FACULTY OF SCIENCE, ENGINEERING AND COMPUTING

School of *Computer Science & Mathematics*

BSc DEGREE
IN
Software Engineering

PROJECT REPORT

Tony Mace
K1451757

Project Title: School Project management software

Project Type: Build

Date: 23/04/2017

Supervisor: Graeme Jones

Declaration

I have read and understood the University regulations on plagiarism and I understand the meaning of the word plagiarism. I declare that this report is entirely my own work. Any other sources are duly acknowledged and referenced according to the requirements of the School of Computer Science and Mathematics. All verbatim citations are indicated by double quotation marks ("…"). Neither in part nor in its entirety have I made use of another student's work and pretended that it is my own. I have not asked anybody to contribute to this project in the form of code, text or drawings. I did not allow and will not allow anyone to copy my work with the intention of presenting it as their own work.

Date: 20/04/2018

Acknowledgement

I would like to take this opportunity to thank Professor Graeme Jones for his constant support and motivation, as a client and supervisor. I could accomplish the project on time by following his weekly advices. He was fully dedicated and always found the time to listen to me and answer my questions, with regular face to face meetings, emails and even late at night on skype. Professor Graeme Jones could provide a great guidance, pushed me to get the best out of me to achieve my project.

Thank you to the lecturers and Beryl Jones, who were also present to support me and dedicated their time to give essential advices through lectures but also during their personal time.

And finally, I would like to thank my partner Ewa Przybyszewska who was always here for me and encourage me all along the way and would dedicate herself to free up the necessary time for me to work on my project. I would not be able to do my project without her.

Table of Contents

Abstract 1
1 Introduction 2
1.1 Problems being discussed 2
1.2 Proposed Solution 2
1.3 Aim of the project 2
1.4 Objectives 2
1.5 Potential Issues 3
1.6 Structure 3
2 Literature review 4
2.1 What is collaboration 4
2.2 Analysis of Online Collaboration Tools in Education 4
2.3 The impact of online collaboration tools 6
2.4 Conclusion 8
3 Analysis and Software Methodology 9
3.1 Software Development Methodology 9
3.2 Planning 10
3.3 Stakeholder Analysis 11
3.4 Risk Analysis 14
3.5 Legal, Ethical, Societal and Security issues 15
3.6 Conclusion 16
4 Requirements Analysis 17
4.1 Requirements Elicitation and Validation 17
4.2 Requirement Management 20
4.3 Conclusion 22
5 Design 23
5.1 Database design 23
5.2 Modelling system interactions 28
5.3 Interface Design 32
5.4 Conclusion 37
6 Implementation 39
6.1 Tools, Technologies and Architecture 39
6.2 Database Implementation 47
6.3 Back-end implementation 50
6.4 Front-end Implementation 54
6.5 Security Concerns 68

6.6 Conclusion 69
7 Testing 71
7.1 Functional testing 71
7.2 Non-functional testing 72
7.3 Usability testing 73
7.4 Unit testing – Angular 5 74
7.5 REST API Testing 77
7.6 Conclusion 80
8 Critical Review 81
8.1 Introduction 81
8.2 Achievements 81
8.3 Problems Encountered and Limitations 81
8.4 Future Work 82
8.5 Legal, Social and Ethical Aspects 82
8.6 Conclusion 83
9 References 84
10 APPENDIX A - Interviews 88
11 APPENDIX B - User stories 89
12 APPENDIX C - Use cases 91
13 APPENDIX D - Use case diagrams 95
14 APPENDIX E – Non-functional requirements 99
15 APPENDIX F - MoSCoW prioritisation 100
16 APPENDIX G - Data dictionary 104
17 APPENDIX H - Activity Diagrams 108
18 APPENDIX I - Sequence Diagrams 112
19 APPENDIX J - Sketches 116
20 APPENDIX K - Wireframes 118
21 APPENDIX L – Complex queries 131
22 APPENDIX M - Slim Setup 139
23 APPENDIX N – Main reasons not to use MySQL 140
24 APPENDIX O - Test cases functional testing 141
25 APPENDIX P - Test cases non-functional testing 145
26 APPENDIX Q - REST API testing 146

Abstract

Since collaboration is one of the most important aspect of modern education, it quickly became apparent that lectures and students need to be supported by an appropriate tool to work as a team and execute their projects successfully.

The client does not have any solution in place to enable the users to either assign the project or work on the assigned projects which in turns affects teamwork and collaboration.

The development of a web-based project management tool will allow the users to plan, organise, schedule, manage and delegate tasks as well as estimate and keep track of deadlines, report issues, control changes and share documentation. This centralised solution will be responsive and accessible on any devices, featuring an intuitive and user-friendly web interface.

1 Introduction

Kingston University involves students in many group projects during their entire studies to gain a genuine experience working as a team and improve their skills to prepare for their future. Groups can be formed by lecturers who may assign distinct roles to each student or students. To stay on track, students should be aware of changes made within the project and plan and manage their tasks. Lecturers need to watch the progress of the students and gather the deliverables.

1.1 Problems being discussed

The current learning management system provides basic functionalities such as allowing users to view the modules, announcements, discussions, assessments, grades and calendar. As such, the users do not have a platform accessible from anywhere at any time to work efficiently and collaboratively on their group projects to achieve a common goal. As a result, students who are unable to attend group catch-ups, but want to contribute from home, find it difficult to commit to the group and stay up to date. The students are unable to correspond and exchange contents therefore they may lose time on supplementary meetings to keep everyone up to date. Currently, lecturers do not have clear visibility of the list of students belonging to each project group. They may use simple excel spreadsheets to keep record of the groups and lose time chasing students to ensure they are all affiliated to a group. They do not have clear visibility of the progress made by each student and as a result, they may be unable to give the right marks for each student. Users cannot assign tasks to a group or specific member of the group so they may not know who should oversee each duty. These issues can affect students' motivation, conflicts may arise and cause some complications within the project groups.

1.2 Proposed Solution

The goal is to create a system that will solve the issues with the following solutions:

- A centralised web-based system where users can collaborate and contribute to the project anytime, anywhere.
- A task management system that allows users to create, distribute and delegate tasks, view completed and ongoing tasks, comment on projects, tasks and subtasks, keep track of the progress and give a clear view of the groups, and users who are not in a group.
- An email notification system that will keep the students up to date on all changes made within a project.
- A content sharing system to ease the exchange of content across the group.

1.3 Aim of the project

The aim of this project is to develop responsive and collaborative project management system to support lecturers and students working on group or individual projects.

1.4 Objectives

The objectives of the project are:

- Capture and implement the user requirements.
- Provide a responsive user friendly, engaging and interchangeable interface to ensure users will have a great experience with the system.
- Provide a secure and efficient database that must accommodate a large amount of data for the students and lecturers of Kingston University.
- Provide a REST API (Representation State Transfer – Application Programming Interface) to retrieve and insert the data for the system.

1.5 Potential Issues

The project needs to be developed over a short period of time. Only ten hours every week over seven months can be dedicated to implement the system and generate a full report for the client. It is essential to consider the right tools and technologies to develop the product, ensure that requirements are fulfilled and the client is satisfied. Angular 5 will be used to implement the frontend. The documentation available online is not clearly detailed and may require extensive research. A lot of data will need to be generated to test efficiently the database and ensure it can support more than 18000 users. The software used to support the implementation of the product are open source due to budgetary restrictions. It should not have a significant impact on the system nevertheless more sophisticated paid solution are available. The lack of experience in a large system implementation should also be considered as it would have a major effect on the result.

1.6 Structure

The report is structured with the main chapters as follow:

- Chapter 2 – literature review that focuses on collaboration within education.
- Chapter 3 – software methodology chosen to manage and develop the software, as well as planning needed to deliver the product on time.
- Chapter 4 – requirement analysis and possible techniques used to elicit and validate the requirements.
- Chapter 5 – design and interaction of the system, including the database and user interface, using sequence diagrams, activity diagrams, sketches and wireframes.
- Chapter 6 – architecture of the system and steps that were carried out to implement it.
- Chapter 7 – tests completed to verify and validate the product.
- Chapter 8 – critical review of the tasks that have been achieved to complete the project.

2 Literature review

2.1 What is collaboration

According to Schrage (1990) collaboration is the process in which two or more individuals with complementary skills interact to create a shared understanding that none had previously possessed or could have come to on their own.

In education on the other hand, collaboration is described as an opportunity for school renewal (Fishbaugh 1997), a chance to involve many individuals in complex educational problem (Council for the Accreditation of Educator Preparation 2017) and an expectation to "promote the most effective teaching possible for the greatest number of students" (Pugach and Johnson 1995, 178).

Collaboration helps create innovative ideas and unique solutions that arise because of the interactions of different perspectives, experiences and knowledge, where "no individual's point of view dominates, authority for decisions and actions resides in the group, and work products reflect a blending of all participants" contributions (JohnSteiner, Weber, and Minnis 1998, 776).

