart indicating the number of risks that are compliant and non-compliant	A risk is compliant if the residual risk is equal or lower than the target risk and non-compliant if it is higher than the target risk	Must-Have
6.3. Chart indicating how many assets are compliant and non-compliant	An asset is compliant if there is a risk assigned to it. If there is no risk assigned to it then the asset is non-compliant	Must-Have
6.4. Custom charts	Users can build their own charts (data shown and chart type would be selected)	Would-Have
7. Asset management page		Must-Have
7.1. Viewing assets		Must-Have
7.2. Creating assets		Must-Have
7.3. Editing assets		Must-Have
7.4. Deleting assets		Must-Have
8. Risk management page		Must-Have
8.1. Viewing risks		Must-Have
8.2. Creating risks		Must-Have
8.3. Editing risks		Must-Have
8.4. Deleting risks		Must-Have
9. Policy management page		Must-Have
9.1. Viewing policies		Must-Have
9.1.1. Downloading the file	A file that contains the policy could be downloaded from the system	Could-Have
9.2. Creating policies		Must-Have
9.2.1. Attaching a file	A file that contains the policy could be uploaded to the system	Could-Have
9.3. Editing policies		Must-Have
9.4. Deleting policies		Must-Have
10. User management page		Should-Have
10.1. Viewing users	Only admins can view the users present in the system	Should-Have
10.2. Editing user types	Only admins can edit the user types of specific users	Should-Have
10.3. Deleting users	Only admins can delete users from the system	Should-Have
11. Event log page	Only admins can see the event log of the system	Could-Have

The non-functional requirements describe how the system should behave. Typical non-functional requirements are performance, scalability, security, etc. These can be seen in Table 2:

Table 2. Non-functional requirements

<u>Name of the requirement</u>	<u>Priority (based on MoSCoW)</u>
6. Testing the implementation for common security vulnerabilities	Must-Have
7. Using security features (e.g. HTTPS)	Must-Have

The goal of these non-functional requirements is to make the system secure and prevent a data breach, especially is that the application is an audit system meaning that the attackers may gather information about the system vulnerabilities.

3.3. Use case diagram

Based on the requirements a use case diagram is created that can be seen in Figure 10. This use case diagram contains every functional requirement that was requested but on a high level. There are three types of users: non-registered person, user, and admin, the admin is a generalization of the user. That means that he is capable of doing the user can do, and also he has access to more functions as well. Every relation the login use case has is 'use' (include), meaning without logging in, these functionalities are not available. This is the same when creating a risk since assets and policies need to be created so that risks can be created.

All the other relationships are 'return' (extend). Those pages where other functionalities can be found use this relation because they extend the number of functionalities a user is able to do on that specific page.

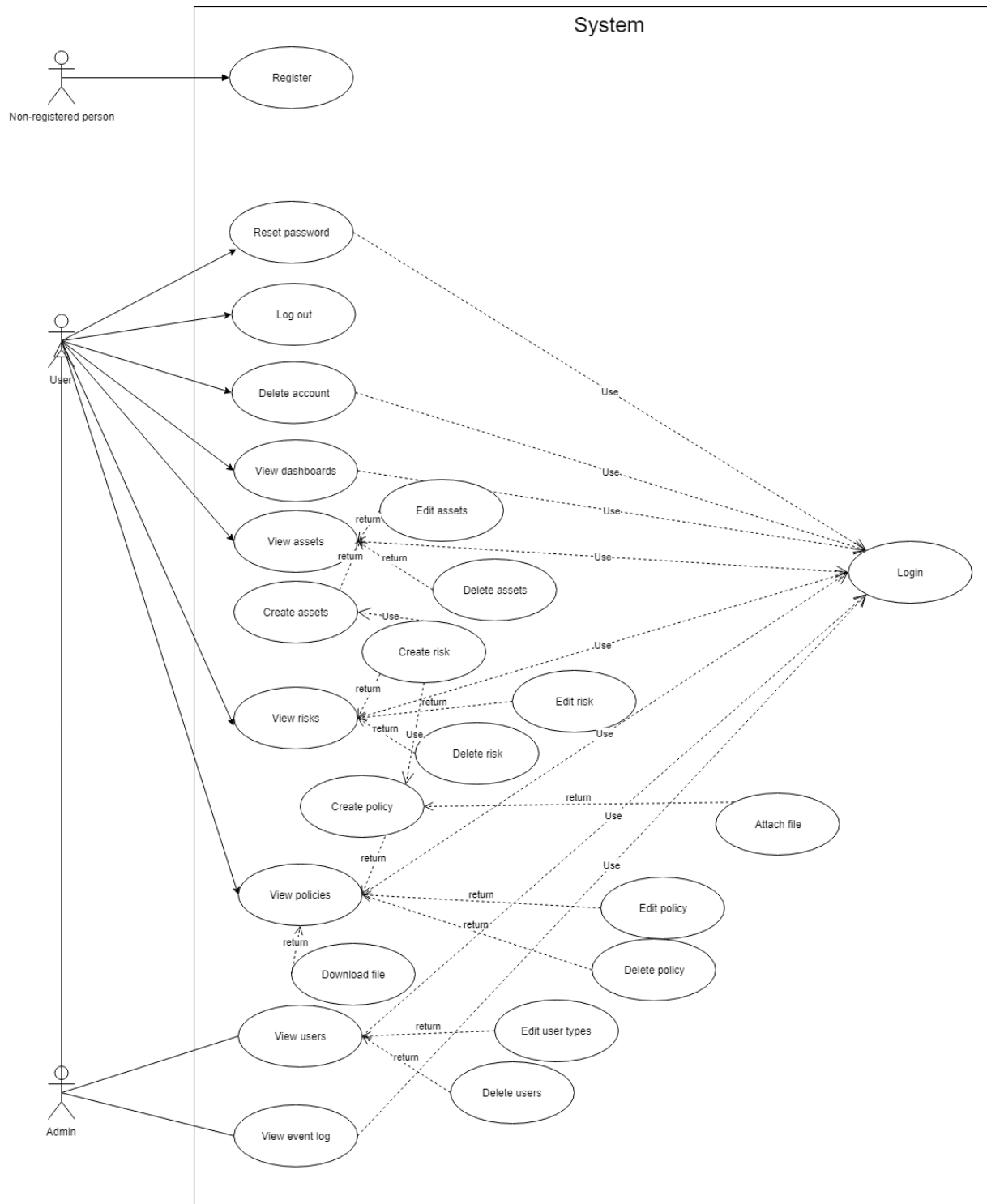


Figure 10. Use case diagram

4. Implementation

So far the chapters were mostly theoretical: the literature has been reviewed, the software solutions currently present in the market have been inspected with identifying their pros and cons, the requirements have been listed based on the stakeholders' needs and a use case diagram has been created which shows how the final product should work in the end. The next step in this process is the actual implementation. This chapter has three main parts: in the methodology Agile and entity relationship diagram is described. In the second part, in high-level design, the architecture of the system is described, the user interface design and database design is shown. In the last part, in low-level design, the specific languages and tools used are defined and the structure of the code is described, where the main focus is on the options chosen during the implementation with reasoning why that specific option has been selected.

4.1. Methodology

4.1.1. Agile

According to a survey, at least 97% of organizations practiced agile in 2018 (QASymphony, 2019). Agile started in 2001 when 'Agile manifesto' was published. It contains four important values: the focus should be on individuals and interactions (instead of tools processes), working software is more important than documentation, customer collaboration is more essential than contract negotiation and the process should respond to change (rather than follow a plan) (Gonçalves, 2019). These values and the 12 principles guide how to create and respond to change and how uncertainty should be dealt with. In agile, every product is developed in sprints. It is a period of time which is allocated for a specific phase of a project (Team Linchpin, 2019). These sprints are completed when the time period expires, which is usually one week or two weeks. After a sprint, a new one starts and the focus is on a new phase of the project.

There are a lot of advantages of using Agile (Gonçalves, 2019): since the client is actively involved in the project through sprint meetings, there is a continuous level of collaboration between the parties. In these meetings, the team can understand the client's needs and thus deliver a high-quality product. This would promote further engagement as well. Another advantage is transparency, meaning that the client is actively involved throughout the whole project. Because of sprints, predictability is high and new features can be delivered frequently and companies are able to plan ahead. Moreover, costs are predictable since they are limited and based on work done in each sprint. Because of this, the decision-making is improved and it allows prioritization which is another advantage. It is important to clarify that during each of these sprints, shippable units of work are delivered. These are different features

of the final product. Agile allows for change: since the focus should be on delivering the agreed subset of the product, there is a place for reprioritization and refinement of the product backlog which is everything that is needed in the product. These changes might be added to the next iteration. By working together closely with the client, their needs can be understood more easily and this can give the most value to the business. By focusing on the user needs, each feature delivers value for the user and better opportunities of receiving feedback are provided through early beta testing so changes could be implemented as soon as possible. Lastly, since the product is broken down into deliverable of units, in every sprint there is enough time to develop and test that specific unit meaning that defects and mismatches can be found and fixed early.

There are some disadvantages of agile (Team Linchpin, 2019): it is rather developer-centric than user-centric, instead of focusing product design it focuses on processes for getting requirements and developing the code, and Agile methodologies may be inefficient in large organizations.

There are some practices that are used when using Agile. Here those will be listed that have been used during the implementation of the product. The key principle of Agile is to have a running software meaning that software is always compiled, built, tested and deployed. With automated testing, other features of the software are protected while making changes to a part of it, moreover, this is a faster and more efficient way to find bugs. Another crucial practice is iteration and task planning. During the development of the software two iterations have been set, each of them lasted two weeks. In the first iteration, the 'Login', 'Assets' and 'User management' parts of the application have been developed while 'Dashboards', 'Risks', 'Policies' and 'Event log' were built up in the last two weeks. The reason for this division is that at first a working product should have been developed that can be used by the stakeholders and only after that the remaining functionalities could be implemented. In every iteration, task planning is important. In this implementation, a risk-based approach has been followed: the tasks have been tackled first were the one that it was known that either they took more time or there was a knowledge gap so there was uncertainty about them. It was a crucial decision to use this approach since if the knowledge of how to prepare a task was present, it meant that the planning of the implementation of a task is easier and more efficient.

Because of these factors, the implementation has been successful. This is one of the reasons Agile was used and not the waterfall methodology. Another reason is that by using Agile the requirements documentation and request for changes are more continuous and it results in a better final product in contrast to waterfall where the developers and the clients agree on a documented requirements list and the client may realize only at the end of the implementation that the product that has been developed is not acceptable, however, the cost of change it is high.

4.1.2. Entity Relationship Diagram

The Entity-Relationship (ER) Diagram is a flowchart that illustrates how entities relate to each other within a system (Lucidchart, 2019a). They show the relationship of the entities stored in a system (smartdraw, 2019). An entity is a component of data and these entities have attributes that define their properties. The ER diagram shows the logical structure of the databases. They were developed in 1976 by Peter Chen. They are used for many purposes and in the next part. ER diagrams are used to troubleshoot existing databases to find and resolve issues. These diagrams are used to design or analyse relational databases used in business processes since these processes use entities, actions, and interplay which can benefit from a relational database.

4.2. High-level design

4.2.1. The architecture of the system

The product uses the classical architecture 3 tier (3T) architecture of a web application. The three are presentation, logic, and data.

The presentation tier is the top-most level of the application: it is the user interface, this is what the user sees and interacts with (Stackify, 2019). The main task of this layer is to translate tasks and results that a user can understand. This is where all the data manipulation and data entry happens. In this layer, the languages used are HTML, CSS, and JavaScript.

The application logic tier is the middle layer of the 3T architecture. This layer coordinates the application, processes commands, performs calculations, and makes logical decisions and evaluations. This tier reads and writes data into the data tier. In this layer languages used are for example PHP.

The data tier is where all the data used are stored and retrieved. Here one is able to store data securely, do transactions and search through values and volumes. After retrieval, the data is passed on to the logic and after that the presentation tier.

There are many advantages of using a 3 tier architecture: every layer can be secured differently and separately using different methods. These layers can be managed separately by adding and modifying their content or introducing new features without affecting the other tiers. Moreover, the layers are scalable and flexible as well. Lastly, the tiers can be reused later for other software projects.

4.2.2. User interface design

Designing the User Interface must take place before starting to code and the reason is it can be shown to the stakeholders so that they can comment it. There may be a difference between the design and

the final outlook of the system, especially if only low fidelity designs were shown to the stakeholders, however, all these changes need to be communicated with the client.

Before explaining the decision made during the design, there are two different types of prototyping that need to be distinguished: low fidelity prototyping and high fidelity prototyping. Low fidelity prototyping is simple and it is a low-tech concept (Esposito, 2018). Its goal is to turn ideas into testable artifacts. High fidelity prototypes are highly functional and interactive. They are close to the final version and used in the later stages when the usability needs to be tested and issues in the workflow need to be identified. The advantages of low fidelity prototyping are that it focuses on design and concept, and more energy can be spent on ideation and not on the technical parts. The benefits of the high fidelity prototyping are that it looks like a live software meaning the users would behave more naturally during testing and during testing, there is a possibility to dive deep into specific parts (e.g. flow, navigation) and receive detailed feedback from the users. In many cases, the low fidelity prototyping is done using pen and paper.

When prototyping the user interface, the first decision is about the type of prototyping. Here low fidelity prototyping was chosen simply because of time constraints. There was a need to receive feedback from the stakeholders to ensure delivery of an interface the user prefers and is able to use. However, to make it more professional and transparent, the design was not prepared by pen and paper but with using mock-ups. The mock-ups were prepared by a software called InVision Studio since it is easy to use and has the best ratings.

The second decision point is regarding the qualities are that needed to be shown by the system. According to section 2.3, those systems are the best that are simple, easy to use and navigate and provide a lot of graphical representations which then help users to make fast and efficient decisions. Another decision was that the system should be reminiscent of the University website. The logic behind that is that the stakeholders are University employees, and by making the ISMS similar to the University website suggests the same brand. There are two obvious ways to help this: with colours and with the University crest. That is why the primary colour on the developed website is the same as the University website and the crest can be found on every page of the application (although this is something that was added to the website after the low fidelity design). Because the feedback from low fidelity prototypes was needed before the start of the first iteration, only those parts of the application that were planned for the first iteration (Login and Assets page) were included. Figure 11 shows the first planned layout. The colours are those that the University website uses. On the right-hand side, the user can manage his/her account (if the password is forgotten, it can be changed, the account can be deleted and the users can logout here). Below the navigation bar, there is a welcome message that

describes what the users can do in that page, on the right-hand side there is a button with a blue background where new assets can be created. The table is the main object of the page which shows the current assets that belong to that user (if the user would be an admin then there would be another column called 'Owner'). The column names are underlined since by clicking on them that column is ordered by in ascending order. In the 'Update' column there are two links in every row: edit and delete. When clicking on delete there is a small confirmation message pop-up since users can accidentally delete the wrong data which may result in unwanted data loss.