2.2 Analysis of Online Collaboration Tools in Education

2.2.1 Technology and Collaborative Learning

The interest in technology supported collaborative learning has been present for many years and it represents a variety of trends such as the development of new tools to support collaboration (Johnson & Johnson, 1996), the rise of constructivist-based approaches to teaching and learning (Kirschner, Martens, & Strijbos, 2004), and the necessity to develop a more effective and engaging learning atmosphere.

Lockyer et al (2001) and Mason & Romiszowski (1996) claim that their research supports the idea that online environments are as compelling as, or more so, than actual face to face classes and students show higher levels of learning in an online environment when compared to face-to-face groups.

The results of various studies analysing collaborative learning have found that computer-based collaborative learning produces a greater quantity and quality of daily accomplishments, better understanding of information, and a higher success rate in problem solving than computer-supported individualistic learning (Resta and Laferrière, 2007).

2.2.2 Online Collaboration Tools

There are many online collaboration tools available on the market, but taking into consideration the needs of the client, Kingston University, the focus will be on solutions available for students and lecturers or teachers.

Online collaboration tools are web-based solutions that allow teachers and students to carry out a wide range of tasks such as: interactive discussions, online collaboration activities, sharing and accessing electronic learning resources and many others. According to A. Herrington and J. Harrington (2006) the opportunity for individuals to collaborate and research various perspectives is a crucial part of the university courses' design, especially for students who are learning or working at a distance. The following are the most popular online collaboration tools:

Google Docs

Google Docs a part of Google Drive. It is one of the most popular and free online word processor available. It allows users to easily create, upload, save, share, and collaborate on documents and it offers quite a remarkable selection of formatting options.

Google Hangouts (GH)

GH is a communication platform that allows users to engage in different communication mediums such as text, voice or video chats. This application is effective in one to one as well as group communications.

Padlet

Padlet is a great tool for mind-mapping services, online-pin boards, sharing digital content and improving students' engagement. This software is appropriate for business as well as school use as it provides a controlled learning environment that helps tutors to enhance the writing process.

TodaysMeet

TodaysMeet is a backchannel chat platform for tutors and students who can join fast and easy to use chat rooms.

Debate Graph

A web-based platform designed for visualizing and sharing arguments and ideas in a collaborative manner. It allows students to participate in group debates and structure collaborative brainstorms on any complex topic. The aim of this online platform is to enable and encourage students to contribute to a discussion, share expertise and therefore allow students working in groups to reach better conclusions or arrive at a consensus in a conflict situation.

Twiddla

Twiddla is one of the most well-known web-based collaboration tool for teachers, referred to as a "web-based meeting playground." It allows users to collaborate in real-time on a virtual whiteboard. Users can choose to upload images, text, websites, or drawings.

Wikispace Classroom

Wikispace is a social writing platform. It provides a safe and private environment for discussions, collaboratively editing pages, and completing group projects. It is equipped in pre-built templates for a variety of projects, but it also allows students to work from a blank slate. What is more, Wikispace allows the teachers to measure student contribution and engagement in real-time and immediately communicate feedback.

Talkboard

Talkboard is an "indispensable tool for illustrating collaborative ideas on the fly" according to Brienne Walsh at Forbes (2013). Users can visually express and build on ideas using iPads. This tool can be especially compelling for language learners, autistic students, and visual learners.

2.2.3 Online Collaboration Tools Analysis

Online collaboration tools are very popular and easy to access nowadays. However, the vast number of the existing technologies makes it hard to find one that meets specific requirements.

Communication Tools

Google Hangouts (GH) TodaysMeet and Twiddla are communication tools. GH and Twiddla allow users to make voice and video calls and GH has an instant message option, whereas TodaysMeet offers chat room and discussion forum communication. Nevertheless, TodaysMeet has other functions such as online seminars that Google Hangouts and Twiddla do not offer.

Content Sharing Tools

Google Docs, Twiddla and Padlet provide users with the possibility of sharing content but only Google Docs allows users to edit documents in real time. It also supports many more file types. Twiddla, on the other hand, has options for online meetings and discussions as well as voice chats while Padlet is great for mind-mapping.

Brainstorming Tools

Talkboard and Debate Graph are designed for visualizing and sharing ideas. Talkboard allows users to use various media files, such as photos and voice files, to create boards and it's more appropriate for younger users while Debate Graph focuses on mapping ideas and issues by building and editing maps of complex debates that accurately reflect all sides of the discussion from a neutral point of view. Debate Graph is more appropriate for higher education.

Project/Classroom Management Tools

Wikispaces and Padlet are both great for project/classroom management. Both tools provide the option for working on group projects and managing classrooms by providing feedback to students and creating collaborative brainstorm sessions and discussions. On the other hand, Wikispaces provides additional options such as social networking and remote learning whereas Padlet allows users to share links and initiate book review sessions.

2.3 The impact of online collaboration tools

2.3.1 What is collaborative learning?

Roschelle's & Teasley's (1995) defines collaborative learning as: "...a coordinated, synchronous activity that is the result of a continued attempt to construct and maintain a shared conception of a problem" (p. 70). Bruffee (1993), on the other hand, suggests that collaborative learning is a re-acculturative action that allows students to become members of knowledge societies. In this case collaborative learning is based on the belief that knowledge is a social design and its activities are most often based on four rules:

- The learner is the focus of instruction.
- Interaction and "doing" are of prime importance.
- Group work is an important method of learning.
- Structured approaches to developing solutions to real-world problems should be incorporated into learning.

Collaborative learning can take place in one to one situations or in larger groups. Peer learning includes students working in pairs or small groups to analyse concepts or look for solutions to problems. Group work or collaborative learning can take many different forms for example: group work activities at university or in a classroom or more involved group projects that stretch over the course of a semester.

2.3.2 Impact of collaborative learning

Research suggests that educational exercises that are active, social, contextual, engaging, and student-owned led to deeper learning. The benefits of collaborative learning include:

- Development of higher-level thinking, oral communication, self-management, and leadership skills.
- Promotion of student-faculty interaction.
- Increase in student retention, self-esteem, and responsibility.
- Exposure to and an increase in understanding of diverse perspectives.
- Preparation for real life social and employment situations.

2.3.3 Advantages of Collaborative Learning (CL)

According to study conducted by Gokhale (1995) collaborative learning supports the development of critical thinking through activities such as discussion, clarification of ideas, and evaluation of others' ideas.

There are other numerous benefits of collaborative learning and Pantiz (1999) and Johnson (1998) described over 50 of them. A great way of arranging these benefits is to put them into various categories such as:

Social benefits

- CL influences and encourages open-mildness among students and teachers
- CL creates a positive environment for designing and practicing cooperation

Psychological benefits

- Student-related instruction boosts students' confidence;
  - It helps to reduce anxiety and develop a positive approach to teachers.

Academic benefits

- CL develops critical thinking skills
- It helps to develop useful problem-solving techniques
- Collaborative learning has been shown to be especially helpful in motivating students

Alternate student and teacher assessment techniques

- Collaborative teaching methods promote a variety of assessments.

2.3.4 Disadvantages of Collaborative Learning

Barker et al. (Barker, Garvin-Doxas, & Jackson, 2002) conducted a research to study the nature of the learning environment in computer science classrooms. Their findings confirmed that approximately 80% of students identified certain disadvantages of working in groups. For instance, some students stated working in groups can be distractive and it’s more difficult to focus on concept formation or difficult problems. They also mention that it is hard to choose your own pace when working with others. Working with other students means that at times the work is not time efficient as everyone needs to agree to solutions. This may also lead to some conflicts.

2.4 Conclusion

According to Okuogume and Jäminki (2011) the development of more robust frameworks, pedagogies and tools for collaboration across the Higher Education sector can help to produce more employable students who are better equipped to deal with the contemporary industrial setting. Collaboration encourages knowledge sharing, communication and exchange of ideas and at the same time it allows students to learn from each other.

To conclude, collaboration and its related technologies have a very positive impact on education and they provide a better learning experience for students by providing them with right tools and learning environments. As for teachers, collaboration and technology help them to keep students interested and engaged.

3 Analysis and Software Methodology

This chapter focuses on the importance of selecting the appropriate software development methodology to structure, plan and control the process of developing the system. Tools such as SWOT (Strengths, Weaknesses, Opportunities, Threats), Power/Interest grid and risk analysis will be used to ensure the project can be conducted, determine the main stakeholders and identify the potential problems that could undermine the business.

3.1 Software Development Methodology

Following extensive research, Agile, DSDM/Atern (Dynamic system development methodology) has been chosen to manage and develop the project for the following reasons:

- Some requirements may evolve during the project and it will be crucial to accommodate any changes to avoid delays.
- Client will be heavily involved during the project, with weekly individual and group meetings to track progress, receive feedback and discuss existing and new requirements in detail.
- Frequent and timely delivery of working software will ensure that client's expectations are satisfied.
- Continuous attention to detail and design is essential during the development of the project as it will help the delivery of a perfect product without compromising the quality. (see Figure 3.1).