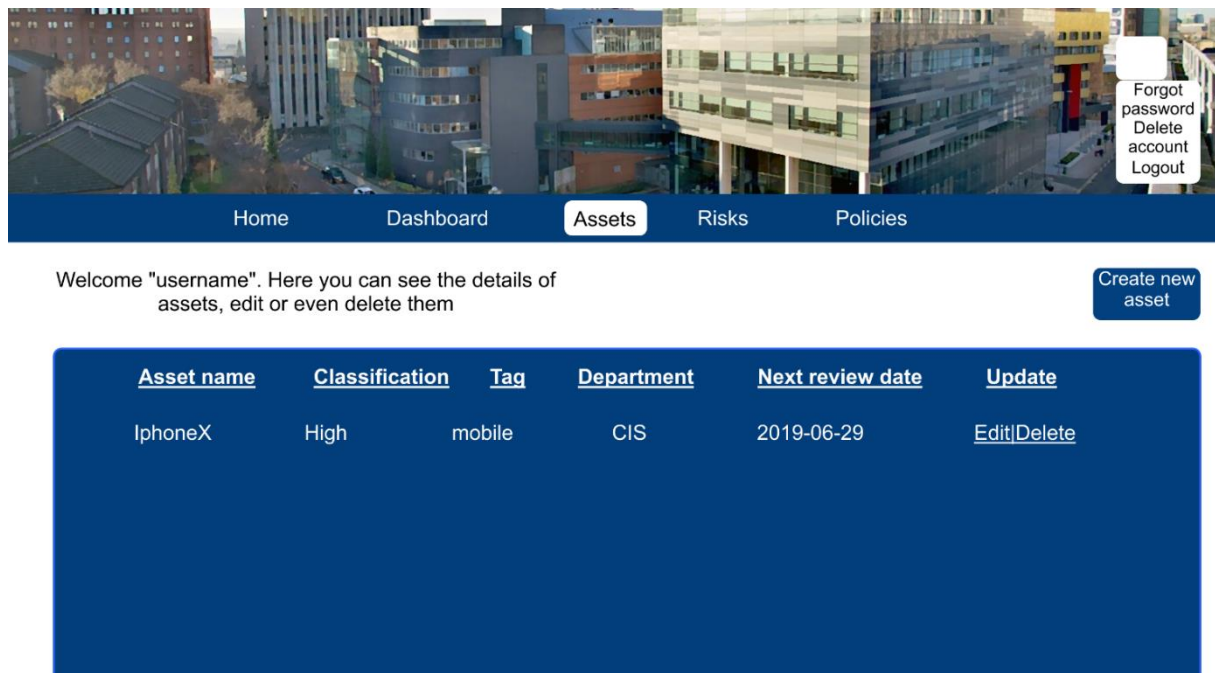


Figure 11. Asset page design

If the users click on Edit, a form which can be found in Figure 12 opens in a new page. Here the colours used are the same as elsewhere. If one of the data input is not correct, an error message is shown after that row. By clicking on submit, the user is taken back to the original page (in this case assets). There is one change that was implemented after the low fidelity design had been finalized: once an asset/risk/policy is created, it cannot be changed. That is why a new value was created which is the 'Owner'. Originally the 'Owner' and the 'Created by' values were the same, but everything is sorted based on the 'Owner' value and that can only be changed by admins, not users. The other pages except 'Dashboard' look the same. With these decisions on design, the goals that have been set earlier can be reached: it should be simple, informative, and easy to use and navigate.



Edit/Create new assets

Assets name:

Classification:

Tag:

Department:

Next Review Date:

Created by:

Submit

Figure 12. Create/edit new assets

4.2.3. Database design

When designing a database, the first step is to have available requirements on which the database tables are based. The system needs to manage data about assets, risks, policies, and users. It also needs an event log which is a derivation of actions happening in the system. There are five entities in the database and that means an entity-relationship diagram can be created (see Figure 13).

The Database Management System that is used is a MySQL database which is available through PHPMYAdmin. This was provided by the University. The primary key of every table is the id field. Although the name fields are unique (username, asset name, risk name, policy name), because of the facts that id is an integer and it auto increments once a new element is created, this makes it easier to use and is more transparent. Every user might create/is responsible for one or more assets, risks and policies, so there the relationship is one to many mandatory on the one. Users table (see table 3) is the most important table amongst all the tables because it holds the login name, the hashed passwords as a varchar variable, and the name of the person, the email address, the user type whether one is admin or user and the user has left columns. Assets, risks, and policies have two columns from this table: created_by and the responsible. Originally, when e.g. an asset is created, the value is of these columns is the same except that while created_by holds the username, responsible holds the real name of the person. The reason is that responsible column can be changed, but the other one cannot be. Because of this, there is a cascade on update and restrict on delete constraint on every responsible column.

Cascade on update means that if a name is changed, the change follows in other tables as well where that name is present (since it is a foreign key).

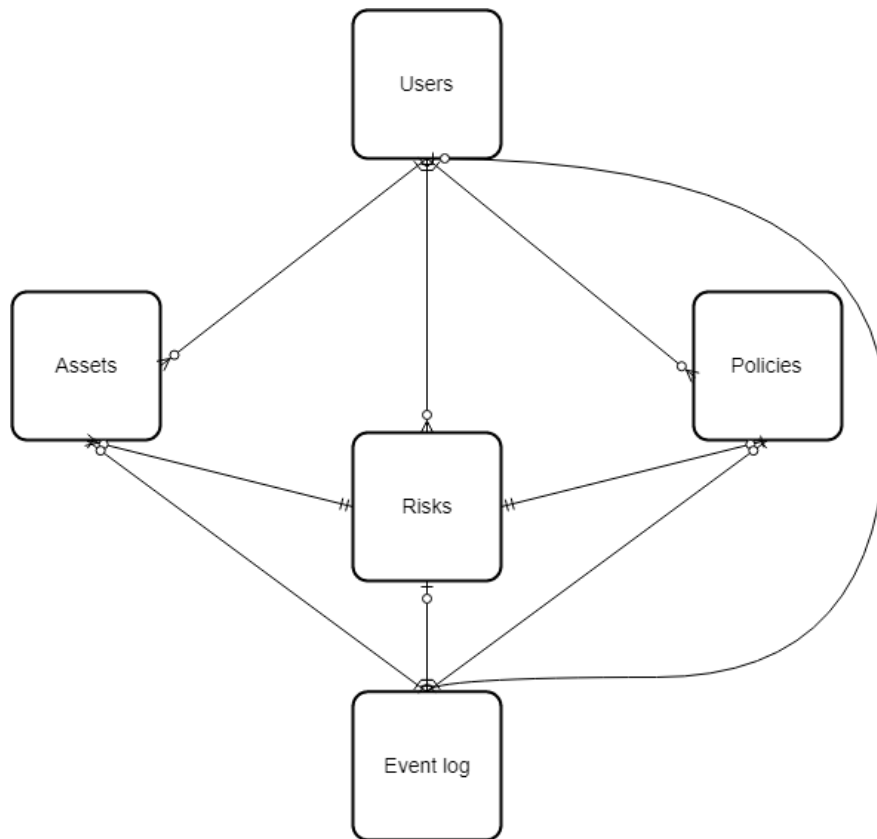


Figure 13. Entity-relationship diagram

Regarding restrict on delete constraint, a user cannot be deleted from the database as long as there is at least one asset/risk/policy he is responsible for. The user who created these is important as well, but not as important as the owner of the elements. That is why there are constraints only on the responsible column.

Table 3. Users

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id 🔑	int(11)			No	None		AUTO_INCREMENT
2	username 🔑	varchar(50)	latin1_swedish_ci		No	None		
3	password	varchar(255)	latin1_swedish_ci		No	None		
4	name 🔑	varchar(255)	latin1_swedish_ci		No	None		
5	email	varchar(255)	latin1_swedish_ci		No	None		
6	created_at	datetime			No	CURRENT_TIMESTAMP		
7	user_type	varchar(100)	latin1_swedish_ci		No	None		
8	userhasleft	varchar(50)	latin1_swedish_ci		No	None		

The asset table (see Table 4) has every information about the assets the stakeholders have requested. There is one feature that is used elsewhere, which is the tag column. That is the reason that the relationship between risks and assets is one mandatory to many optional. The logic behind it is that similar assets gather into bigger categories, that is why a tag is used e.g. if there are three laptops with different names then they have a laptop tag. It is important because the main goal of the product is to mitigate risks. The simplest way to do is to gather the assets into categories which then have a certain risk level. By calculating risk for every asset that is present in the company would also be difficult to ensure consistency in the risk assessment for similar assets. By simplifying the process the user is allowed to use the system in a simpler and clearer way so that mistakes can be avoided. Because of this process, there is a cascade on update and restrict on delete constraint on the asset tag meaning that if an asset tag is changed in Asset Management page it is changed in Risk Management page as well and cannot be deleted as long as it has a risk assigned to it.

Table 4. Assets

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id 🔑	int(11)			No	None		AUTO_INCREMENT
2	Asset_name	varchar(30)	latin1_swedish_ci		No	None		
3	Classification	varchar(10)	latin1_swedish_ci		No	None		
4	Tag 🔑	varchar(30)	latin1_swedish_ci		No	None		
5	Department	varchar(90)	latin1_swedish_ci		No	None		
6	Review_date	date			No	None		
7	Created_by 🔑	varchar(255)	latin1_swedish_ci		No	None		
8	Responsible 🔑	varchar(255)	latin1_swedish_ci		No	None		
9	Time_created	timestamp			No	CURRENT_TIMESTAMP		

The policies table (see table 5) has been created in order to help mitigate the risks. Since one policy is able to control only one risk, the relationship between these two tables is one to one mandatory on both sides. The constraints are the same as earlier, but in this case for policy name, meaning that a policy name change results in a change of that name in risk table and it cannot be deleted from the policy table as long as it is assigned to a risk.

Table 5. Policies

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id 🔑	int(11)			No	None		AUTO_INCREMENT
2	name 🗝️	varchar(255)	latin1_swedish_ci		No	None		
3	created_by	varchar(255)	latin1_swedish_ci		No	None		
4	responsible 🗝️	varchar(255)	latin1_swedish_ci		No	None		
5	review_date	date			No	None		
6	link_to_policy	varchar(255)	latin1_swedish_ci		No	None		
7	timestamp	timestamp			No	CURRENT_TIMESTAMP		


The risks table (see table 6) consists of several columns and it is an aggregative table meaning that it contains every information that is needed to understand how risks are mitigated in the system.

Table 6. Risks

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id 🔑	int(11)			No	None		AUTO_INCREMENT
2	name	varchar(255)	latin1_swedish_ci		No	None		
3	description	text	latin1_swedish_ci		No	None		
4	riskscenario	text	latin1_swedish_ci		No	None		
5	created_by	varchar(255)	latin1_swedish_ci		No	None		
6	responsible 🗝️	varchar(255)	latin1_swedish_ci		No	None		
7	tag	varchar(255)	latin1_swedish_ci		No	None		
8	impact	int(11)			No	None		
9	likelihood	int(11)			No	None		
10	category	varchar(255)	latin1_swedish_ci		No	None		
11	policy 🗝️	varchar(255)	latin1_swedish_ci		No	None		
12	residualrisk	int(11)			No	None		
13	targetrisk	int(11)			No	None		
14	review_date	date			No	None		
15	assettag 🗝️	varchar(255)	latin1_swedish_ci		No	None		
16	timestamp	timestamp			No	CURRENT_TIMESTAMP		

The event log table (see table 7) does not have any independent fields, moreover, there are no restrictions because it stores historical actions meaning that even if a value is deleted from the database, there is no need to delete it from this table. It shows the name, the action type, the section, the actor and the timestamp of every action that has happened in the system so far. The name is the entity's name, action types can either be new, delete or edit, and the place is where the entity belongs to (risk, asset, policy or user).

Table 7. Event log

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id 	int(11)			No	None		AUTO_INCREMENT
2	name	varchar(15)	latin1_swedish_ci		No	None		
3	section	varchar(15)	latin1_swedish_ci		No	None		
4	action	varchar(15)	latin1_swedish_ci		No	None		
5	responsible	varchar(255)	latin1_swedish_ci		No	None		
6	timestamp	timestamp			No	CURRENT_TIMESTAMP		

Lastly, when deciding on the database design another option would have been present besides restricting on delete relationship between the tables and that is cascade on delete. The difference between these is that while restrict on delete does not allow to delete any rows from the parent table as long as that value is present in the child table (e.g. policy table as the parent table and risks table as a child table), cascade on delete automatically deletes that row from the child table. The latter one is more drastic and when designing a database it always needs to be kept in mind that care must be taken and avoiding to delete even unnecessary data is more important than deleting data without one more confirmation from the user.

4.3. Low-level design

In this section the coding part of the implementation is illustrated: what kind of programming languages and tools have been used, the structure of the code is shown and what are those decisions that are made during the implementation.

4.3.1. Programming languages and tools used

For each tier of the 3-tier architecture, the following technologies were used. For the backend, MySQL is used which is managed by PHPMyAdmin. For the middle tier, PHP is used while for the frontend, HTML, CSS, and JavaScript are used. There are no files whose extension is HTML, only files with PHP, CSS or js extensions. The PHP and JavaScript used are vanilla languages meaning that there are no frameworks that are used (for example Laravel for PHP and Node.js for JavaScript).

Before describing the structure of the code, the tools used should be defined. There are two tools that have been used while developing the application. Here the coding solution is not described, only the decisions made in favour of these tools. One of the tools is the Google Charts API and the other one is DataTables. In 3.2. Requirements gathering section it was specified that dashboard should be included in the final solution. In this dashboard three diagrams are shown. Diagrams should be informative, easy to understand without any prior knowledge and decisions can be based on these. This is the reason

Google Charts have been used: the implementation is fast and easy, the charts are customizable, free to use and secure. This was the best tool available for drawing charts so it is why it was chosen. The other tool that has been used is DataTables. The main purpose of this tool is to show data from a database but in an interactive way. All the columns are sortable, the user can select how many entities should be shown, if there are more entities, they are shown in pages so pagination works and there is a search button. After searching for data, users can sort the remaining values as well. Before using this tool, the solution was that the table that had shown the values could have been sorted and it highlighted those rows where the review date was before the actual date. By using DataTables, this functionality has been lost, however, there are a lot of features that have been implemented thanks to this tool that otherwise would have been extremely circumstantial to implement. Since the users can sort and search for the review date, users have other tools to filter on the non-compliant entities while being able to use the table more interactively and with more features at the same time. As there is a dashboard available this functionality was not as useful. Other advantages of DataTables are that it is easy to implement and it is customizable.

There were two features for which other people's code was used. One of them is the PHP MySQL Login System (TutorialRepublic, 2019). This is the code that was used for inspiration but it had to be adapted when building the login page. It was a sample because changes according to the web application's needs have been implemented, moreover, every code was rewritten from mysqli procedural style into mysqli object-oriented style. This decision has only one reason: with the object-oriented style the coding is more transparent, it is easier to follow what happens. This source was used for the login page, the registration page and for the password reset pages as well. The other source that has been used shows how to populate a drop-down list from MySQL database (This interests me, 2019). This small script is written using a PDO object and it loops through a database column and makes the values available in a drop-down list in a form (an example can be seen in Figure 14). It is extremely useful when the goal is to limit what a user can input, but by pulling the data from a database it makes sure once that column has a new value, it could be chosen to be the users in that drop-down list. The pages where this script is used are the ones where there is a need to edit values: edit assets, edit risks and edit policies. Although admins can edit users, there is no need for this script.

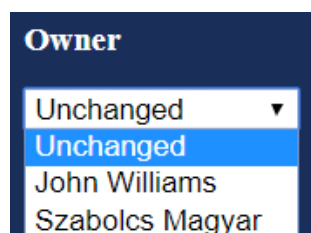


Figure 14. Example of populating a drop-down list

4.3.2. Structure of the code

By making the structure easier to understand not only security oversights could be avoided but possible future changes could be implemented more easily. Moreover, since duplication of the code always needs to be avoided, there are three separate files (besides styling) that do not represent functions but serve the purposes mentioned earlier. These are session.php, adminsession.php, and database.php and are called through the 'require' command of PHP. Session.php is used in those files that every user can access, adminsession.php is called on pages that only admins can retrieve and database.php is required on every page that has a database connection. The styling is external, however, it is inline for showing the error messages. The style.css contains all of the styling information the web application uses. Every feature the system has is in a separate PHP file e.g. risks.php, editrisks.php. The PHP and HTML codes are present at the same time in these files. Although this is not very elegant, it works. There is only one exception from this structure which is the dashboard because by doing so malfunctions can be corrected easily. This is one of the advantages of AJAX (Asynchronous JavaScript and XML) besides that it reads data from the server, updates the server and sends data to the server. The reason there are three files for the 'Dashboard' is that one is the actual webpage, there is a JavaScript file that contains AJAX and there is another PHP file that holds the data. AJAX was needed in this case because implementing a new tool may result in an error and by using that, these errors could have been shown so that the implementation became faster.

4.3.3. Security of the code

HTTPS stands for HyperText Transfer Protocol Secure. This is not only important because this is one of the must-have requirements, but HTTPS uses Secure Sockets Layers (SSL). The reason it is inevitable is that it is the technology that keeps any internet connection secure, it safeguards any sensitive data that is sent between systems and prevents criminals from reading and modifying any information transferred (Symantec Corporation, 2019).

There are some coding practices which have been used in not only one PHP file but in most of the files. The reason for starting with this is that these are not individual solutions but their intention is to make sure the code is easily readable and understandable and it works properly. The Open Web Application Security Project (OWASP) release their top 10 threat paper every four years. In this paper, they not only describe these threats but they have a cheat sheet against these as well. Although it is useful, not all the threats are handled if someone would pay attention to this paper, but most of the most common ones. The SQL injection is the top threat in 2017 (OWASP, 2017). The reason this threat is highlighted at the beginning of the subchapter is that the main activity the application does is to send and retrieve data from a MySQL database which then will be shown, edited or deleted. SQL injection targets these systems where there is a frontend with a database backend and in the forms, malicious SQL codes can

be injected in order to retrieve data the attackers would not have rights to or delete tables, etc. There are some ways to prevent this attack, during the implementation the method that is used is the object-oriented mysqli. The reason mysqli works are that it uses parameters: at first, the SQL statement is prepared and if there is some condition in the 'WHERE' clause then these values are substituted with a question mark (?). After that these variables are bidden to the SQL statement, then the statement is executed and the results can be fetched before closing the connections. An example can be seen in Figure 15: the statement is prepared, then the name variable is bidden to the statement where the first variable shows the type of the bidden value (string, integer, date), then the statement is executed and results can be shown. The reason it works is the parameters: the system handles parameters as entities on their own meaning that even if a quote, a semicolon, etc. are inserted, they mean no harm to the system because these are only parameters of the SQL statement. By using this method during coding SQL injections can be and are prevented which is the biggest threat to web applications.

```
$stmt = $conn->prepare("SELECT * FROM `assets` WHERE Responsible = ?");  
$stmt->bind_param('s', $name);  
$stmt->execute();  
$result = $stmt->get_result();
```

Figure 15. Object-oriented mysqli example

Every form that is present in the system is using the 'Post' method for sending the data. It is vital since the Post requests cannot be cached, bookmarked and they do not remain in the browser's history.

Another coding practice which has been used through every file is that once functionalities are working then the error messages should be deleted from the code. It does not only have practical reasons meaning that they become useless once the code is working but it has security logic as well. If some of the functions would not work and the error messages would have been shown it would mean a piece of information for the attackers.

4.3.4. Login page

The first page a user sees is the login page. The background image is taken from Bleuwire (Cepero, 2018), although the resolution has been changed. The reason this picture has been chosen is that it clearly shows the system is a security audit one. If the user has not registered before, there is a link at the bottom of the form. When registering, a unique username, a name, the email address, a password and the confirmation of the password is needed. The name and password are validated through regular expressions where the password must be at least eight characters long and contain at least one lowercase, one uppercase letter, and one digit. As stated in 4.2.3. Database design, the passwords are stored as varchar, and the hashed value is stored. Hashing is needed for encryption and it uses the current version of it, which is now bcrypt. The code of encryption can be seen in Figure 16.

```
$param_password = password_hash($password, PASSWORD_DEFAULT);
```

Figure 16. Password encryption

The email is validated through filter_var where the second attribute is FILTER_VALIDATE_EMAIL, moreover, in HTML 5 the type of the field can be set in the form and in this case, it is set to email. This field is important since if somebody forgets his password, the admin can email him to this address a new one while the name field helps to identify the people in the system. Every person who creates an account becomes automatically a 'user'. Users are able to see only those entities in the system where they are the owner while admins can see everything. Another field that is automatically set is the user has left the field. This indicates whether the user has deleted his account or not. When creating an account this is set to 'no'.

The session starts and if the user has been logged in then it redirects him to 'Dashboards'. If not, then it checks if the username and password are valid and if yes, then the user is able to log in to the system. There is a huge difference in how error messages (i.e. if the user has input a wrong text) are shown. PHP validation is used for validation inputs and the error messages are PHP codes as well. In other forms, the error messages are shown right below the field which has errors and the source of the error is indicated as well. In the login page by not showing the attacker the source of the problem (username or password), he does not know whether he has input a wrong username or password making the system more secure.

4.3.5. My account functionalities

After a successful login, a user has four pages on the navigation bar (Dashboards, Assets, Risks, and Policies) on the navigation bar while the admin has two more (User management, Event log). On the right-hand side, there is a drop-down list which is presented after hovering over on 'My account'. Here there are three functionalities a user can access: reset the password, delete the account and log out. When a user wants to reset his password, he not only needs to create a new one with the same requirements at registering (minimum eight characters long password with one upper case, one lower case letters, and one digit) but the old password must be input as well. It has security reasons: if for example, somebody leaves the laptop open for a minute, anyone can create a new password for him if there would not be a need to input the older one as well. This solution adds another layer of security to the system. The passwords are stored in the hashed form and after a successful password reset the session is destroyed, a user is redirected to the login page where he can log in again with the new password. If a user would like to delete his account, he can click on 'Delete account'. After that, he sees a confirmation message where he can confirm his intention to delete the account. If he clicks on yes, he is redirected to the login page and he is not able to log in anymore. However, his account is not deleted. As stated earlier in 4.2.3. Database design, there is a column in the 'users' table which

indicates whether the user has left. Originally this value is set to 'no' for every user. Once a user deletes his account, this value is updated to 'yes'. Moreover, his name is updated to 'Deleted' as well, but not only in this table but in all tables as well. This process has two advantages: if the user deleted his account by mistake, the admins are able to undo it. Another advantage is that, as shown later, until a user has at least one entity assigned to him, the account cannot be deleted because of database design constraints. If a user deletes his account, because of the column is updated to 'yes' and his name is to 'Deleted', the admins are able to not only see this change in the 'User management' page but in Asset/Risk/Policy Management pages as well. That is the reason the 'name' is updated in every table: the admins are able to identify those assets/risks/policies whose owner has left the company, so reassigning them is easier. Furthermore, when assigning entities to a new owner, this process minimises the chance of an error if those users are not listed who are deleted from the system by themselves. The third functionality a user can choose from the drop-down list is to log out. Here the session is destroyed and the user is redirected to the login page.

4.3.6. Admin pages


There are two pages that can be seen by only admins. This function is tracked through \$_SESSION variables and is built in not only in the respective pages but in the Navigation bar of other pages as well, that can be seen in Figure 17. This is an example taken from the Asset Management page:

```
<ul>
<li><a class = "notactive" href="dashboard.php">Dashboard</a></li>
<li><a class = "active" href="assets.php">Asset Management</a></li>
<li><a class = "notactive" href="risks.php">Risk Management</a></li>
<li><a class = "notactive" href="policies.php">Policy Management</a></li>
<?php if ($_SESSION['user_type'] == 'admin') {
    echo "<li><a class = 'notactive' href='usermanagement.php'>User Management</a></li>";
    echo "<li><a class = 'notactive' href='eventlog.php'>Event Log</a></li>";
}
?>
</ul>
```

Figure 17. Admin pages in the Navigation bar

In the User Management page, the name, email, user type, and the user has left columns are listed. The reason email is required in the registration is that if a user forgets his password and cannot log in, the admin can email him a new password which can be set in the database and after the successful login the user changes it. To make it more user-friendly, there is a mailto link () in the email column meaning that once an admin clicks on it, his default mailing software is opened with the recipient being that email address. The reason the subject of this email is not automatically set to password reset is that there might be other reasons an admin would like to write to users. In the last column there are two links for each row: edit and delete. When clicking on 'Edit', a new page opens. An example can be seen in Figure 18. The name of the user is highlighted on the top and the two fields that can be changed are the user type and the user has left

values. By highlighting the name and not making it a read-only form field makes the usage of the form obvious and not has the impression that the name of the user can be edited. This solution i.e. that the name is highlighted on the top and is not present as a form field is the same for the remaining pages where editing entities is available. After clicking on 'Update', the user is directed back to User Management page and if there was a change, the result can be seen. When clicking on 'Delete', a confirmation message appears that asks for confirmation of the deletion. It is implemented because admins may click on the deletion of a specific user by accident which would result in unwanted data loss.



John Smith

User type

Admin ▼

User has left?

Yes ▼

Update

Figure 18. Editing users

As defined in 4.3.1. Programming languages and tools used, DataTables are used to show every table coming from the MySQL database. If there would be no other users only the admin who is currently logged in then a short message would be shown which indicates that there are no users to be shown. To implement DataTables, four lines of code is needed to be placed in the head tag of the page, three of them are scripts and one of them is a stylesheet and there is another code snippet that needs to be listed. This can be seen in Figure 19:

```
<script>
    $(document).ready( function () {
        $('#table').DataTable();
    } );
</script>
```

Figure 19. Implementing DataTables

The only variable that is changed is the #table which is the id of the table shown on the page. Before this tool the solution consisted only the sorting of the table, however, it highlighted those rows in red where the user has left. That helped decision making. By implementing DataTables this functionality has been lost, on the other hand, admins can search for 'yes' in the search field which would result in

the same result as highlighting the rows with more functionalities that can be used to make the user interface more advanced.

The other page that can be only seen by admins is the Event log. Once somebody creates/edits/deletes a new entity or a user, it will be shown in the Event log table that can only be seen by admins. Although logins are not tracked and after editing the field the admin is not able to know which features have been edited, the name of the entity/user, the action type (new, edit, delete), the place of the action (assets, risks, policies, users) and the person who did that indicated with his name and the timestamp of the action is logged in to the system.

4.3.7. Functionalities available for every user

In Asset Management page a user can see those assets where he is the owner (an admin is able to see every asset with their owner listed as a new column). On the right-hand side, new assets can be created. By clicking on this button, a new page is opened. The asset name should be unique. Asset name and tag can contain only number and letters. This is tested through ctype_alnum () command and if this requirement is not satisfied then after clicking on the submit button, an error message in red is shown and indicates the source of the error. The classification of the asset is either high, medium or low and can be chosen from a list. Since one of the requirement has been to include not only the departments from the University but other departments as well e.g. HR, finance, etc., these can be found in a list. Next review date helps the auditing procedure meaning that every entity needs to be reviewed frequently which is regulated by the company. This is why this field can be set only later than today (Figure 20):

```
<p class="formLabel">Next Review Date</p>
<input type="date" class="form-control" name="Review_date" min=
<?php
    echo date('Y-m-d');
?>>
```

Figure 20. Next review date

When the user clicks on 'submit' he is redirected to the original page if there are no errors after filling out the form, where the new asset can be seen. Every row can be edited but not deleted unless there is an asset tag that is present in the Risk Management page (restrict on delete constraint). When editing the assets, the name is present on the top and cannot be edited. The users need to be careful when editing tags since the tag is used to group assets and once one of the tags are changed then there might be a situation where that asset has no control meaning that it is not compliant anymore. The department of that asset that has been selected originally is the first element that is shown in the list. That means if the user does not change that field then the original and the edited asset has the same department. However, since the next review date always needs to be later than today and the system

does not show the date that was selected earlier when editing assets, users need to be cautious since changing the review date but not actually reviewing that asset might result in serious consequences in an audit. If an admin edits risks, the owner can be changed as well. The value of 'Owner' is set to unchanged meaning that if the admin does not change that field the owner is the same.