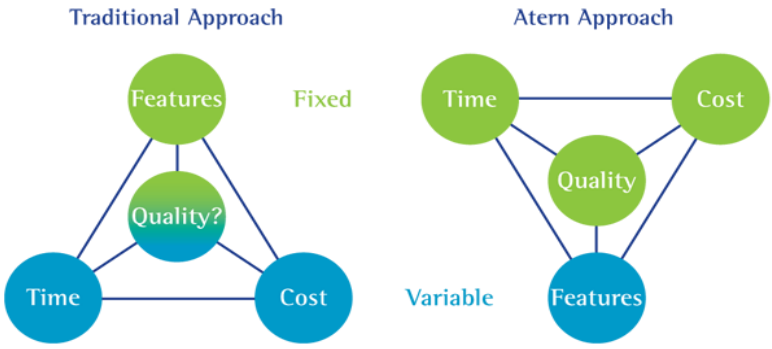


Figure 3.1 – Traditional Approach versus Atern Approach

The following methodologies will not be used for the project:

Project Management Methodology:

Prince2 is not an iterative development approach, it does not offer the level of flexibility that modern management methodologies provide and it includes heavy documentation.

Software development Methodology:

Waterfall is too rigid and it focuses on up front requirements capture and design. As such it is very hard and costly to make changes during the project and by extension to make any improvements.

Agile scrum was disregarded because it is only suitable for a team effort. Also, it is hard to ensure the quality of product until it has been thoroughly tested.

Extreme programming (XP) focuses mainly on the code rather than design. XP programming also lacks documentation and therefore the quality of the product may be compromised and bugs may occur in the future.

3.2 Planning

3.2.1 Gantt Chart

Gantt chart provides a clear understanding of the major milestones within the project including the key deadlines.

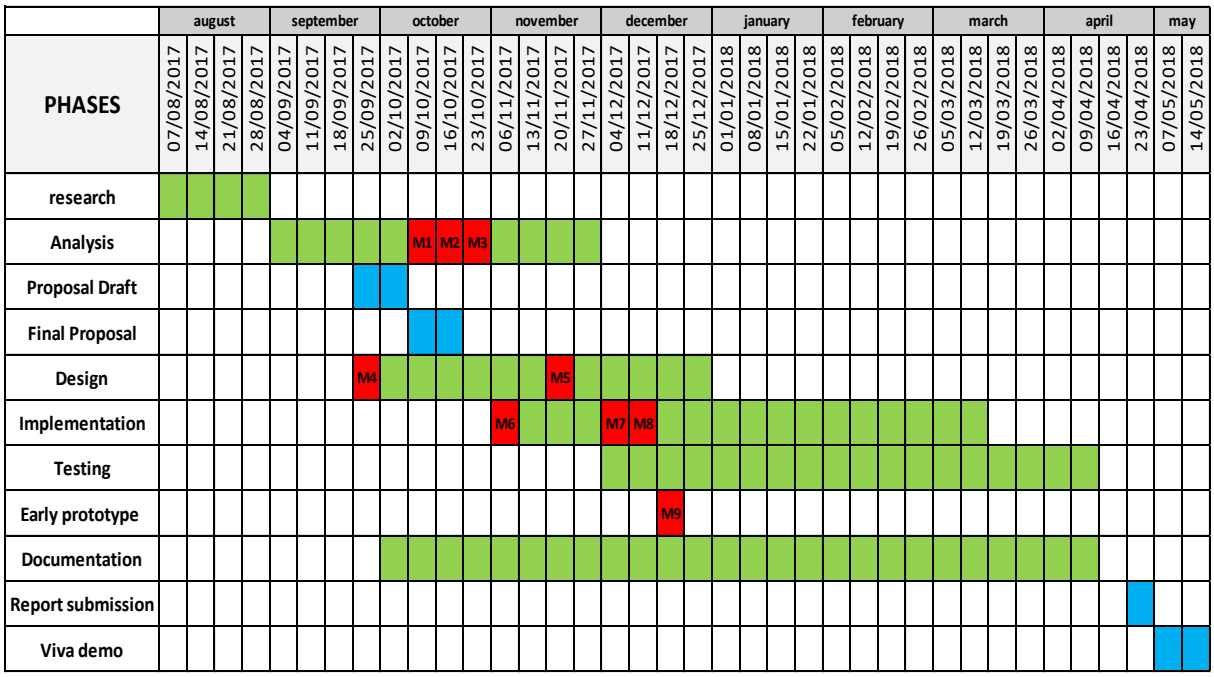


Figure 3.2 – Gantt Chart

3.2.2 List of the Milestones

- Research
- Analysis

  M1 SWOT analysis
  M2 Stakeholder Analysis, Power/interest
  M3 MoSCoW

- Design

  M4 ERD diagram, data dictionary
  M5 Wireframes, sketches

- Implementation

  M6 Database
  M7 Front-end
  M8 Back-end
  M9 Prototype

- Testing
- Documentation

3.2.3 Deliverables

See below the list of deliverables that have been already established with the client in Figure 3.3. Weekly group meetings are planned to discuss current and future deliverables.

29/09/2017	Documentation Methodology	Proposal Appendices, Initial ERD
06/10/2017	Documentation	Complete draft proposal
13/10/2017	Methodology Documentation	SWOT, **Project Proposal**
20/10/2017	Implementation Methodology	Install DB software, Stakeholder analysis
20/10/2017	Documentation	TOC Requirements Chapter
27/10/2017	Implementation	Establish PHP connection to database, Data and DB Creation
27/10/2017	Methodology	Use Cases, User Stories
03/11/2017	Implementation	Simple SQL, PHP report
03/11/2017	Methodology	Requirements / MoSCoW
03/11/2017	Research	Identify Review Topic
10/11/2017	Implementation Documentation	Simple SQL insert, Draft Requirements Chapter
10/11/2017	Research	Bootstrap/Mobile API Interface
17/11/2017	Documentation, Methodology, Implementation	TOC Design Chapter, Sketches/Wireframes, Inner Join report
24/11/2017	Documentation	Requirements Chapter
24/11/2017	Methodology	Navigation/Sequence diagrams
01/12/2017	Research	Identify SQL queries from requirements
01/12/2017	Implementation	Implement mock-ups of key webpages, Key report 1
08/12/2017	Research	Plan for prototype demonstration
08/12/2017	Implementation	Key form 1
15/12/2017	Presentation	**Early Prototype demonstration**
12/01/2018	Methodology	Revised ERD, Data Dictionary
12/01/2018	Documentation, Research	Draft Design Chapter, List of key reference material
19/01/2018	Implementation, Documentation	Key report 2, Design Chapter
19/01/2018	Research, Implementation, Documentation	Test Strategy, Key form 2, TOC Review Chapter
26/01/2018	Research, Implementation	Identify most complex SQL, Key report 3
26/01/2018	Documentation	Draft Review Chapter
02/02/2018	Implementation	Key form 3
09/02/2018	Documentation	TOC Implementation Chapter, Review Chapter
16/02/2018	Implementation, Documentation	Key report 4, Technologies, Architecture Section
23/02/2018	Research, Documentation, Implementation	Identify viva story, Second Major Section, Key form 4
02/03/2018	Documentation, Implementation	Draft Implementation Chapter and Key report 5
09/03/2018	Documentation	Testing Chapter
09/03/2018	Implementation	Key form 5
16/03/2018	Research	Identify viva-oriented functionality
16/03/2018	Documentation	Implementation Chapter
23/03/2018	Documentation	Introduction, Conclusion
30/03/2018	Documentation	Appendices, references
06/04/2018	Documentation	Complete draft report
13/04/2018	Implementation	Last functionality implemented
23/04/2018	Documentation	**Report Submission**

Figure 3.3 – Deliverables

3.3 Stakeholder Analysis

3.3.1 Identification of stakeholders and Benefits

Stakeholders having internal or external interest in the projects are listed below. The diagram "Figure 1" beside represents the generic Stakeholder categories.

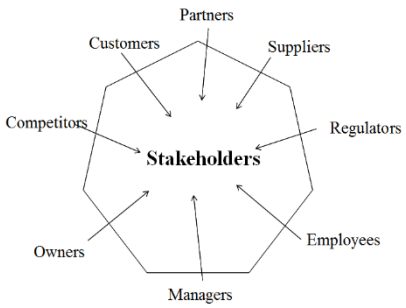


Figure 3.4 – Stakeholders Categories

Kingston University

The client is represented by Graeme Jones, Head of the Computer Science Department. He will attend all the meetings to monitor the progress and evolution of the project and provide continuous feedback.

Benefits

The School Project Management Software (SPMS) will benefit Kingston University by improving the engagement amongst the students which will result in more desirable results and increased grades.

Lecturers

They are involved with many groups of students during the year as they teach one or few modules at the time. They are directly vested in the project and they provide ongoing feedback.

Benefits

The lecturers will be able to monitor the students' participation in the project and its progress. It will allow them to work more efficiently and create and assign a single task or more complex project to number of students with only few clicks.

Students

Students have been identified as the primary users of the SPMS as they will use it daily. It is important that they get the opportunity to test the software during the development stages and provide their feedback.

Benefits

They will be able to collaborate and work efficiently on mutual and individual projects regardless of the time and location, access this responsive application from any devices as well as share contents and keep track of the project deadlines. This solution will allow students to always be on top of their projects and therefore improve their grades.