Those fields that are common in entities are using the same solution meaning that for example the name, tag, next review date and owner behave the same way in risks and policies as well. However, deletion of entities is only common in assets and policies and not in risks. The explanation is simple: database constraints. Before implementing this functionality, if there was an asset tag that was present in Risk Management and there was only one asset where this tag was assigned to, if the user wanted to delete the asset, by clicking on delete nothing happened, that asset stayed in the table without describing the reason for the user. Now, when a user clicks on delete and the former situation happens, he is directed to a new page where it is explained that there is a risk that uses this asset tag. Since the message contains two links, the user has two options: he can either return to the Asset Management page or to Risk Management page where he could edit those risks. The database constraints are adapted to the application so that there are no differences between them. This is the process when deleting the policies as well. Figure 21 shows how it is solved in PHP:

```
$stmt3 = $conn->prepare("SELECT Tag FROM assets WHERE id = ? and Tag in (Select Tag from assets WHERE Tag in (SELECT assettag FROM risks))");
$stmt3->bind_param('i', $id);
$stmt3->execute();
$result3 = $stmt3->get_result();

if ($result3->num_rows == 0) {
```

Figure 21. Asset database constraint

By using this query, the sub-query inspects whether that asset tag is present in risk table. If yes, then the number of results is higher than zero meaning that a user receives the message. If the query does not return any rows then the delete statement can be executed. The reason WHERE IN condition works and not WHERE EXISTS is that since Tag is a foreign key for Asset tag meaning that every Tag value exists in the Asset tag field even though they are not present currently. That is the reason WHERE IN needs to be used.

The next page is Policy Management. Here all the controls are shown and stored. When creating a policy, a file needs to be uploaded in the system (that has a name) and the policy has a name as well and these two can be different. The policy name and the uploaded file's name are unique. The policies are uploaded to the DEVWEB/2018/policies folder. At first, the target directory, file name, and file type are set. The file size cannot be higher than 5.000.000 bytes which is a bit less than five MB. There are

only three file types that are permitted to upload into the system: PDF, MS Word, and Docx since policies are mostly stored in these file formats. If one uploads a policy where the extension is different than these, the policy is not created. The policy is then uploaded to the folder through the `move_uploaded_files()` command. Clicking on the Edit button, the policy name and the uploaded file cannot be changed, only the net review date and the owner (if the user is admin). The deletion of the policy works the same way as the asset deletion, the only difference here is that the uploaded file needs to be unlinked from the system. This can be prepared by `unlink()` command. In Policy Management if the user clicks on the name of the policy, a download starts. However, it is not only a link that points to the `DEVWEB/2018/policies` folder because the attackers could then receive valuable information about where the files are saved making the system significantly less secure. This is the reason these links are using a download script that can be seen in Figure 22. The content-type variable is needed because there are three file formats (mime content types) that are permitted and there is only one content type that is allowed in the header. Content-Disposition shows that it is an attachment, the file is not cached and Content-Length shows the file size. `Ob_clean` empties the buffer while `flush` prints out what is in the buffer. After that, a file is read and the download starts. This script is used in Risks as well.

```
$policytodownload = $result2["link_to_policy"];
$contentType=mime_content_type($policytodownload);
if (file_exists($policytodownload)) {
    header("Content-Type: ".$contentType);
    header('Content-Disposition: attachment');
    header("Cache-Control: no-cache");
    header('Content-Length: ' . filesize($policytodownload));
    ob_clean();
    flush();
    readfile($policytodownload);
    exit;
}
```

Figure 22. Download script for policies

The last page where entities can be added is Risk Management. When adding a new risk, there are numerous fields that are needed to be filled out: the name is unique, the description, risk scenario, and tag are text fields where only letters, numbers and white space is allowed to be input. The policy and asset tag are fields from Policy Management and Risk Management, the same script is used for them that allows selecting owners for entities. In Risk Management there are four input fields that use numbers: impact, likelihood, residual risk, target risk. Impact and likelihood follow the risk rating matrix the University uses (University of Strathclyde, 2019). According to this paper, both impact and likelihood are present in a scale of 1-5: regarding impact, the scale is between minor (1) and critical (5) while 1 means rare and 5 means almost certain for likelihood. Based on this the risk category is set: if the score of the multiplication is below 5 then this risk is low, if it is between 5 and 14 then the risk is

medium and above 14 is high. When creating and editing the risks the minimum and maximum values of these fields are set accordingly to the risk rating matrix. Since the maximum score of risks is 25, the residual and target risks follow this methodology as well. When editing the risks, the only difference between this entity and other entities that policy and asset tag need to be selected again (this is the same as the next review date field). Because there are no database constraints, risks can be easily deleted once the user has confirmed his intention.

The remaining page is Dashboard. This page is set up by the Google Charts API. As described in 4.3.1. Programming languages and tools used AJAX and JQuery are used for this page because of error handling. This page uses dashboard.js which is set in the <script> tag in dashboards.php file along with a script which is needed for JQuery. This JavaScript file contains the Google Charts API that draws the charts. At first, the API is loaded with 'corechart' packages, a callback is set to run once the API is loaded and the 'drawcharts' function is created. This function uses Ajax and an example can be seen in Figure 23. The 'URL' shows where the data is coming from, while the data specifies the id that is set in the body of dashboard.php file. Ajax uses JSON and Ajax exchanges data with the server with parseJSON command. Then the column names of the charts, the title, width, and height are set. After that, the chart variable is created where the chart type is specified. At last, the chart is drawn. In dashboarddata.php there are three cases that are the same as the ids set earlier. In every case, two SQL commands are created (one for admins and one for users) and the results of the queries are converted into number with the intval () command. At the end of the file, the data is converted into JSON with the json_encode () command.

```
$.ajax({
  url: "dashboarddata.php",
  type: 'POST',
  data: {
    chart_id: "columnchart"
  }
}).done(function (r) {
  console.log(r);
  r = $.parseJSON(r);
  var data = new google.visualization.DataTable();
  data.addColumn('string', 'Classification');
  data.addColumn('number', 'Number of assets');
  data.addRows(r);
  var options = {
    title: 'Asset classification',
    legend: { position: 'none' },
    width: 900,
    height: 500
  };
  var chart = new google.visualization.ColumnChart(document.getElementById("columnchart"));
  chart.draw(data, options);
});
```

Figure 23. Google Charts with Ajax

The first chart, which is a column chart, shows the number of assets in each asset categories (high, medium, low). The other two charts are pie charts: the second chart shows the percentage of risks where the residual risk is equal or smaller than the target risk and the third chart shows the percentage of those assets whose tag has been assigned to one of the risks meaning that they are compliant. Every asset has a risk and a policy (control) is assigned to the risks to mitigate them. If the asset has no control assigned to them (indirectly) then that means that they are not compliant.

5. Testing

As shown in 4.1.1. Agile, two sprints have been used to develop the application, and at the end of each testing was completed. The reason is that after every iteration the goal is to have working modules and that is why testing is included there. There are five main types of software testing: unit testing, use case testing, load testing, user interface testing and vulnerability testing. These types of testing are described in this chapter. After the testing types are described, the testing strategy is shown with information about how various methods have been applied.

5.1. Testing types

5.1.1. Unit testing

In unit testing, individual units of software are tested (Software Testing Fundamentals, 2019). The purpose is to confirm that each unit performs as designed, where the unit is the smallest testable part of any software. The smallest unit can be a function, procedure (procedural programming) or a method (object-oriented programming). It is a white box testing method. That means that the tester can see the internal structure of the code that is tested and it is like a white/transparent box.

Unit testing has a lot of benefits: it increases confidence in maintaining the code since every time it is changed unit testing needs to be performed in order to make sure the code works. Code needs to be modular if unit testing is used meaning that code is easier to reuse. The development is also faster thanks to the modularity of the code. Because of this, the cost to find errors at this low level is lower than if the errors are detected at higher levels. Since unit testing is a continuous process debugging is easier since only the latest version of the code needs to be tested.

5.1.2. Use case testing

Use case diagrams have been described in 3.1.3. Use case diagrams and in 3.3. Use case diagram sections. Although there is only one use case diagram (Figure 10. in chapter 3.3.), all the different use cases (30 in total) can be found in the Use cases part of the Appendix. Use case testing helps identify test cases that cover entire systems on a transaction by transaction basis from start to finish (ToolsQA, 2019). It ensures that user journeys are working in the system. Regarding the coverage, it not only covers end to end flow (positive test cases) but alternate test cases as well (negative test cases) based on user interactions and the system's responses. Use case testing is based on use cases. It also a black box testing method meaning that the tester cannot see what is inside the software, he only provides inputs verifies results against expected outcomes.

Since process flow is described in use cases describing how the system should work and how it is used, use case testing can uncover integration defects that are caused by incorrect interactions. The reason it is beneficial is that these users would come across these errors when using the system first.

5.1.3. Load testing

The main purpose of load testing is to understand the behaviour of the application under a specific expected load (TRY QA, 2019). It is used to determine a system's behaviour under normal and peak conditions, whether the infrastructure used for hosting is sufficient or not. Load testing shows how many simultaneous users the system can handle. It helps identify the maximum operating capacity with its bottlenecks and which elements cause degradation. The primary goal of load testing is to define the maximum amount of work the software can handle without significant performance loss. It is also non-functional testing meaning that it is mainly used for testing the performance of applications that are web-based and client/web servers. Although some organizations do not perform this testing at all, performance issues can be filtered out with this testing that may prevent the organization from revenue loss.

5.1.4. User interface testing

The user interface is where one interacts with a computer using anything but text (Guru 99, 2019). User interface testing tests the graphical user interface. It involves checking the screen with controls like menus, icons and all types of bars e.g. toolbar, dialog boxes, etc. The focus is on the design structure since this is what the user sees and interacts with. The reason user interface testing is important is that if the user does not understand the user interface that may result in him/her not being comfortable to use the application so proper testing should be carried out. The following elements should be checked: size, position, length and width of elements; error messages (if they are displayed correctly), font used in application, alignment of text, colour of font and error messages (if they are aesthetically enjoyable, images are properly aligned or not) and lastly, positioning of user interface elements for different screen resolution.

5.1.5. Vulnerability testing

Vulnerability assessment is the process where vulnerabilities are defined, identified, classified and prioritized in computer applications, systems and network infrastructures (Rouse, 2019). The main goal is to identify threats and the risks they pose so that security weaknesses can be uncovered and it provides direction on risk assessment associated with those weaknesses and threats. There are five types of vulnerability scans: network-based scans that are used to identify network security attacks. Wireless network scans Wi-Fi networks and it focuses on points of attack in the wireless network infrastructure. Host-based scans are used to locate weaknesses in workstations and servers. Application scans are used to test websites so that software vulnerabilities and erroneous

configurations in the network can be detected. Lastly, database scans are used to identify weak points in a database. Since every organisation faces the risk of cyberattacks, they can benefit from vulnerability testing. Since these vulnerabilities enable hackers to access IT systems, it is crucial to identify and remediate weaknesses before they can be exploited.

5.2. Findings

In 5.1. different testing techniques have been described that are the tests normally a project needs to go through before the evaluation starts. However, in this case, where there is a really strict deadline, only unit and use case testing have been applied. After a code implementation that represents a small function, a unit testing made sure that it works the way it supposed to be. At the end of every sprint when the implementation phase has been finished, all of the uses cases have been looked at. Testing happened based on these use cases and so that errors in the system could have been uncovered. The reason load and user interface testing have not been carried out is the following. As shown in 5.1.3. Load testing is a testing type that is used in most of the cases, but not every time. The reason this testing has not been used is that there are only a few people in the University who might work in this system. That means even in peak conditions the system would work safely since the number of people using the system would be limited. Regarding user interface testing, if there is a time constraint, it might be enough to test is through user evaluation. This is not a best practice since user interface testing needs to happen every time, because of the circumstances, this type of testing has been left for the user evaluation where participants could have commented the user interface if they have uncovered any issues.

Every time new functionality has been implemented, unit testing has been used and if errors were uncovered, the code has been changed and that use case has been tested again. This is called regression testing. Since unit testing has been used continuously, all of the test results cannot be shown, only an example. In 4.3.7. it is shown that the system checks whether the name and tag of an entity contain only numbers and letters with the `ctype_alnum()` command. After this the unit has been tested, the finding has been that although the error message shows that white space cannot be used. Not using whitespace for a name or description field does not make sense. This is the reason this function has been substituted with a regular expression, where the input is validated based on a pattern that can be found in that specific regular expression. Moreover, by implementing it not only English characters, but Chinese, Arabic, etc. letters could be used in names and tags, which might be useful for a user.

5.2.1. Nessus scan

Nessus is a security scanning tool and it identifies the vulnerabilities a network or web application has. It is open-source i.e. free and easy to use that is why it has been chosen to run the vulnerability scan. In 3.2. Requirements gathering one of the non-functional requirements has been to test the implementation for common security vulnerabilities.

Here a basic scan has been run where the following settings have been used: the scan crawled from login.php (not from devweb2018.cis.strath.ac.uk), all HTTP methods have been tried, HTTP parameter pollution have been attempted, all combination of parameters (per each form) have been tested and it looked for all flaws. Moreover, the maximum run has been set to 25 minutes. This indicates the maximum amount spent on each individual generic web attack type. The reason it has been set to 25 minutes because the first time, when the test run, it had been set to five minutes and the report indicated that this number should be increased since there had not been enough time for some of the attack types. The results of the test can be found in Figure 24. In the list below there is only one vulnerability that could have been avoided: the web application is potentially vulnerable to clickjacking, this vulnerability refers to a missing Content-Security-Policy (CSP) on the reset password and dashboards pages, all the other vulnerabilities are server-side vulnerabilities meaning that these could not have been changed. Because of the 'High' severity vulnerabilities, the developed application could not have been used in a professional environment until these would not have been fixed.