Project Leader and software developer

Project leader who is managing and developing the SPMS will be attending all individual and group meetings to present the continuous progress of the software and discuss current and eventual new requirements.

Benefits

The following skills will be improved as a result of the completion of the project:

- Project management and decision making
- Time and deadline management
- Organisational skills
- Research skills
- Development skills
- Programming skills

These will allow them to gain very valuable experience and skillset that can be used in any future projects.

Government

The government provides funds for Kingston University. They will not be present during any of the meeting.

Benefits

The main benefit for the government is the improvement of students' education by supporting them with the appropriate tools.

3.3.2 Stakeholder Prioritization Power/interest

The power interest matrix shown in Figure 3.5 is the simple tool that assesses the interests of the stakeholders and their impact. The introduction of the SPMS will provide students with a new solution to support their education and improve their grades. Lecturers will be able to closely monitor the progress of all students working on multiple projects. The government will have the opportunity to improve the education of all the students at Kingston University. Kington University will not only be making sure that all their students are supported with an efficient tool to collaborate on group project, but also improve their ranking and reputation across all Universities. For the project leader and software developer it is a wonderful opportunity to get experience and develop their skills.

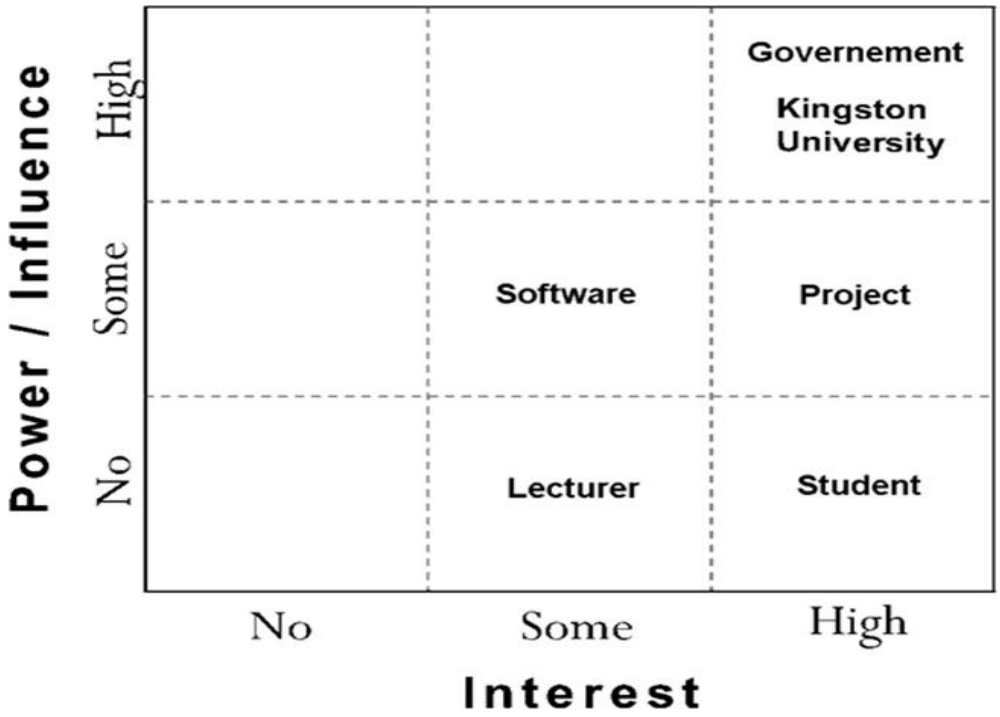


Figure 3.5 – Power/interest Matrix

3.3.3 Stakeholder Management strategies SWOT

SWOT analysis is a strategic planning method to help identify the strengths, weaknesses, opportunities, and threats of an organisation to make sure that the project can go ahead.

	Positives	Negatives
Internal Current	Strengths • Experience in a start-up company • working as a quality assurance • Previous experience in a business project • Proficient in Web Programming languages (PHP, HTML, CSS, MySQL) • Experience using multiple project management software • Real inside with the client • Low cost to develop the software • Feedbacks from professionals	Weaknesses • Time constraints with multiple jobs • Lack of experience in angular • Deadlines and constant deliverables • Lack of time to complete the project • Only one developer working on the project
External future	Opportunities • Project management software is an emerging market • Many organisations are currently using software management system • Tracking progress of users become important • Current solution for the client do not support project management software • Possibility to extend the product to other clients • The government is always willing to improve the education of the students with modern solution	Threats • Project management software competitors • Constant evolution of the market • Hackers

Figure 3.6 – Swot analysis

Following completion of the Swot Analysis as shown in Figure 3.6, the project does not face any real constraints. Weaknesses are mainly focused on timing issues which can be eliminated by good organisational skills. The threats will be considered during the evolution of the product, but they would not have a direct negative impact. It is safe to say that the project can be conducted.

3.4 Risk Analysis

Risk Analysis as shown in Figure 3.7, is a process to identify potential problems that could undermine the business. It is important to anticipate them before starting the project. It will help decide whether the project should move forward. It will also support the decision on the equipment and technology to use.

Risk	Likelihood	Impact	Migration
Database corruption	Low	High	Regular back-up
Client dissatisfied with final product	Low	High	Regular meetings to ensure that all the requirements are met
Not meeting the final deadline	Low	High	Ensure that all the millstones are accomplished on time
Prolong web-hosting out of service	Low	High	To have a second server locally
Shortage of skills	Low	Medium	Conduct extensive research and ask for support
Database overloaded	Low	High	Find appropriate database to replace existing one
Budget for web-hosting	Low	medium	Find the best web hosting solution within the budget

Figure 3.7 – Risk Analysis

3.5 Legal, Ethical, Societal and Security issues

The main issues that need to be addressed area:

Legal issues

- The law of copyright and the relevant regulations that are applicable in relation to databases and the provisions of the 1996 EC Council Directive on the legal protection of databases called "database right".
- The rules under Data Protection Act, 1998 that apply in relation to storing users' personal information.

Ethical issues

- Copyrighted or credited source code may be published freely on public website but the author will need to be credited.
- Reverse engineering code
- Not addressing known bugs

Social issues

- Building an application that cannot be used by certain group of people due to physical, cultural or emotional reasons. For example, teaching users with vision impairment on how to use the software.

Security issues

- Computer malware such as viruses.
- Cyber safety.
- Huge amounts of information (which may be unsupported, unverifiable, misleading or incorrect).

3.6 Conclusion

Agile DSDM/Atern development methodology has been chosen over other methodology as it allows to accommodate constant changes during the development of the product and always involve the client. The plan has been established with weekly iterations and regular meetings with the client to get constant feedback. The stakeholders have now been identified and the risks have been considered to ensure the client is pleased and the project can be conducted.

4 Requirements Analysis

In this chapter, all the requirements will be elicited and validated, and prioritised by using various tools and getting constant feedback from the client.

4.1 Requirements Elicitation and Validation

Requirements elicitation process is one of the most critical process for the success of the product. It is intended to gain knowledge about the user's requirement or need. Requirement gathering techniques may be different from one project to another. Here is a list of the main techniques:

- Document Analysis
- Interviewing
- Observation
- Focus group
- Brainstorming
- Requirements Workshop
- Surveys
- Prototyping
- User stories
- Use cases
- Use case diagrams

The project will be focusing on user stories, use cases, use case diagrams and interviews.

4.1.1 Interviews

One-to-one interviews will be conducted to gather the requirements as they are a primary source of requirements. It is important to know who the stakeholders are and what are their needs at this stage. The interviews will contain open-ended questions as well as closed-ended questions and a combination of the structured and semi-structured interviews will be used to conduct the interviews.

Interview with stakeholders

It is important to focus on students and lecturers as they are the main users of the system. Interviewing the Lecturers and Students will provide important feedbacks and essential requirements.

It is also essential to conduct several interviews with the client. The questions in the Appendix A are a summary of the main questions asked during face to face interviews to clarify and understand the requirements.

Interview Summary

Several interviews were conducted with the stakeholders to have a clear understanding of the user requirements and consider current solution already in place.

The client wants team leader to validate tasks once they have been created by team member and they should be responsible for inviting all the new team member and assigning the tasks. Roles within the team should also be interchangeable. A task should only be assign to a single user. Content should be shared within tasks or on the project section itself. The client is mainly concerned about notifying users when main changes are made on tasks and projects. This is not a priority at the moment. The interface could be customisable in the future but the client specified that it is not essential.

At present, students do not have the tools to work on a project and communication is often difficult. They can only create groups and share contents and they are not happy with current

solution. Students only meet during workshops and spend most of the time sharing what has been done individually instead of focusing on the next task. Students would find it useful to be able to collaborate over a project and assign tasks to different members, so they can easily track the progress of each member and be updated in real time with email notification.

The lecturers do not have a solution to keep track of all the groups within their courses and modules, and may be using simple excel spread sheet to do so. Some students may not even be part of a group until late during group project assignment. Lecturers do not have the possibility to track the progress of the groups, they can view all the members of the course and modules and they would be grateful if they could view and manage all the groups within each module and course.