Vulnerabilities				Total: 34
SEVERITY	CVSS	PLUGIN	NAME	
HIGH	7.5	123416	phpMyAdmin 4.x < 4.8.5 Multiple Vulnerabilities (PMASA-2019-1) (PMASA-2019-2)	
HIGH	7.5	125855	phpMyAdmin prior to 4.8.6 SQLi vulnerability (PMASA-2019-3)	
MEDIUM	6.9	123642	Apache 2.4.x < 2.4.39 Multiple Vulnerabilities	
MEDIUM	6.8	108758	Apache 2.4.x < 2.4.33 Multiple Vulnerabilities	
MEDIUM	6.8	122060	Apache 2.4.x < 2.4.33 Multiple Vulnerabilities	
MEDIUM	5.0	111788	Apache 2.4.x < 2.4.34 Multiple Vulnerabilities	
MEDIUM	4.3	117807	Apache 2.4.x < 2.4.35 DoS	
MEDIUM	4.3	121355	Apache 2.4.x < 2.4.38 Multiple Vulnerabilities	
MEDIUM	4.3	85582	Web Application Potentially Vulnerable to Clickjacking	
MEDIUM	4.3	125856	phpMyAdmin 4.x < 4.9.0 CSRF vulnerability (PMASA-2019-4)	

Figure 24. Nessus scan result

6. Results and evaluation

The next and last step in software development is the evaluation. After the implementation and testing have taken place, users need to evaluate the software before (in normal case) it goes live. In the chapter, first development results are shown. After that, usability testing is described with its outcomes. Later, requirement-based evaluation follows. After that the systems' security is evaluated with future improvements. Lastly, the reflective evaluation of the development process is discussed.

6.1. Results

In this section, a walkthrough of the system is provided that is supported by screenshots. At first, a user needs to log in that can be seen in Figure 25.

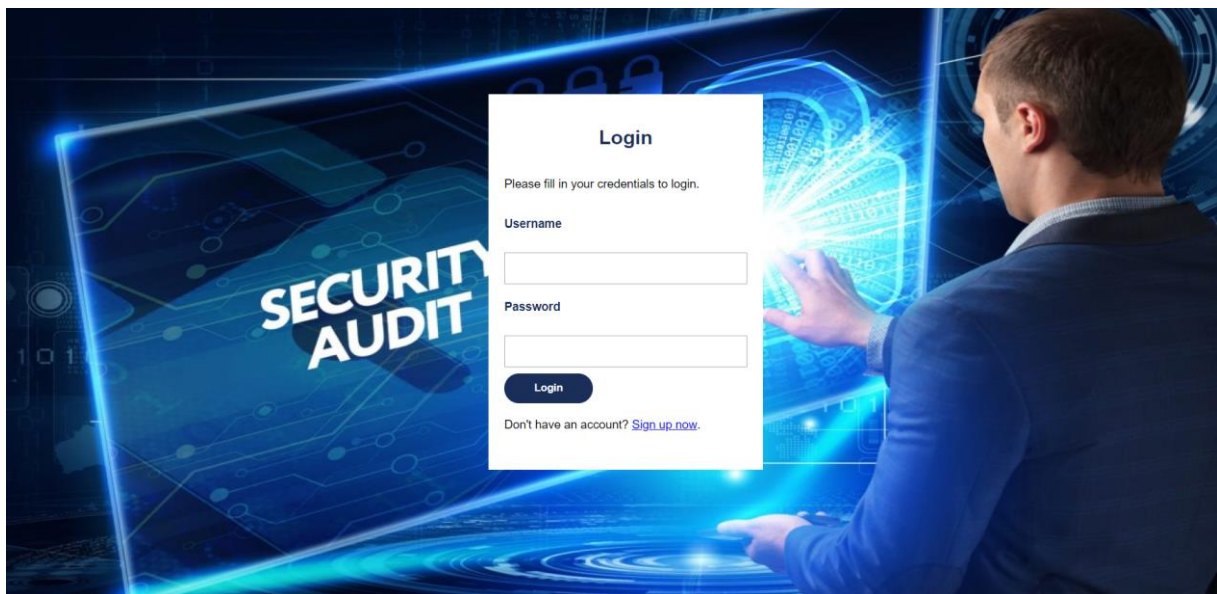


Figure 25. Login page

After the login page, the user is directed to the Dashboard (Figure 26). Since the account that was used in the login is an admin account, all of the functionalities can be seen as well. On the left-hand side, the University logo can be seen. Next to it, the name of the system is present while on the right-hand side a user can reset his password, log out and delete his account. Below that there is a navigation bar where all the other pages can be reached. In this picture, one of the diagrams can be seen.

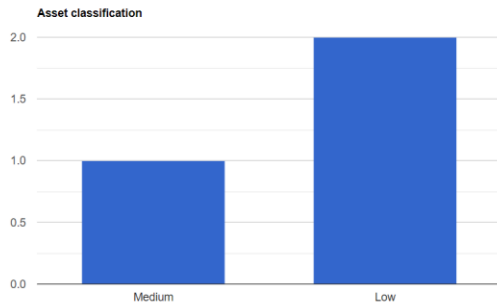


Figure 26. Dashboard

The next page is the Asset management page that can be seen in Figure 27:

Asset name	Classification	Tag	Department	Next review date	Owner	Update
Lenovo z5	Low	Laptop	Accounting & Finance	2019-08-20	John Williams	Edit Delete
Mobile	Medium	Mobile	Work, Employment and Organisation	2019-08-31	Szabolcs Magyar	Edit Delete
Scandisk Ultra Fair	Low	pendrive	Management Science	2019-08-30	John Williams	Edit Delete

Figure 27. Asset management

As shown in 4.3.1. DataTables is used for making the table more interactive. Thanks to this tool, the columns can be sorted, the values can be searched on and the user can select how many entries he would like to see. On the right-side, new assets can be created and that form is shown in Figure 28. The reason it is the only form that is shown in the walkthrough is the layout is the same only form fields are different.

Name

Classification

High

Tag

Department

Accounting & Finance

Next Review Date

Submit

Figure 28. New asset creation

Risk Management page can be seen in Figure 29:

University of Strathclyde

USISS

My account

University of Strathclyde Information Security System

Dashboard
Asset Management
Risk Management
Policy Management
User Management
Event Log

Show 10 entries

Create new risk

Search:

Risk name	Description	Risk scenario	Tag	Impact	Likelihood	Category	Policy	Residual risk	Target risk	Asset tag	Next review date	Owner	Update
Laptop theft	Laptops are stolen	Incident	Theft	5	2	Medium	Laptop Usage Policy	5	4	Laptop	2019-08-25	John Williams	Edit Delete

Showing 1 to 1 of 1 entries

Previous

1

Next

Figure 29. Risk Management

Policy Management is shown in Figure 30:

University of Strathclyde

USISS

My account

University of Strathclyde Information Security System

Dashboard
Asset Management
Risk Management
Policy Management
User Management
Event Log

Show 10 entries

Create new policy

Search:

Policy name	Next review date	Owner	Update
Email	2019-08-31	John Williams	Edit Delete
GDPR	2019-08-29	John Williams	Edit Delete
Laptop Usage Policy	2019-08-12	John Williams	Edit Delete

Showing 1 to 3 of 3 entries

Previous

1

Next

Figure 30. Policy Management

The reason these two pages are shown at the same time is that it can be seen that policy names are links meaning once a user clicks on those, that policy is being downloaded.

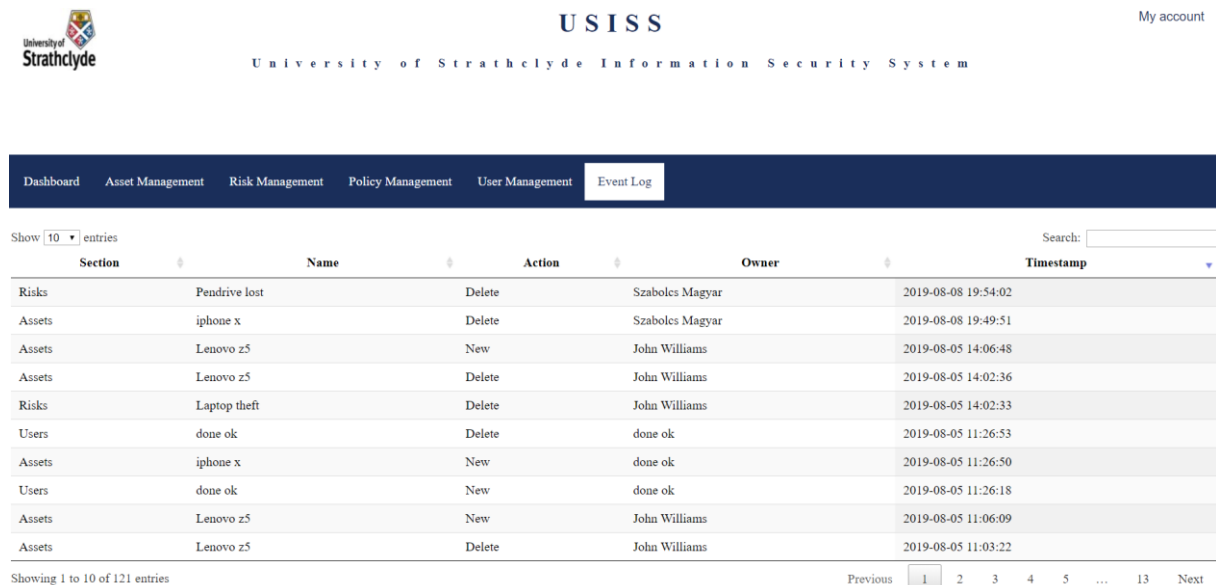
In the User management page (Figure 31) besides editing user rights and deleting users, admins can write direct emails to users. In the example, an example can be seen where the user has deleted himself since the name changed to 'Deleted' and not only in this table but in every asset where he was the Owner.



Name	Email	User type	Have the user left?	Update
Deleted	lucasscott1234@gmail.com	user	yes	Edit Delete
John Williams	johnwilliams055@gmail.com	admin	no	Edit Delete

Figure 31. User management

Lastly, the event log can be seen in Figure 32. The Timestamp column is in descending order. Since there are 121 entries, these entries are paginated into 13 pages (see bottom right corner):



Section	Name	Action	Owner	Timestamp
Risks	Pendrive lost	Delete	Szabolcs Magyar	2019-08-08 19:54:02
Assets	iphone x	Delete	Szabolcs Magyar	2019-08-08 19:49:51
Assets	Lenovo z5	New	John Williams	2019-08-05 14:06:48
Assets	Lenovo z5	Delete	John Williams	2019-08-05 14:02:36
Risks	Laptop theft	Delete	John Williams	2019-08-05 14:02:33
Users	done ok	Delete	done ok	2019-08-05 11:26:53
Assets	iphone x	New	done ok	2019-08-05 11:26:50
Users	done ok	New	done ok	2019-08-05 11:26:18
Assets	Lenovo z5	New	John Williams	2019-08-05 11:06:09
Assets	Lenovo z5	Delete	John Williams	2019-08-05 11:03:22

Figure 32. Event log

6.2. Usability testing

The main goal of usability testing is to evaluate the software by their future users and learning from their feedback. It is crucial since this enables to reach the best possible user experience (Chi, 2019). During the test, participants are asked to complete a set of tasks whilst the organizer watches them navigate the product, listens to their praises and concerns so it will be clearer when participants can clearly and successfully complete the tasks and where they enjoy the user experience and encounter problems. The objectives of usability testing are (Foggia, 2018): gaining insights from users, seeing if the users' expectations have been met, checking if users can perform the tasks and if the design is matching the real-world usage and get user reactions and feedback. It has many advantages over for example a questionnaire (Chi, 2019): it provides an unbiased, accurate and direct examination of the product or website, it is convenient since only a room and notes are needed. Usability testing can tell what users do on the product or website and the reason they take these actions. It also addresses the product's issues in order to prevent developing it in the wrong way. It finds problems where they are easy and cheap to fix. However, usability testing is a continuous process meaning that it needs to be repeated until the design is not confusing.

There are two types of usability testing: qualitative and quantitative. In the study, six participants took part, all of them has worked with similar systems in the past. Since the number of participants should have been a lot higher than six in a quantitative study (because the collected data may have been biased otherwise), qualitative usability test had been chosen. Although the results of the test are not completely objective and reproducible (Mortensen, 2019), it can give an in-depth understanding of products in ways that impossible to reduce to numbers. Since the main goal of the study has been to evaluate the system and their functionalities based on the users' comments, this is something that could not have been expressed with numbers.

The study, which can be seen in the Appendix with the consent form, data management plan, and the actual questions, had two parts: in the first 15 minutes, participants have been asked to perform eight different tasks in the system and notes have been made if these people had any issues using the system or if there were some particular tasks that took more time than expected. In the second part of the study, six questions have been asked from the participants and they have answered these questions. Although the questions that have been asked gathered qualitative data, to make them formal, SUS questions have been used as a basis. SUS stands for System Usability Scale, this is usually a questionnaire at the end of a quantitative usability test. There are 10 questions in total and the participants need to rate every question in a 1-5 scale and after that, the survey is evaluated based on a methodology. These questions have been transformed so that they can be open rather than closed

questions and they have been the basic questions of this study. The participants have been asked to complete the tasks one by one and they started it after their consent has been received although they could have been revoked it during any point of the study. Since the study has been anonymous other details about the participants are not described.

In this paragraph, the outcomes of the study are described except for the fifth question (additional functions that could have been implemented). During the first part of the study, the participants have easily found the functions that they needed to do except for one task. In the sixth task, the participants have been asked to click on the policy name and describe what happens. They clicked on it in every case, however, they did not know what to do after that. The original intent was to show them that after they click on the policy, they can save it and if they open it, it is the same policy that has been uploaded to the Policy Management page. After this had been described to the participants they have understood it and they completed it, so there was an issue with the description and not with the system itself. The completion of every other task has gone as planned, some of them has been slower, but everybody could complete all the tasks. There has been only one time when a participant needed help: when inputting the name of the asset one tried to delete the placeholder value, but since it is a placeholder once one starts typing it vanishes. After explaining it, the completion of the task continued. In the second part of the study, everybody said that the application is fairly/quite easy to use and straightforward. According to the participants the system is not complex to use, the layout is good, plain and extremely consistent, moreover, the modules look the same, however, the layout that is used for new and edit pages of the entities looks like a layout that has been designed for phone usage so that could have been changed. When uploading the policy, the uploaded file's name is present in the form, however, since the text is black (instead of white) it is really hard to read. Moreover, the logo on the web page is squashed. There has been a comment on the font as well: since the system has been designed for the University, not only the colours but fonts could have been the same so as to guarantee that it is the same brand (the University's brand). The last comment concerning the layout has been that since 'Dashboard' is different from the other functions, it could have been shown above or below the other parts in the navigation bar so that the users can understand the difference. Every participant responded that the system would be very useful for them.

Before starting to go through the pages, one participant said that since the sign out button is in a drop-down list that is shown only when the user hovers over the 'My account' button, it is difficult to find so it could be a separate button that is independent of 'My account'. Regarding the 'Dashboard' page more people commented that there could be not only an explanation field beside each diagram that explains the meaning of that chart but a drill-down in the charts as well so when users click on for example on the 'Asset classification' chart 'Medium' column, those assets would be shown who are

classified as medium. Other ideas have been that if a user clicks on that column, he should be redirected to that page with that filter present. Another comment was that when somebody clicks on the Non-compliant risks or assets, there should be a form where the action plan can be tracked where further actions could be listed with dates that describe the way that risk/asset is going to be compliant. Before describing the specific improvements that could be implemented in the other pages, there are some general changes that would be beneficial if they would happen: more participants replied that the review date field is not obvious, because in the USA the date format is different and it is not clear which date format this field uses. One idea has been that instead of showing the months as a number, it should be written so that users do not have the chance to input the wrong review date. The other solution that has been suggested is that instead of manually inputting the date, only the calendar should be shown because it was not obvious for most of the participants that this is an option as well and by choosing this as the only option, the date format will not matter at all. Another improvement has been that if the next review date is near e.g. in five days, the owner of that entity should receive a reminder email automatically generated by the system so that reviews would not be forgotten. A participant has mentioned that when clicking on the policy, the file may be opened in a new tab instead of downloading it. One participant commented that bulk uploading assets (from a CSV file) could be a good addition. Moreover, since the name of the assets might not be unique, a serial ID should indicate the assets. In the Risk Management page, when creating a risk, there should be other controls field than policies meaning technical and physical control as well since they mitigate the risk as well. The residual risk calculation might happen automatically by the system itself and this is something that might be implemented through machine learning. The last improvement that has been advised by the participants is that more clarity should be on how the assets and risks are connected to each other.

6.3. Requirement-based evaluations

Although the software is not only working, but the participants are satisfied with it too, there is a place for improvements. As listed in 3.2. Requirements gathering, one of the requirements was that the users would create custom charts. This functionality would be valuable because different users may have different KPIs that they need to concentrate on and by being able to build custom charts they could utilize the system more. Furthermore, if the software would be used later by other universities or by different firms, this would be one of the first functionalities that need to be implemented at first since their focus on risk management would probably be different. One possible solution could be that the user could choose the diagram type, which columns with what kind of cumulation (sum, count or nothing) he wants and after that the chart is drawn. Another are of improvements concerns the tag column in the asset table. The reason is that this connect assets and risks and if two people write the

same tag but with e.g. different capitalization, these would be two different words for the system although they supposed to be the same. By implementing a system where once a user starts typing, values that are already present could be shown so that duplicates can be avoided. Typeahead is a JavaScript library through that this improvement could be implemented. Another JavaScript related upgrade would be that validation could happen through JavaScript as well. The reason is that users would not have to type in the values in the form twice if they have not used the form as they should have since once a mistake happens the user could see it at the same time and correct it. The last possible improvement concerns forgotten passwords. As shown in 4.3.6. Admin pages, if a user has forgotten his password, an admin can email him a new one so that he can use the system again. A button could be implemented, and if he would click on it, a new page would be shown where only the email address would be needed to input. If this email address could be found in the database, the system would send a user a new password, however, after the first login, he would need to change it because of security reasons. The reason this functionality would be useful is that users would not need to wait for admins to change their passwords, moreover, the workload of admins would be reduced as well.

6.4. Security evaluations

Although the system is secure against SQL injection and there is only one vulnerability that is present because of the coding, there is a huge space for improvements. Against cross-site scripting `htmlspecialchars()` is used in the forms, but using HTML purifier and PHP filter functions would make this more efficient and better. One of the most crucial vulnerabilities of the system is that it has no protection against brute force attacks. Brute force attacks happen when the attackers input many username password combination in order to get access to the system. The first step to prevent them is that there should be a table in the database which tracks login attempts. This table can indicate that the attempt has been successful or not. Based on this, progressive delays might be used i.e. after few unsuccessful logins attempt the system locks the user out for a certain period of time which then increases with each subsequent failed attempt. Another solution that might be useful is the 2-Step verification where the user would need to input a code that he received to his phone number to make sure it is not a bot who tries to log in. As shown in 5.2.1. Nessus scan the system is vulnerable against clickjacking so Content-Security-Policies need to be implemented wherever it is possible. Lastly, when the cookies would be implemented, CSRF and XSFR attacks could be prevented as well although there are a lot of possible solutions for that and they do not work in every situation.

6.5. Reflective evaluation of development process

As the last step, there is a reflective evaluation of the development process. Although there are some aspects of the work done that could have been improved, as shown earlier, if everything would start from the beginning, nothing would be done differently. The reason is simple: every goal has been achieved and the participants were satisfied with the product. In the beginning, there was a huge gap between the coding knowledge and the knowledge required for implementing the requirements, everything that has been included in the final version of the system is knowledge gained throughout the project. Learning by doing it is the best way to learn new things since one gains not only theoretical but practical knowledge at the same time so that he can remember it easily. Moreover, participating in an end-to-end project i.e. from the requirements gathering until the evaluation is something that adds another practical experience that has been gained during the studies.

If there must be two factors highlighted that would require more emphasis during the project are the planning and being able to say no. The first is obvious, especially during an implementation, deadlines cannot be strict enough. There is always a new function that would have been implemented if there would be more time, however, sometimes it is difficult to decide on this since one always wants to do his best. By accepting that everything has been achieved that could have been accomplished during this project and everything worked out perfectly is something that needs to be done and is done.

7. Conclusion

Information Security Management Systems are the topic of the thesis. This chapter has two sections: in the first section research questions are answered and outcomes are shown. In the second section, possible improvements to the developed system derived from sections 6.2, 6.3 and 6.4., are described.

7.1. Summary of thesis contributions

The main objective of this thesis was to develop an Information Security Management System that was based on best practices of asset, risk and policy management from the literature and other ISMS system. All of the best practices of asset, risk, and policy management concern the ownership and transparency of these entities. Moreover, they need to be connected to each other so that the risk that is present in the system through assets could be mitigated. Since there are a lot of standards, the system should support as many standards as possible and have a dashboard that shows graphically the state of the IT audit system. Although every ISMS solution follows the best practices from the literature, its solutions are completely different from each other especially in case of the features that this software contains. The main problem is that in many cases these systems are either too difficult to use or they do not have a good graphical interface that would support the decision-making process. These are the main issues the developed system aims to improve. That means that the developed system should be simple, informative and provide a good graphical interface.

The functionalities a simple ISMS system should support are mostly based on the requirements coming from the stakeholders since every ISMS system is based on the literature, however, the requirements might be different. A system should have a registration and a login page where users can log in to the system. Users should be able to change their password, log out and delete their account as well. If there are admins (and not only users) in the system then these admins should have more rights than users. Admin should be able to manage users (edit their right and delete them) and see what events happened in the system. The proposed system should also have a graphical interface where charts that show the Key Performance Indicators (KPIs) are provided. Moreover, assets, risk, and policies should be able to be created, viewed, edited and deleted and the system should clearly indicate how these are connected to each other. Users should have the opportunity to upload and download policies. Furthermore, since the main goal of the system is to mitigate risk, risk calculation should be transparent for the users.

The best methodology that should be used for developing such a system is Agile. It is an effective choice since the communication happens continuously with the stakeholder, which means that once

an issue/a question arises, it can be solved/answered immediately. Working in iterations according to a risk-based approach ensures that these functionalities are developed first that need more time and deadlines are followed because in most of the cases time is the most important constraint. Every software can be improved and that is why there might a lot of functionalities that could be implemented, however, due to the lack of time, they are not or they are developed in other iterations at a later stage.

A simple system should be built by using the 3 tier architecture: (1) the presentation tier should use HTML, CSS, and JavaScript, (2) the logic tier should be based on PHP and (3) for the back-end, a database should be used for example MySQL. In the database the tables need to be logical, most importantly IDs should be used for identification purposes, the passwords need to be stored in a varchar variable and in a hashed form. If there are columns that are present in more than one table, using foreign keys with constraint is efficient. A good practice is that if a value from the parent field is updated, it is updated in the child table as well, however, a value from the parent table cannot be deleted as long as it is present in the child table.

As a presentation and logic tier, besides using the coding languages shown earlier, there are best practices here as well. Firstly, code that is present in more than one file should be used as a separate file which then can be called. It is useful because these code snippets can be edited easily and make the overall code structure transparent. Using Ajax is another best practice since it facilitates the debugging procedure and again, makes the code more transparent. Regarding charting, Google Charts API is free to use, easy to implement and it is customizable. Another tool is DataTables that adds a huge amount of functionalities to a table that consists of data. It is free to use and customizable as well.

Since an Information Security Management System contains data about system risks, it is crucial that the system should be secure. When the back-end is a MySQL database, it can be secured against SQL injection by using object-oriented mysqli that uses parameters. Since cross-site scripting is a common threat, escaping characters through htmlspecialchars() makes the system more secure. By not showing error messages or not specifying where the error is (e.g. wrong username/password) the attackers cannot use this vulnerability. Forms that send sensitive data back to the server should always use the 'Post' method and passwords need to be stored in a hashed form and as a varchar variable (not as a string). Moreover, the system should be secure against brute force attacks and clickjacking as well.

The system's usability can be ensured by making it simple, straightforward and informative one. The more graphical solutions are used (while keeping in mind the simplicity), the better since improved decisions could be made. A system should be modular so that users could find functions on different

pages easily, colours should be made visible and wherever there is a possibility of misunderstanding, comments should be present that help the users. Lastly, there should be no errors in the system and that can be achieved by unit, use case, load and user interface testing.

7.2. Further development perspectives

There are three categories of improvements: crucial ones, the cosmetic improvements and those that could provide a basis for future projects. This list is based on 6.2, 6.3 and 6.4, they are prioritized.

7.2.1. Crucial improvements

These are improvements that concern mostly security: the system should be secure against brute force attacks (by implementing progressive delays), clickjacking (by implementing Content-Security-Policies) and CSRF and XSFR attacks. Another crucial improvement should be to implement the ability to create custom charts since this has been one of the requirements mentioned in 3.2. and it would add huge value to the product. So would do the ability to add more controls i.e. not only policies but physical and technical controls as well so that risk calculation would be more precise.

7.2.2. Cosmetic improvements

These improvements are connected to the user interface and the explanations of the diagrams. The font should be changed to the University one, the 'Log out' functionality should be shown separately from 'My account', there could be a distinction between Dashboard and other pages. The date field in the forms could use a calendar form and there should be explanations beside each diagram and risk calculations. Moreover, when clicking on an asset, the file might not be downloaded but opened in a new page.

7.2.3. Improvements for future projects

Some of the improvements concern the diagrams: there could be a drill-down opportunity that would filter on data that is clicked on. Moreover, if a risk/asset is non-compliant a table could show the next steps so that these entities are compliant again. Since tag connects assets and risks, by implementing Typeahead, users could see tags already present in the system that would save a lot of time. By having the ability of password reminders users who forgot their password could log back to the system faster. Lastly, the system could send a reminder message to owners whose entities' review date is in the next five days.

References

- Advisera, 2019. *What is ISO 27001?*. [Online]
Available at: <https://advisera.com/27001academy/what-is-iso-27001/>
[Accessed 07 06 2019].
- Agile Business Consortium, 2019. *MoSCoW Prioritisation*. [Online]
Available at: <https://www.agilebusiness.org/content/moscow-prioritisation>
[Accessed 24 06 2019].
- Armerding, T., 2018. *CSO*. [Online]
Available at: <https://www.csoonline.com/article/2130877/the-biggest-data-breaches-of-the-21st-century.html>
- Biswas, P., 2019. *ISMS: Asset Management*. [Online]
Available at: <http://isoconsultantpune.com/isms-asset-management/>
[Accessed 06 07 2019].
- Bowen, P., Hash, J. & Wilson, M., 2006. *Information Security Handbook: A Guide for Managers*. Gaithersburg: NIST .
- Calder, A., 2013. *ISO 27001/ ISO 27002*. 2nd ed. Ely: IT Governance Publishing.
- Calder, A., 2018. *Information Security & ISO 27001*. s.l.:IT Governance Ltd.
- Calder, A. & Watkins, S., 2008. *IT Governance: A Manager's Guide to Data Security and ISO 27001 / ISO 27002*. 4th ed. London and Philadelphia: Kogan Page.
- Capiro, 2019. *Requirements elicitation – 8 Steps to requirements success*. [Online]
Available at: <https://capiro.co.uk/requirements-elicitation-8-steps-to-success/>
[Accessed 09 08 2019].
- Capterra, 2019. *Risk & Audit*. [Online]
Available at: <https://www.capterra.com/p/147599/Investigations-Incident-Software/#reviews>
[Accessed 05 06 2019].
- Capterra, 2019. *StandardFusion*. [Online]
Available at: <https://www.capterra.com/p/151388/StandardFusion/#reviews>
[Accessed 05 06 2019].
- Cepero, R., 2018. *Best Log Management Tools for Security Auditing*. [Online]
Available at: <https://bleuwire.com/best-log-management-tools-for-security-auditing/>
[Accessed 13 08 2019].
- Ceta, N., 2019. *All You Need to Know About UML Diagrams: Types and 5+ Examples*. [Online]
Available at: <https://tallyfy.com/uml-diagram/>
[Accessed 22 07 2019].
- Chi, C., 2019. *The Beginner's Guide to Usability Testing*. [Online]
Available at: <https://blog.hubspot.com/marketing/usability-testing>
[Accessed 06 08 2019].
- Darby, M., 2019. *Information Security Risk Management Explained - ISO 27001*. [Online]
Available at: <https://www.isms.online/iso-27001/information-security-risk-management-explained/>
[Accessed 10 07 2019].

Disterer, G., 2013. ISO/IEC 27000, 27001 and 27002 for Information Security Management. *Journal of Information Security*, Issue 4, pp. 92-100.

Dosal, E., 2018. *Importance of IT asset management*. [Online]
Available at: <https://www.compuquip.com/blog/importance-of-it-asset-management>
[Accessed 06 07 2019].

Dutton, J., 2017. *What is an ISMS and 9 reasons why you should implement one*. [Online]
Available at: <https://www.itgovernance.co.uk/blog/what-is-an-isms-and-9-reasons-why-you-should-implement-one>
[Accessed 04 06 2019].

eramba, 2019a. *eramba*. [Online]
Available at: <https://www.eramba.org/>
[Accessed 05 06 2019].

eramba, 2019b. *Enterprise Subscription*. [Online]
Available at: <https://www.eramba.org/enterprise-subscription/>
[Accessed 05 06 2019].

Eramba, 2019c. *Demo*. [Online]
Available at: <https://demo.eramba.org>
[Accessed 05 06 2019].