4.1.2 User stories

A user story is a brief description of one or more system features that user will use to reach a goal. It describes a specific need that user has when using the system and it is written in a simple language easily understandable by all Users.

A user story would usually be written as such; "E.g. As a < type of User >, I want < some goal > so that < some reason > "however it does not need to have a predefined format.

It provides a straightforward way to enhance communication between the stakeholders and help software team better understand the system. It also helps to prevent omitting important requirements.

User stories define what needs to be built during the development of the system.

As a/an	I want to	So that
Administrator	Add an organisation	I can add all the users to the organisation.
Administrator	View an organisation	I can see if the organisation details are up to date.
Administrator	Edit the organisation details	I can update the details of the organisation.
Administrator	Add users	I can keep users' list up to date.
Administrator	View a list of all users	I can keep track of all the users.
Administrator	Delete users	I can keep users' list up to date.
Administrator	Add courses	I can assign modules and users to courses.
Administrator	View courses	I can see if the courses are up to date.
Administrator	Edit courses	I can update and make changes as required.
Administrator	Delete courses	I can remove unnecessary contents.
Administrator	Add modules	I can assign users to modules.
Administrator	View modules	I can see if the modules are up to date.
Administrator	Edit modules	I can update and make changes as required.
Administrator	Delete modules	I can remove unnecessary contents.
Administrator	Reset a password	I can help a user if he is having difficulty with his password.
Administrator	Back-up all the data	I can retrieve the data after a system failure.

Figure 4.1 – User stories for administrator

All the user stories can be found in the Appendix B.

4.1.3 Use cases (UC)

UC is a step by step on how to do the user story. It is a sequence of interactions between actors and the system to achieve a goal. It is a coherent story on how the system will behave while being used. UCs are easy to understand and provide an excellent way to communicate between stakeholders and can be used to support other phases of the software development such as cost estimation, project planning. They will focus on the main requirements of the SPMS. Table 4.2 shows all the steps needed for users to invite a team member their project and complete the flow.

Use case 2	Invite Team member	
Goal in context	A Team leader invite a new Team member to join the Team	
Preconditions	The Team leader should be logged in	
Post Condition	The system is available, and the user has been added to the system, the Team leader has already created a project	
Success End condition	The Team leader invites a new Team Member	
Failed End condition	The Team leader could not invite a Team Member	
Primary, Secondary actors	Team leader, Team member	
Main Flow	**Step**	**Action**
Navigate to the side bar	1	The Team leader navigates to the sidebar to find the project section
Open current project	2	The Team leader open the current project by clicking the name of the project
Invite a new member	3	The Team leader press the “invitation” button
An invitation email is sent to the user	4	An email is sent to the future Team Member
The user validates the invitation	5	The user who received the email validate the invitation by pressing a redirection link
Alternative Flow		none
Priority		must
Performance		The email should be sent within 30 seconds for the user to validate the invitation
Frequency		

Table 4.2 – Use case for Invite member

UCs that will focus on the main requirements of the SPMS (Appendix C) and listed below:

- Use case 1: Create a project
- Use case 2: Invite a member
- Use case 3: Create a task
- Use case 4: Assign a task

4.1.4 Use case diagrams

A use case diagram is used to capture, clarify and organise the system requirements from a user perspective as well as showing the interactions between the actors, the use cases and the system.

For better understanding it will be easier to split main functionalities of the system in multiple use case diagrams. It will be easier to see the association for different type of users. A Task and a subtask have the same functionalities. The only difference is that a subtask always

belongs to a task. All the functional requirements related to tasks also apply for subtasks. All the use case diagrams can be found in the Appendix D.

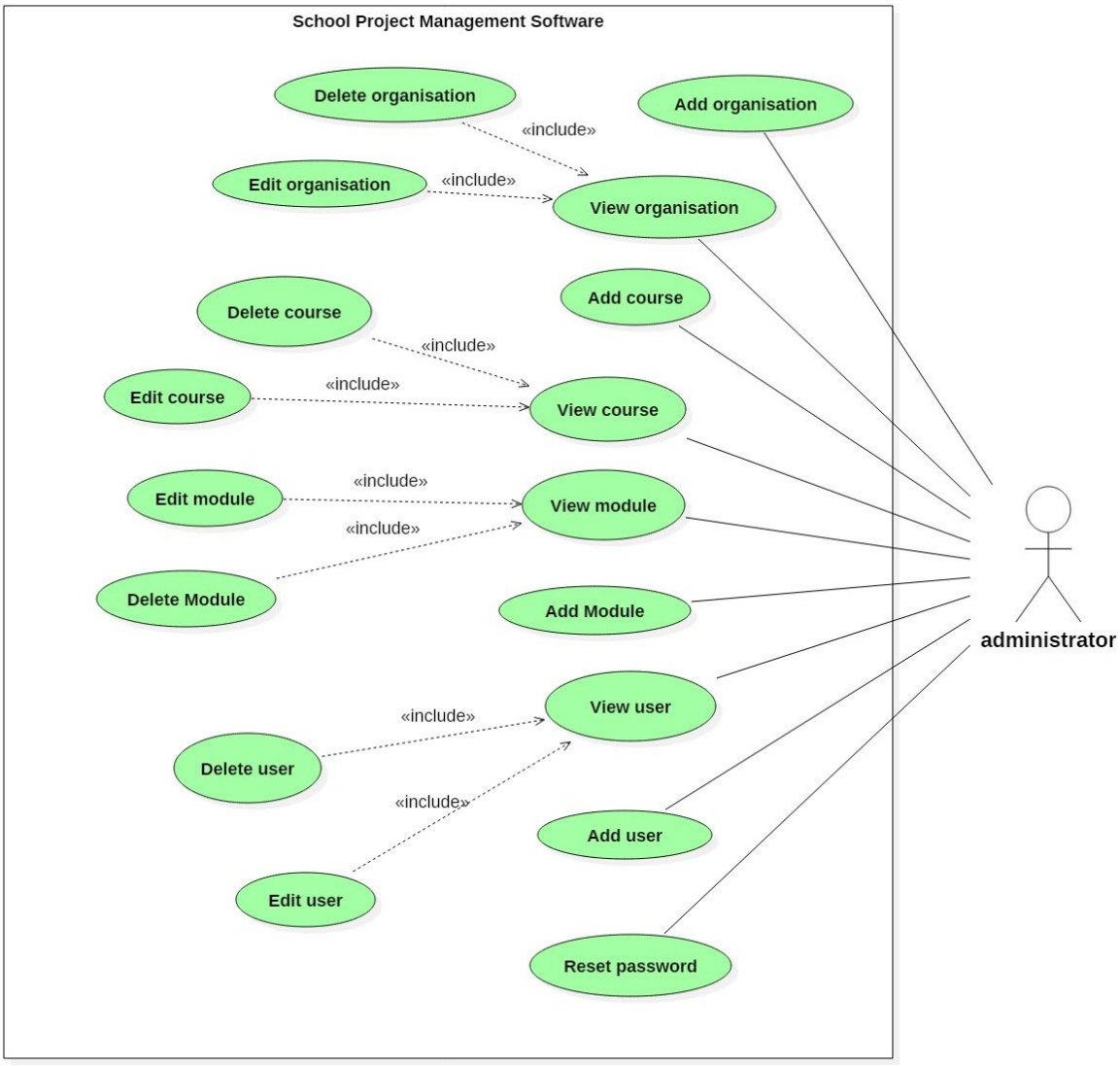


Figure 4.3 – Use case diagram Administrator

4.1.5 Negotiation and Validation

Requirements gathered during the elicitation process may overlap or conflict. Some may not be discovered yet and some requirements may be ambiguous or impractical. The negotiation and the validation cannot be dissociated and are executed in parallel during the elicitation process to refine the requirements. All the requirements present in the draft will be negotiated with the client and removed if necessary while new requirements may be added. Once the negotiation is completed with a clear list of the requirements, meeting with the stakeholders will be conducted to validate the requirements.

4.2 Requirement Management

4.2.1 Functional requirements

Functional requirements define a function of the system or its component. The following list in the Table 4.4 represents the main functional requirements gathered during the several elicitation techniques, negotiated and validated. It is important to mention that Task and a Subtask have the similar functionalities. The only difference is that a subtask always belongs to a task. All the functional requirements related to tasks also apply for subtasks.