Eriksson, U., 2012. *Functional vs Non Functional Requirements*. [Online]
Available at: <https://reqtest.com/requirements-blog/functional-vs-non-functional-requirements/>
[Accessed 22 07 2019].

Esposito, E., 2018. *Low-fidelity vs. high-fidelity prototyping*. [Online]
Available at: <https://www.invisionapp.com/inside-design/low-fi-vs-hi-fi-prototyping/>
[Accessed 26 07 2019].

Foggia, L., 2018. *Usability testing: what is it and how to do it?*. [Online]
Available at: <https://uxdesign.cc/usability-testing-what-is-it-how-to-do-it-51356e5de5d>
[Accessed 06 08 2019].

Gonçalves, L., 2019. *WHAT IS AGILE METHODOLOGY*. [Online]
Available at: <https://luis-goncalves.com/what-is-agile-methodology/>
[Accessed 23 07 2019].

Guru 99, 2019. *GUI Testing Tutorial: User Interface (UI) TestCases with Examples*. [Online]
Available at: <https://www.guru99.com/gui-testing.html>
[Accessed 03 08 2019].

Haughey, D., 2019. *MOSCOW METHOD*. [Online]
Available at: <https://www.projectsmart.co.uk/moscow-method.php>
[Accessed 24 06 2019].

Hsu, C., Wang, T. & Lu, A., 2016. *The Impact of ISO 27001 Certification on Firm Performance*. Koloa, IEEE.

ins2outs, 2019. *How to Implement an Information Security Management System*. [Online]
Available at: <https://ins2outs.com/implement-information-security-management-system/>
[Accessed 06 03 2019].

ISO 27001 Security, 2019a. *ISO/IEC 27001:2013 - Information Technology - Security techniques - Information security management systems - Requirements (second edition)*. [Online]
Available at: <https://www.iso27001security.com/html/27001.html>
[Accessed 07 06 2019].

ISO 27001 Security, 2019b. *ISO/IEC 27002:2013 - Information technology - Security techniques - Code of practice for information security controls (second edition)*. [Online]
Available at: <https://www.iso27001security.com/html/27002.html>
[Accessed 07 06 2019].

Klassen, M., 2018. *How to Select the Right ISMS Software for Your Organisation?*. [Online]
Available at: <https://www.cherwell.com/library/blog/how-to-select-the-right-isms-software-for-your-organization/>
[Accessed 05 06 2019].

Kosutic, D., 2019. *The basic logic of ISO 27001: How does information security work?*. [Online]
Available at: <https://advisera.com/27001academy/knowledgebase/the-basic-logic-of-iso-27001-how-does-information-security-work/?icn=free-knowledgebase-27001&ici=top-the-basic-logic-of-iso-27001-how-does-information-security-work-txt>
[Accessed 06 07 2019].

Kothari, A., 2019. *Policy Management: What Is It and Why Is It Important?*. [Online]
Available at: <https://tallyfy.com/policy-management/>
[Accessed 10 07 2019].

Lomas, E., 2010. Information governance: information security and access within a UK context. *Records Management Journal*, 20(2), pp. 182-198.

Lucidchart, 2019a. *What is an Entity Relationship Diagram?*. [Online]
Available at: <https://www.lucidchart.com/pages/er-diagrams>
[Accessed 24 06 2019].

Lucidchart, 2019b. *UML Use Case Diagram Tutorial*. [Online]
Available at: <https://www.lucidchart.com/pages/uml-use-case-diagram>
[Accessed 22 07 2019].

McLachlan, P. E., 2011. *Five Steps for Effective IT Policy Management*. [Online]
Available at: <http://www.baselinemag.com/c/a/IT-Management/Five-Steps-for-Effective-IT-Policy-Management-236349>
[Accessed 10 07 2019].

Mortensen, D., 2019. *Best Practices for Qualitative User Research*. [Online]
Available at: <https://www.interaction-design.org/literature/article/best-practices-for-qualitative-user-research>
[Accessed 06 08 2019].

NI Business Info, 2019a. *IT risk management*. [Online]
Available at: <https://www.nibusinessinfo.co.uk/content/what-it-risk>
[Accessed 10 07 2019].

NI Business Info, 2019b. *Different types of IT risk*. [Online]
Available at: <https://www.nibusinessinfo.co.uk/content/different-types-it-risk>
[Accessed 10 07 2019].

Ohki, E. et al., 2009. Information Security Governance Framework. WISG.

OWASP, 2017. *OWASP TOP 10 - 2017*. [Online]
Available at: https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf?utm_referrer=https://www.google.com/
[Accessed 29 07 2019].

ProductPlan, 2019. *Product Management: MoSCoW Prioritization*. [Online]
Available at: <https://www.productplan.com/glossary/moscow-prioritization/>
[Accessed 24 06 2019].

QASymphony, 2019. *Agile Methodology: The Complete Guide to Understanding Agile Testing*. [Online]
Available at: <https://www.qasymphony.com/blog/agile-methodology-guide-agile-testing/>
[Accessed 23 07 2019].

Rahman Ahlan, A. & Arshad, Y., 2012. Understanding Components of IT Risks and Enterprise Risk Management . In: J. Dr. Emblemsvag, ed. *Risk Management for the Future - Theory and Cases*. s.l.:InTech, pp. 297-318.

Rapid7, 2019. *Information Security Risk Management*. [Online]
Available at: <https://www.rapid7.com/fundamentals/information-security-risk-management/>
[Accessed 10 07 2019].

Resolver, 2019. *Resolver*. [Online]
Available at: <https://www.resolver.com/>
[Accessed 04 06 2019].

Rouse, M., 2019. *vulnerability assessment (vulnerability analysis)*. [Online]
Available at: <https://searchsecurity.techtarget.com/definition/vulnerability-assessment-vulnerability-analysis>
[Accessed 13 08 2019].

smartdraw, 2019. *Entity Relationship Diagram*. [Online]
Available at: <https://www.smartdraw.com/entity-relationship-diagram/>
[Accessed 24 06 2019].

Software Testing Fundamentals, 2019. *Unit Testing*. [Online]
Available at: <http://softwaretestingfundamentals.com/unit-testing/>
[Accessed 03 08 2019].

Stackify, 2019. *What is N-Tier Architecture? How It Works, Examples, Tutorials, and More*. [Online]
Available at: <https://stackify.com/n-tier-architecture/>
[Accessed 24 07 2019].

StandardFusion, 2019a. *StandardFusion*. [Online]
Available at: https://www.standardfusion.com/?gclid=Cj0KCQjwrdjnBRDXARIsAEcE5YkjTUfVf-iHexwIBmDCvkDICI9-aAyf8rVqaTRzYGUPIdDuza_CvcaAiLdEALw_wcB
[Accessed 04 06 2019].

StandardFusion, 2019b. *Pricing*. [Online]
Available at: <https://www.standardfusion.com/pricing/>
[Accessed 05 06 2019].

Symantec Corporation, 2019. *What is SSL, TLS and HTTPS?*. [Online]
Available at: <https://www.websecurity.symantec.com/security-topics/what-is-ssl-tls-https>
[Accessed 07 08 2019].

Team Linchpin, 2019. *A Beginners Guide To The Agile Method & Scrums*. [Online]
Available at: <https://linchpinseo.com/the-agile-method/>
[Accessed 23 07 2019].

This interests me, 2019. *PHP: Populating a drop down list from MySQL..* [Online]
Available at: <https://thisinterestsme.com/populate-dropdown-list-mysql/>
[Accessed 27 06 2019].

ToolsQA, 2019. *Use Case And Use Case Testing in Software Testing*. [Online]
Available at: <https://www.toolsqa.com/software-testing/use-case-and-use-case-testing-in-software-testing/>
[Accessed 03 08 2019].

TRY QA, 2019. *What is Load testing in software testing? Examples,How To Do,Importance, Differences*. [Online]
Available at: <http://tryqa.com/what-is-load-testing-in-software/>
[Accessed 03 08 2019].

TutorialRepublic, 2019. *PHP MySQL Login System*. [Online]
Available at: <https://www.tutorialrepublic.com/php-tutorial/php-mysql-login-system.php>
[Accessed 28 06 2019].

University of Strathclyde, 2019. *Appendix 4: Risk Rating Matrix*, Glasgow: s.n.

VComply Editorial, 2018. *Why policy management needs to be a crucial part of your organization*. [Online]
Available at: <https://blog.v-comply.com/policy-management/>
[Accessed 10 07 2019].

The Appendix

Use cases

ID:	1
Title:	Registration
Description:	The user can register an account
Primary Actor:	User/Admin
Preconditions:	The user has not created an account before
Postconditions:	The user is able to use the login page.
Main Success Scenario:	The user inputs a username, his name, email address, a password, and password confirmation. After clicking on submit, the user is redirected to the login page.
Extensions:	Error messages which indicate the following: <ul style="list-style-type: none">• Either that there is a field a user has input nothing OR:• The username is not unique (= it is in the system already)• The name is invalid• The email address is invalid• That password must be at least 8 characters and must contain at least one lower case letter, one upper case letter, and one digit.• Passwords do not match
Frequency of Use:	Every time a new user account is created.
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	2
Title:	Log in
Description:	The user can log in to the system.
Primary Actor:	User/Admin
Preconditions:	The user has created an account before.

Postconditions:	The user is able to use the system.
Main Success Scenario:	The user inputs his username and password. After clicking on Login, the user can see the functionalities of the system.
Extensions:	<p>Error messages can indicate the following:</p> <ul style="list-style-type: none"> • Either that there is a field a user has input nothing OR: • The account has been deleted. In this case, the user needs to create a new account • The username or password is invalid
Frequency of Use:	Every time a user wants to use the system
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	3
Title:	Reset password
Description:	The user would like to change his password
Primary Actor:	User/Admin
Preconditions:	The user is logged in
Postconditions:	The password has changed and the user is redirected the login page
Main Success Scenario:	The user inputs his current password, his new password and the confirmation of it. After clicking on submit, the user is redirected to the login page where he needs to use this new password
Extensions:	<p>Error messages can indicate the following:</p> <ul style="list-style-type: none"> • Either that there is a field a user has input nothing OR: • That password must be at least 8 characters and must contain at least one lower case letter, one upper case letter, and one digit. • Passwords do not match. • The current password does not match with the one that has been input
Frequency of Use:	Every time a user wants to change his password
Status:	Developed

Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	4
Title:	Log out
Description:	The user would like to sign out of the system
Primary Actor:	User/Admin
Preconditions:	-
Postconditions:	The user is redirected the login page
Main Success Scenario:	The user clicks on the log-out and he is redirected to the login page (with his session being destroyed)
Extensions:	
Frequency of Use:	Every time a user wants to sign out
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	5
Title:	Delete account
Description:	The user would like to delete his account
Primary Actor:	User/Admin
Preconditions:	The user is logged in
Postconditions:	The user is redirected to the login page and he is not able to sign in with those credentials anymore.
Main Success Scenario:	The user clicks on Delete account. A confirmation box is shown where he can confirm the deletion. After that, he is redirected to the login page where he is not able to use his previous credentials to log in anymore.
Extensions:	

Frequency of Use:	Every time a user wants to delete his account
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	6
Title:	Viewing dashboard page
Description:	The user would like to view the 'Dashboard' page where the charts can be found
Primary Actor:	User/Admin
Preconditions:	The user is logged in
Postconditions:	The user is able to make decisions based on the numbers he sees on the diagrams
Main Success Scenario:	The user clicks on the Dashboard. Here he sees 3 charts: asset classification, risk compliance, and asset compliance. He sees the numbers as well and he can make decisions (if needed) based on this data)
Extensions:	
Frequency of Use:	Every time a user wants to see the diagrams
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	7
Title:	Viewing the asset page
Description:	The user would like to view the assets present in the system
Primary Actor:	User
Preconditions:	The user is logged in
Postconditions:	The user is able to identify the assets present in the system.

Main Success Scenario:	The user clicks on Asset management. Here he can see that he owns: all the assets with its name, classification, tag, the department they belong to, the next review date. If there are no assets currently in the system, a message is shown that describes that state.
Extensions:	
Frequency of Use:	Every time a user wants to see the assets present in the system
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	8
Title:	Viewing the asset page
Description:	The admin would like to view the assets present in the system
Primary Actor:	Admin
Preconditions:	The user is logged in
Postconditions:	The user is able to identify the assets present in the system.
Main Success Scenario:	The user clicks on Asset management. Here he can see every asset with its name, classification, tag, the department they belong to, the next review date and the owner as well. If there are no assets currently in the system, a message is shown that describes that state.
Extensions:	
Frequency of Use:	Every time a user wants to see the assets present in the system
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	9
Title:	Create new assets
Description:	The user would like to create new assets

Primary Actor:	User/Admin
Preconditions:	The user is logged in.
Postconditions:	The new asset is created which can be seen in the Asset Management page
Main Success Scenario:	The user clicks on Create new asset. He inputs a unique name and a tag, he chooses a classification, the department, and the next review date. After clicking on submit, he is directed to the Asset Management page where he can see this new asset
Extensions:	<p>Error messages can indicate the following:</p> <ul style="list-style-type: none"> • Either that there is a field a user has input nothing OR: • Asset name can only contain letters, number and white space • The tag name can only contain letters, number and white space • The asset name is already taken.
Frequency of Use:	Every time a user wants to create a new asset
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	10
Title:	Edit assets
Description:	The user would like to edit one of the assets
Primary Actor:	User
Preconditions:	The user is logged in.
Postconditions:	The asset has been changed which can be seen in the Asset Management page
Main Success Scenario:	The user clicks on Edit. He might change the tag, he might choose a classification, the department and the next review date. After clicking on submit, he is directed to the Asset Management page where he can see the changes made