FR ID	Functional requirements	FR ID	Functional requirements
01	Add organisation	37	View objective
02	Edit organisation	38	Add goal
03	Delete organisation	39	Edit goal
04	View organisation	40	Delete goal
05	Add user	41	View goal
06	Delete user	42	View project status
07	Edit user	43	Edit project status
08	View user	44	Edit task status
09	Add project	45	View task status
10	Archive project	46	Upload content
11	Edit project	47	Download content
12	View project	48	View content
13	Add project member	49	Upload assignment
14	Delete project member	50	Download assignment
15	View project member	51	View assignment
16	Create password	52	Add module
17	Reset password	53	Edit module
18	Update password	54	Delete module
19	Add task comment	55	View module
20	Edit task comment	56	Add course
21	Delete task comment	57	Edit course
22	View task comment	58	Delete course
23	Add project comment	59	View course
24	Edit project comment	60	View overall progress
25	Delete project comment	61	View deadline project
26	View project comment	62	Add deadline task
27	Create task	63	Delete deadline task
28	Edit task	64	View workflow step
29	Delete task	65	Add workflow step
30	Assign task	66	Delete workflow step
31	Un-assign task	67	View dashboard
32	View task	68	Edit dashboard
33	Validate task	69	View user settings
34	Add objective	70	Edit user settings
35	Edit objective	71	Edit user profile
36	Delete objective	72	View user profile

Table 4.4 – Functional requirements

4.2.2 Non-functional requirements

Non-functional requirements define the system properties and the constraints. They may be more critical than functional requirements and need to be considered for the system to be fully functional. The list of the main non-functional requirements captured during interviews with users and the prototyping phase can be found in the Appendix E.

4.2.3 List of Requirements / MoSCoW

MoSCoW prioritisation will be used to prioritise all the requirements. The complete list of prioritised requirements can be found in the Appendix F.

MoSCoW stands for:

Must have, the requirements are guaranteeing to be delivered. If one of the requirement is not included in the final product the project is considered as a failure.

Should have, the requirements are important but not vital for the product to be completed. They are not as critical and can be held back until a future delivery timebox.

Could have, the requirements are not necessary but could in some instance, improve user experience or customer satisfaction.

Won't have this time, the requirements are not needed for the project and will not be delivered. The requirements will be either dropped or reconsidered in a later timebox.

4.3 Conclusion

All the main functional and non-functional requirements have now been clearly elicited and prioritised with MoScoW. The client may introduce new functionalities so the requirements will be revisited as they may change during the development process. Use cases and use case diagrams were helpful to capture, clarify and organised the requirements and to show the interactions between them and the users.

5 Design

Following the gathering of the essential requirements and filtering the crucial information and requests from the stakeholders, the design phase will be focusing on documenting and designing the database as well as the User Interface (UI) of the system.

The design process, an important phase of the agile methodology, follows the product during its entire lifetime. Modifications of the design are required to improve the product but also to integrate and accommodate new requirements for the iterations during the development phase. The design team will meet with the business, development team and testers to present the design and eventual modifications or issues that may occur during iterations. In DSDM, prototyping helps to ensure that the system is correctly designed and gives an overview of the product for all stakeholders. Understanding the interaction between the system and the users is essential during this process to ensure the best User Experience (UX).

The design chapter contains three important sections:

- The database design supported by the entity relational diagram and the data dictionary. The system data need to be stored safely without any duplication and should be accessible at any time.
- The modelling system interaction that includes activity and sequence diagram. Understanding the interaction between the user and the system is a really important factor allowing to capture the dynamic behaviour of the system during the design process.
- The interface design that comprise the design methodology, sketches, wireframes, navigation and design decisions. The interaction between the user and the interface is an essential factor for the product to fulfil the requirements.

The design phase is a good way of gathering the feedback from the stakeholders, ensuring the project is on track, better understanding the system, discovering where the complications stand and ensuring a positive user experience.

5.1 Database design

5.1.1 Entity Relational Diagram

The entity relational diagram (ERD) shown in the Figure 5.1 is a graphical representation of the system illustrating the relationships between people, objects, places, concepts or events within the system. It is a data modelling concept that helps determine business processes.

The ERD elements are:

- Entities (real-world item or concept that exist on its own).
- Relationships (association that describes the interaction between entities).
- Attributes (property that describes the entity).

At first, the entities are identified and defined (e.g. Users) and then the interactions between the entities are determined to define the cardinality of the relationships. In Figure 5.1 “A user” can only be part of one organisation but the organisation can have many users. ERD is essential when creating good database design.

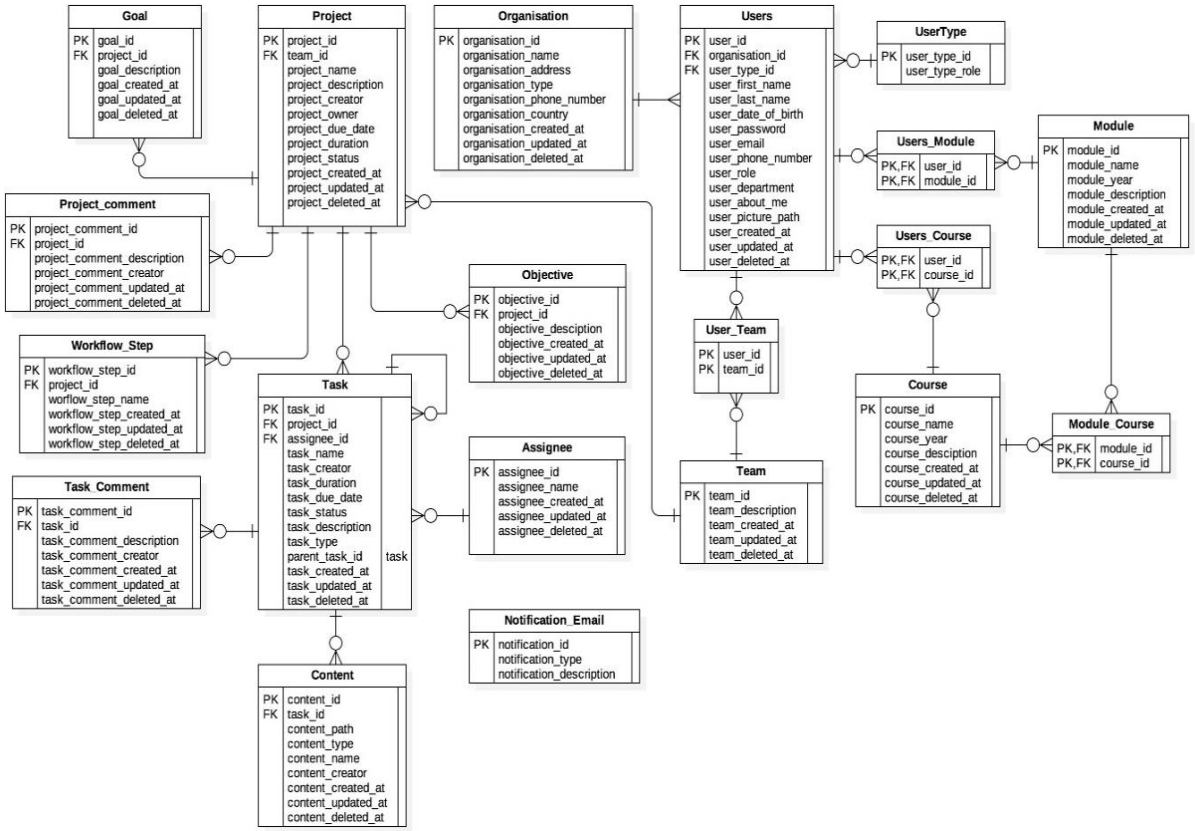


Figure 5.1 – Entity relation diagram

To perform data modelling, it is necessary to know the functionalities that are being developed and have clear user stories. The functional requirements should be well defined at this stage. The entities have been derived from the functional requirements and user stories extracted during the requirement analysis. The entities, attributes and relationships have been validated during meeting and brainstorming with the stakeholders in order to support the data stored in the database.

The ERD diagram in figure 5.2 includes three essential sections. The first section of the ERD diagram represents the relationships between users, modules, courses and organisations. A user is only part of one **Organisation** and an **Organisation** can have many **Users**. This is represented by one to many relationship in the diagram. A user would only have one unique **UserType**. Within the system, a user can only be an administrator, a lecturer or a student. This is represented by the relationship one to many between **Users** and **UserType**. The client has agreed that a student could only be enrolled in one course but can have multiple **Modules**. On the other hand, a lecturer can be assigned to multiple **Courses** and **Modules,** and it is modelled with many to many relationship between **Course** and **Module**, **Users** and **Course** and **Users** and **Module**. Many to many relationships have been resolved by introducing the tables **Users_Course, Module_Course** and **Users_Module.**

It is necessary that the loop represented in the diagram starts and finishes with **Users** table. The client specified that a maximum of four modules will be assigned to a course but a student or a lecturer may not necessary be assigned to all the modules.

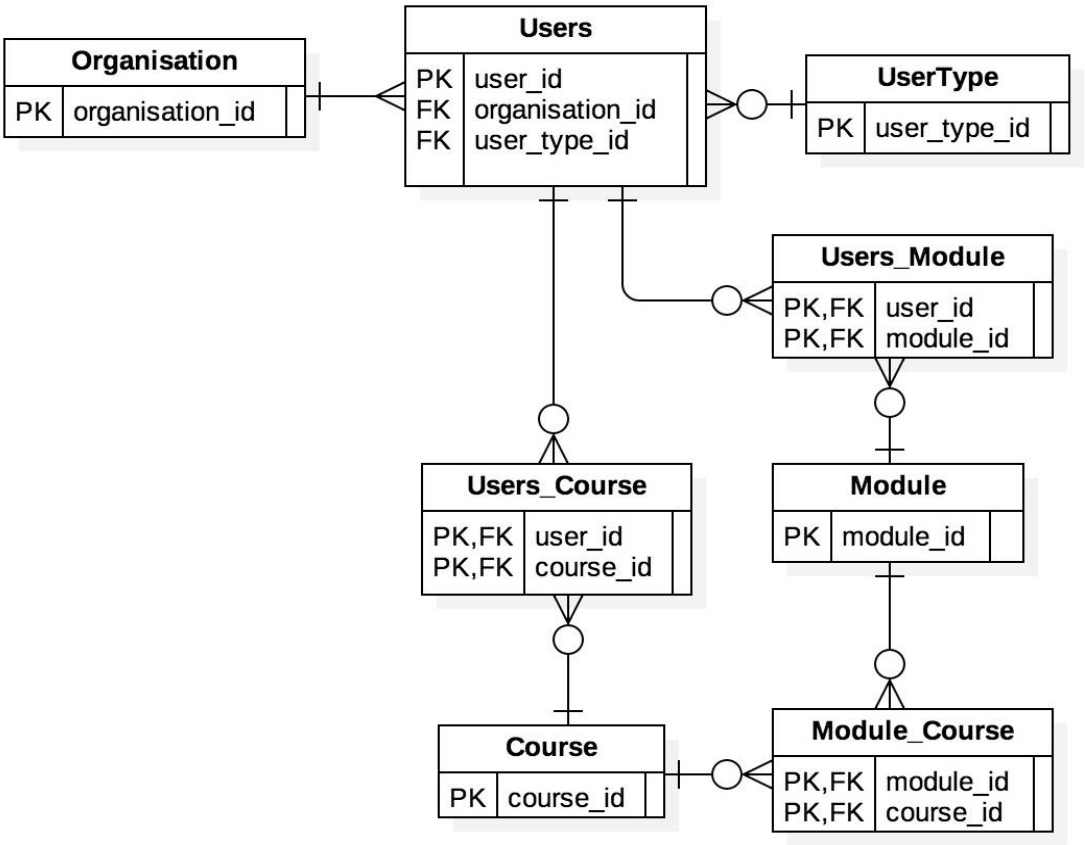


Figure 5.2 - ERD User section

The second section of the ERD diagram in Figure 5.3 represents the relationship between **Project**, **Task**, **Comment**, **Goal**, **Objective**, **Workflow_Step**, **Content** and **Assignee** tables. The **Task** table contains a recursive association that represents the association between **Task** and **Subtask**. As mentioned in the requirements a **Task** can enclose multiple **Subtasks** but a subtask can only have a single parent **Task**. A **Task** can only be related to one single project. Following the requirements from the client, a project should accommodate multiple **Goals**, **Tasks**, **Comments**, **Objectives** and **Workflow_Steps.** This is described by one to many relationship between the **Project** table and the tables aforementioned. The client also specified that a user should be able to assign a task, comment on the task as well as upload some content directly to the task. A **Task** may have many **Comments** and **Contents** but will always have one **Assignee**.

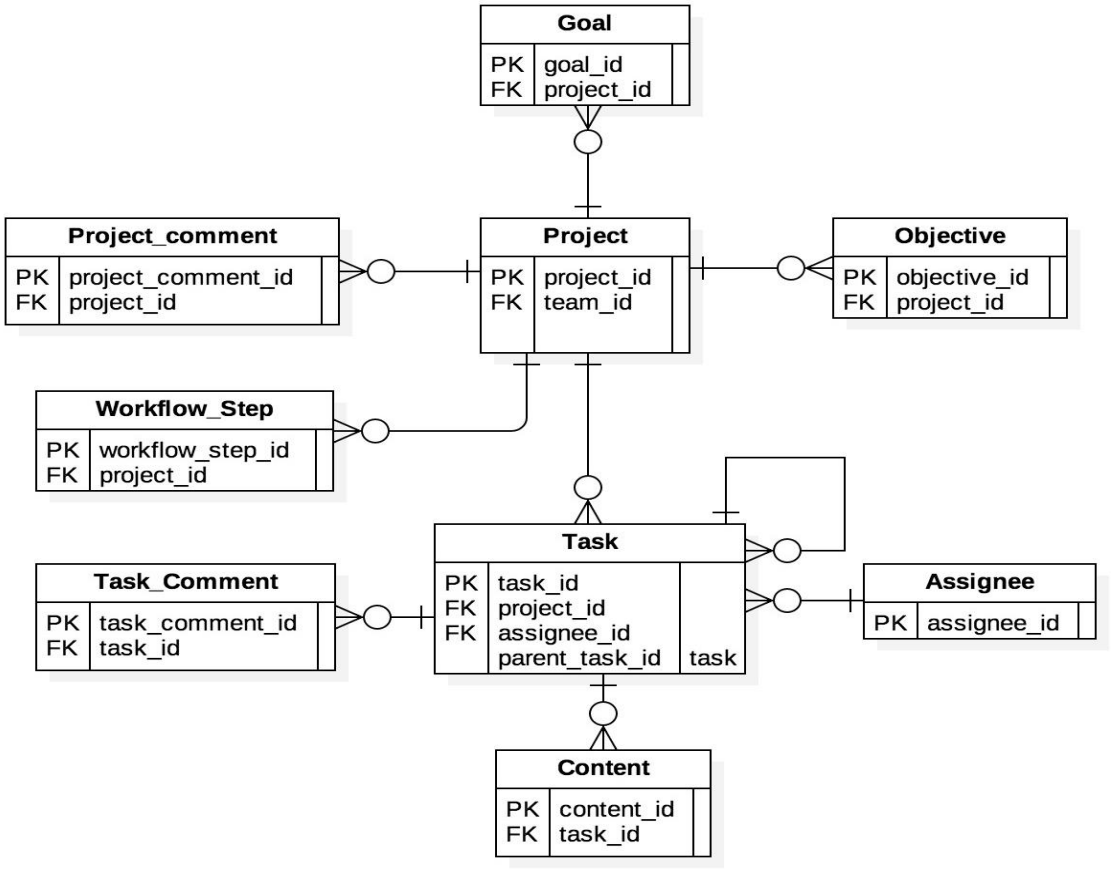


Figure 5.3 - ERD Project section

The third section of the ERD diagram in Figure 5.4 represents the relationship between **User**, **Project** and **Team** tables. The client wants to ensure that the system allows users to be part of the multiple teams and work on multiple projects. The relationship between **Users** and **Team** is a many to many and it has been resolved by inserting the table **User_team**.

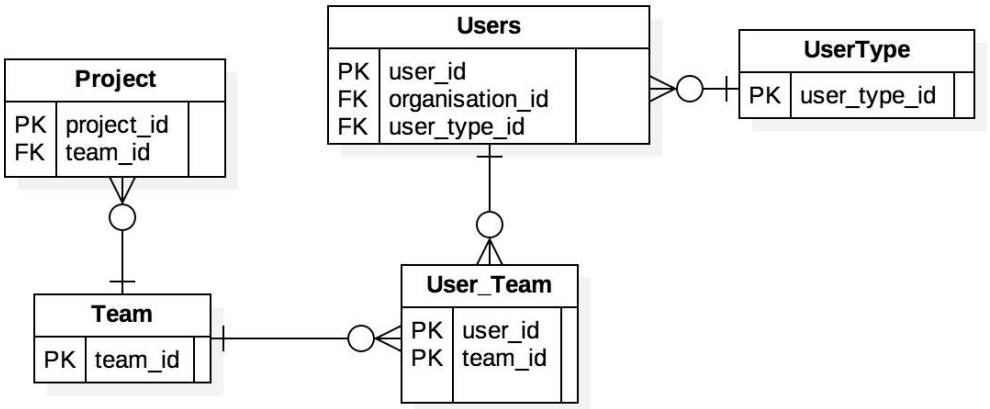


Figure 5.4 - ERD Team section

5.1.2 Data Dictionary

The data dictionary is a document designed for administrators. It describes a database schema, provides well-structured and clear information about the database. Following the identification of the database entities, relationships and attributes, it is important to create the data dictionary. Data dictionary includes the names and descriptions of the tables and the fields contained in each table. It also documents information about data type, field length and the constraints for each attribute. The data dictionary is only accessible by the data management staff and is hidden from the users to prevent any data loss or damage.