Extensions:	Error messages can indicate the following: <ul style="list-style-type: none"> • Either that there is a field a user has input nothing OR: • The tag name can only contain letters, number and white space • The review date he chooses has already passed
Frequency of Use:	Every time a user wants to edit an asset
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	11
Title:	Edit assets
Description:	The admin would like to edit one of the assets
Primary Actor:	Admin
Preconditions:	The admin is logged in.
Postconditions:	The asset has been changed which can be seen in the Asset Management page
Main Success Scenario:	The user clicks on Edit. He might change the tag, he might choose a classification, the department, the next review date, and the owner as well. After clicking on submit, he is directed to the Asset Management page where he can see the changes made
Extensions:	Error messages can indicate the following: <ul style="list-style-type: none"> • Either that there is a field a user has input nothing OR: • The tag name can only contain letters, number and white space • The review date he chooses has already passed
Frequency of Use:	Every time an admin wants to edit an asset
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	12
Title:	Delete assets
Description:	The user would like to delete one of the assets
Primary Actor:	User/Admin
Preconditions:	The user is logged in.
Postconditions:	The asset has been deleted which then cannot be seen in the Asset Management page or if there was its asset tag present in the Risk Management page, he can change whether he wants to be directed to the Asset Management or Risk Management page
Main Success Scenario:	The user clicks on Delete. There is a confirmation message where the user needs to confirm the deletion. If the asset tag of that specific asset is not present in Risk management page, then the asset is deleted and cannot be seen in the system anymore. If the asset tag is present then the user is directed to a page which shows the reason and he can choose to be redirected to either Asset Management page or Risk Management page
Extensions:	
Frequency of Use:	Every time a user wants to delete an asset
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	13
Title:	Viewing risk page
Description:	The user would like to view the risks present in the system
Primary Actor:	User
Preconditions:	The user is logged in

Postconditions:	The user is able to identify the risks present in the system.
Main Success Scenario:	The user clicks on Risk management. Here he can see all the risk he owns with its name, description, risk scenario, tag, impact, likelihood, category, the policy that controls the risk and by clicking on it, it can be downloaded, residual risk, target-risk, the asset tag belongs to that risk, the next review date. If there are no risks currently in the system, a message is shown that describes that state.
Extensions:	
Frequency of Use:	Every time a user wants to see the risks present in the system
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	14
Title:	Viewing risk page
Description:	The user would like to view the risks present in the system
Primary Actor:	Admin
Preconditions:	The user is logged in
Postconditions:	The user is able to identify the risks present in the system.
Main Success Scenario:	The user clicks on Risk management. Here he can see every risk with its name, description, risk scenario, tag, impact, likelihood, category, the policy that controls the risk and by clicking on it, it can be downloaded, residual risk, target-risk, the asset tag belongs to that risk, the next review date, and the owner. If there are no risks currently in the system, a message is shown that describes that state.
Extensions:	
Frequency of Use:	Every time a user wants to see the risks present in the system
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	15
Title:	Create a new risk
Description:	The user would like to create new risk
Primary Actor:	User/Admin
Preconditions:	The user is logged in
Postconditions:	The new risk is created which can be seen in the Risk Management page
Main Success Scenario:	The user clicks on Create new risk. He inputs a unique name, a description, risk scenario, a tag, an impact, a likelihood, a residual risk, a target risk, he chooses a policy, an asset tag and the next review date. After clicking on submit, he is directed to the Risk Management page where he can see this new risk
Extensions:	<p>Error messages can indicate the following:</p> <ul style="list-style-type: none"> • Either that there is a field a user has input nothing OR: • Risk name/Description/Risk scenario/Tag can only contain letters, number, and white space • The risk name is already taken.
Frequency of Use:	Every time a user wants to create a new risk
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	16
Title:	Edit risks
Description:	The user would like to edit one of the risks
Primary Actor:	User
Preconditions:	The is logged in and there is at least one risk present in the system
Postconditions:	The risk has been changed which can be seen in the Risk Management page

Main Success Scenario:	The user clicks on Edit. He might change the description, risk scenario, tag, impact, likelihood, residual risk, target-risk, he might choose a different policy, asset tag and next review date. After clicking on submit, he is directed to the Risk Management page where he can see the changes made
Extensions:	Error messages can indicate the following: <ul style="list-style-type: none"> • Either that there is a field a user has input nothing OR: • Description/Risk scenario/Tag can only contain letters, number, and white space • The review date he chooses has already passed
Frequency of Use:	Every time a user wants to edit a risk
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	17
Title:	Edit risks
Description:	The admin would like to edit one of the risks
Primary Actor:	Admin
Preconditions:	The admin is logged in and there is at least one risk present in the system
Postconditions:	The risk has been changed which can be seen in the Risk Management page
Main Success Scenario:	The user clicks on Edit. He might change the description, risk scenario, tag, impact, likelihood, residual risk, target-risk, he might choose a different policy, asset tag, next review date, and the owner as well. After clicking on submit, he is directed to the Risk Management page where he can see the changes made
Extensions:	Error messages can indicate the following: <ul style="list-style-type: none"> • Either that there is a field a user has input nothing OR: • Description/Risk scenario/Tag can only contain letters, number, and white space • The review date he chooses has already passed
Frequency of Use:	Every time a user wants to edit a risk

Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	18
Title:	Delete risks
Description:	The user would like to delete one of the risks
Primary Actor:	User/Admin
Preconditions:	The user is logged in.
Postconditions:	The risk has been deleted which then cannot be seen in the Risk Management page.
Main Success Scenario:	The user clicks on Delete. There is a confirmation message where the user needs to confirm the deletion. If the user clicks on yes, the risk is deleted and cannot be seen in the system anymore.
Extensions:	
Frequency of Use:	Every time a user wants to delete a risk
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	19
Title:	Viewing policy page
Description:	The user would like to view the policies present in the system
Primary Actor:	User
Preconditions:	The user is logged in.
Postconditions:	The user is able to identify the policies present in the system.

Main Success Scenario:	The user clicks on Policy management. Here he can see all the policies he owns with its name and next review date. If there are no policies currently in the system, a message is shown that describes that state.
Extensions:	
Frequency of Use:	Every time a user wants to see the policies present in the system
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	20
Title:	Viewing policy page
Description:	The admin would like to view the policies present in the system
Primary Actor:	Admin
Preconditions:	The admin is logged in
Postconditions:	The user is able to identify the policies present in the system.
Main Success Scenario:	The admin clicks on Policy management. Here he can see every policy with its name, next review date, and the owner. If there are no policies currently in the system, a message is shown that describes that state.
Extensions:	
Frequency of Use:	Every time a user wants to see the policies present in the system
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	21
Title:	Download a new policy
Description:	The user would like to view the policy file uploaded in the system

Primary Actor:	User/Admin
Preconditions:	The admin is logged in
Postconditions:	The user is able to identify the policies present in the system.
Main Success Scenario:	The user clicks on a policy name. A download starts and the user is able to see the policy file that has been uploaded to the system.
Extensions:	
Frequency of Use:	Every time a user wants to download the policies present in the system
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Could-have

ID:	22
Title:	Create a new policy
Description:	The user would like to create a new policy
Primary Actor:	User/Admin
Preconditions:	The user is logged in.
Postconditions:	The new policy is created which can be seen in the Policy Management page and by clicking on the policy name, the download starts
Main Success Scenario:	The user clicks on Create new policy. He uploads a file, inputs a unique name, he chooses the next review date. After clicking on submit, he is directed to the Policy Management page where he can see this new policy
Extensions:	<p>Error messages can indicate the following:</p> <ul style="list-style-type: none"> • Either that there is a field a user has input nothing OR: • Policy name can only contain letters, number and white space • The policy name is already taken. • The uploaded file already exists. • The uploaded file is too large. • That only PDF, Word and Docx files are allowed to upload

Frequency of Use:	Every time a user wants to create a new policy
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	23
Title:	Upload a new policy
Description:	The user would like to create a new policy and a file upload is needed
Primary Actor:	User/Admin
Preconditions:	The user is logged in and a new policy is being added
Postconditions:	The new policy is created which can be seen in the Policy Management page and by clicking on the policy name, the download starts
Main Success Scenario:	The user clicks on add file. He uploads it so that the creation of the policy could be continued.
Extensions:	<p>Error messages can indicate the following:</p> <ul style="list-style-type: none"> • The uploaded file already exists. • The uploaded file is too large. • That only PDF, Word and Docx files are allowed to upload
Frequency of Use:	Every time a user wants to create a new policy
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Could-have

ID:	24
Title:	Edit policies
Description:	The user would like to edit one of the policies
Primary Actor:	User

Preconditions:	The user is logged in
Postconditions:	The policy has been changed which can be seen in the Policy Management page
Main Success Scenario:	The user clicks on Edit. He might choose a different next review date. After clicking on submit, he is directed to the Policy Management page where he can see the changes made
Extensions:	
Frequency of Use:	Every time a user wants to edit a policy
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	25
Title:	Edit policies
Description:	The admin would like to edit one of the policies
Primary Actor:	Admin
Preconditions:	The admin is logged in.
Postconditions:	The policy has been changed which can be seen in the Policy Management page
Main Success Scenario:	The admin clicks on Edit. He might choose a different next review date and an owner as well. After clicking on submit, he is directed to the Policy Management page where he can see the changes made
Extensions:	
Frequency of Use:	Every time an admin wants to edit a policy
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	26
Title:	Delete policies
Description:	The user would like to delete one of the policies
Primary Actor:	User/Admin
Preconditions:	The user is logged in
Postconditions:	The policy has been deleted which then cannot be seen in the Policy Management page or if this policy is present in the Risk Management page, he can change whether he wants to be directed to the Policy Management or Risk Management page
Main Success Scenario:	The user clicks on Delete. There is a confirmation message where the user needs to confirm the deletion. If the policy is not present in Risk management page, then the policy is deleted and cannot be seen in the system anymore. If the policy is present then the user is directed to a page which shows the reason and he can choose to be redirected to either Policy Management page or Risk Management page
Extensions:	
Frequency of Use:	Every time a user wants to delete a policy
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	27
Title:	Viewing user management page
Description:	The admin would like to view the users present in the system
Primary Actor:	Admin
Preconditions:	The admin is logged in
Postconditions:	The admin is able to identify the users present in the system.
Main Success Scenario:	The admin clicks on User management. Here he can see all the users with their name, email address, user type and whether they have left or not. If there are no other users currently in the system, a message is shown that describes that state.

Extensions:	
Frequency of Use:	Every time an admin wants to see the users present in the system
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	28
Title:	Edit users
Description:	The admin would like to edit one of the users
Primary Actor:	Admin
Preconditions:	The admin is logged in
Postconditions:	The user has been changed which can be seen in the User Management page (if the user has left column has been changed to 'yes' the name of the user is changed to 'Deleted' as well).
Main Success Scenario:	The admin clicks on Edit. He might change the user type or whether the user has left. After clicking on submit, he is directed to the User Management page where he can see the changes made
Extensions:	
Frequency of Use:	Every time an admin wants to edit a user
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	29
Title:	Delete users
Description:	The admin would like to delete one of the users
Primary Actor:	Admin
Preconditions:	The admin is logged in and there is at least one other user present in the system whose 'user has left' column value is 'yes'

Postconditions:	The user has been deleted which then cannot be seen in the User Management page
Main Success Scenario:	The admin clicks on Delete. There is a confirmation message where the admin needs to confirm the deletion. After that, the user is deleted and cannot be seen in the system anymore
Extensions:	
Frequency of Use:	Every time an admin wants to delete a user
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

ID:	30
Title:	Viewing event log page
Description:	The admin would like to view the event log
Primary Actor:	Admin
Preconditions:	The admin is logged in
Postconditions:	The admin is able to identify the changes made in the system
Main Success Scenario:	The admin clicks on the Event log. Here he can see all the changes made in the system with the name of that entity, where it belongs, what the action was and a timestamp
Extensions:	
Frequency of Use:	Every time an admin wants to see the changes made in the system
Status:	Developed
Owner:	Szabolcs Magyar
Priority:	Must-have

Usability test

Welcome! My name is Szabolcs Magyar and I'm an Information Management MSc student. My thesis topic is Security Data Governance System and I have developed a system for this.

I was informed that you have already worked with similar systems that is why I asked you to participate in the study. In this study, you will be asked to perform a number of tasks. These tasks will be presented on a paper once you start using the system. The evaluation will be anonymous and no personal data will be collected. While carrying out the tasks feel free to ask for clarification if needed, but I will be neutral throughout the test. Keep in mind we are evaluating the software/feature/product, not you, so there are no wrong answers. Try to complete the tasks as if you were doing this for real. Spend as little or as much time as you normally would doing these tasks. It is OK if you cannot complete each task, and we may not get to every task.

In the second part of the study, there is 6 question about your opinion on the system. You will find the questions on the paper once you start the study.

If you have any questions or concerns about this study then please free to speak to my supervisor, Sotirios Terzis (sotirios.terzis@strath.ac.uk, tel.: +44 141 548 3839) or the Departmental Ethics Committee (ethics@cis.strath.ac.uk).

Data Management Plan

On 19.10, two months after the research is to be submitted in physical and electronic copies for the Department, the MS Word that contains all the participants' answers will be deleted.

In this usability evaluation:

- You will be asked to perform certain tasks on a computer.
- You will be asked to answer some questions and freely comment on the system.

Participation in this usability study is voluntary. All information will remain strictly confidential. The descriptions and findings may be used to help improve the application. However, at no time will your name or any other identification is used. You can withdraw your consent to the experiment and stop participation at any time. If you have any questions after today, please contact Szabolcs Magyar (szabolcs.magyar@uni.strath.ac.uk) or Sotirios Terzis (sotirios.terzis@strath.ac.uk, tel.: +44 141 548 3839).

Before the study starts please confirm that you have read and understood the information on this form and had all of my questions answered.

Thank you for your participation in the study.

In the first part of the study I would like you to do the following tasks:

1. Log in to the system with these credentials:

- Username: pxtrva
- Password: N1ckn@me

2. Go to the asset page. Here you can find some default assets. Create this asset:

- Asset name: Lenovo z5
- Classification: Medium
- Tag: Laptop
- Department: Accounting and Finance
- Review date: 20/08/2019

3. Change asset classification to 'Low'.

4. Go to the policy page. Here you can find some default policies. Create this policy:

- Please select and upload "Laptop usage policy" from Desktop
- Name: Laptop Usage Policy
- Review date: 12/08/2019

5. Go to the risks page. Here you can find some default risks. Create this risk:

- Name: Laptop theft
- Description: Laptops are stolen
- Risk scenario: Incident
- Tag: Theft
- Impact: 5
- Likelihood: 2
- Policy: Laptop Usage Policy
- Residual risk: 5
- Target risk: 4
- Review date: 25/08/2019

6. Please click on the policy. What did you find after clicking on it?

7. Please go to 'Dashboard'. Can you find the value of 'Medium' category risks? How many percentages of risks are compliant? How many assets are not compliant?

8. Log out

In the second part, please answer these questions:

1. Which parts of the system did you find easy and which complex to use?
2. What do you think of the UI of the system in terms of consistency and appeal?
3. Are there any parts of the system you found cumbersome or unclear?
4. How useful/not useful you think the system will be to you?

5. What do you think, are there any additional functions that could have been implemented? If yes, what are these?
6. Do you have any further comments about the system?