relation name	attribute name	Data type	length	PK/FK	NOT NULL	other constraints
Organisation	organisation_id	number	20	PK	NOT NULL	AUTO INCREMENT
	organisation_name	varchar	250		NOT NULL	
	organisation_address	varchar	500		NOT NULL	
	organisation_type	varchar	250		NOT NULL	
	organisation_phone_number	varchar	30		NOT NULL	
	organisation_country	varchar	100		NOT NULL	
	organisation_created_at	datetime				
	organisation_updated_at	datetime				
	organisation_deleted_at	datetime				

relation name	attribute name	Data type	length	PK/FK	NOT NULL	other constraints
User_Type	user_type_id	number	20	PK	NOT NULL	AUTO INCREMENT
	user_type_role	varchar	250		NOT NULL	

relation name	attribute name	Data type	length	PK/FK	NOT NULL	other constraints
users	user_id	number	20	PK	NOT NULL	AUTO INCREMENT
	organisation_id	number	20	FK	NOT NULL	
	user_type_id	number	20	FK	NOT NULL	
	user_first_name	varchar	250		NOT NULL	
	user_last_name	varchar	250		NOT NULL	
	user_date_of_birth	date			NOT NULL	
	user_password	char	250		NOT NULL	
	user_email	varchar	250		NOT NULL	
	user_phone_number	varchar	30		NOT NULL	
	user_departement	varchar	250		NOT NULL	
	user_about_me	varchar	500			
	user_picture_path	varchar	500			
	user_created_at	datetime				
	user_updated_at	datetime				
	user_deleted_at	datetime				

relation name	attribute name	Data type	length	PK/FK	NOT NULL	other constraints
Course	course_id	number	20	PK	NOT NULL	AUTO INCREMENT
	course_name	varchar	250		NOT NULL	
	course_year	date			NOT NULL	
	course_description	varchar	1000		NOT NULL	
	course_created_at	datetime				
	course_updated_at	datetime				
	course_deleted_at	datetime				

Figure 5.5 – Data dictionary

The tables in figure 5.5 do not represent the complete data dictionary, this can be found in Appendix G. The primary keys (PK) representing a unique row in the table are auto incremental. If a new row is inserted in the table **Users**, the user_id will be incremented by one to ensure the uniqueness. The data type number for the IDs contains a length of 20 to ensure the system can support large number of users and data. Some fields in the database may be mandatory and are represented with the constraint NOT NULL. The PKs and foreign keys (FK) should always be set to NOT NULL. Varchar are used for text field but can also support numbers.

5.1.3 Conclusion

The database should accommodate the necessary data for the system. Once the design is completed, the “many to many” relationships should be resolved and the entities, PKs, FKs and attributes should be clearly identified. The validation of the database design will be enforced when implementing and inserting data, as well as querying the relevant data. A good database design ensures data integrity and eliminates data redundancy. Duplicated data not only wastes money but also storage and can lead to inconsistency and ambiguity. The performance of the database can be affected by a bad design and become slow, inflexible, hard to extend and

maintain. After multiple iterations during the design of the database, the client is now confident that the database can accommodate the data.

5.2 Modelling system interactions

5.2.1 Activity Diagram

Activity diagram is a behavioural diagram used to capture the dynamic behaviour and represents a series of actions or flow in the system. It is a simple and intuitive illustration of what happens in a workflow, what activities can be executed in parallel and if alternative paths are present. They can describe the steps in a use case diagram and show the workflow from the start point to finish point detailing many decision paths that are present in the progression of events contained in the activity. Activity diagrams are useful for business modelling and detailing the process involved in the business activities. The diagram in Figure 5.6 represents the flow of the user logging into the system. The main elements of the activity diagram are as follows:

- The start point of the activity diagram represented by a black circle.
- The activity, a non-interruptible action of objects represented by a rounded rectangle.
- The action flow, illustrating the transitions from one action to another represented by an arrow.
- The decision represented by a diamond defines an alternate path required prior to moving to the next activity. It is always labelled with a condition or a guard expression.
- The end point, the final action state, is represented by a filled circle nested inside another.
- Swimlane is a way to group activities performed by the same actor and it is represented by a column.

In the activity diagram we considered the user already register to the system. Beginning with the “start point” at the top of the activity diagram, the system will first display the login page to the user. After entering the login details, two paths are available. If the user can login, the system would redirect the user to the home page, display it, and the flow would be completed. If the user cannot login, the system will display an error message and sensible feedback. As an alternative the user will be able to re-enter the login details or press “forgot password” link to reset the password. The system will then send an email for the user to reset the password. If the email address is valid, the user will receive an email, reset the password and complete the flow. If the email address is invalid the user will have the possibility to retype the email or contact the administrator.

Only main activity diagrams were designed to demonstrate the non-linear activities as shown in the Appendix H. The client wanted to prioritise the following complex activity diagrams:

- The user login
- The user creates a project
- The user creates a task
- The user invites a member

Other activity diagrams may be considered if time is available.

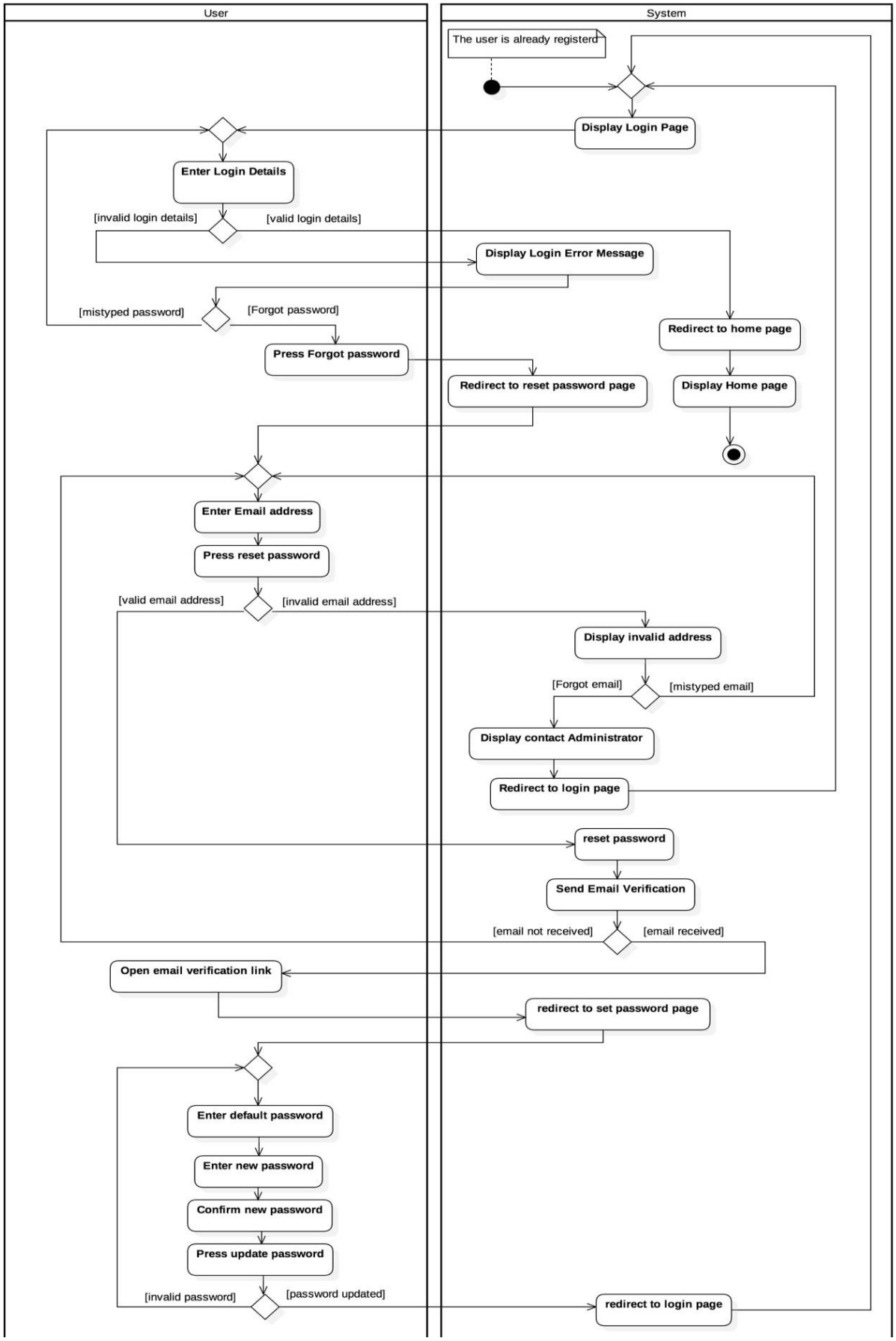


Figure 5.6 - Activity Diagram User Login

5.2.2 Sequence Diagram

Sequence diagrams are used to show the interactions between objects in the sequential and chronological order to fulfil a specific task. They represent the objects and classes involved in the scenario and the sequence of messages exchanged between objects that are necessary for the functionality of the scenario. Sequence diagrams are usually associated with the creation of the use cases and are really useful to document system's requirements and flush out the system's design.

The diagram in Figure 5.7 represents the interaction between objects when a user wants to login to the system. The main elements of the sequence diagram are the following:

- Participants describing the way an object behaves in context represented by a horizontal rectangle.
- Activation defining the time an object needs to complete a task represented by a vertical rectangle.
- Message defining the communication between two objects represented by an arrow. Half-arrowed lines will be used for asynchronous messages.
- Lifelines that indicates the presence of the object represented by a vertical dashed line.
- Destroying object is represented with a cross on the lifeline.

The flow presented in the sequence diagram is the same as in the activity diagram. The sequence diagram represents the interaction and messages exchanged between the user, graphical user interface, backend, database and the Application Programming Interface (API). The other sequence diagrams can be found in Appendix I.

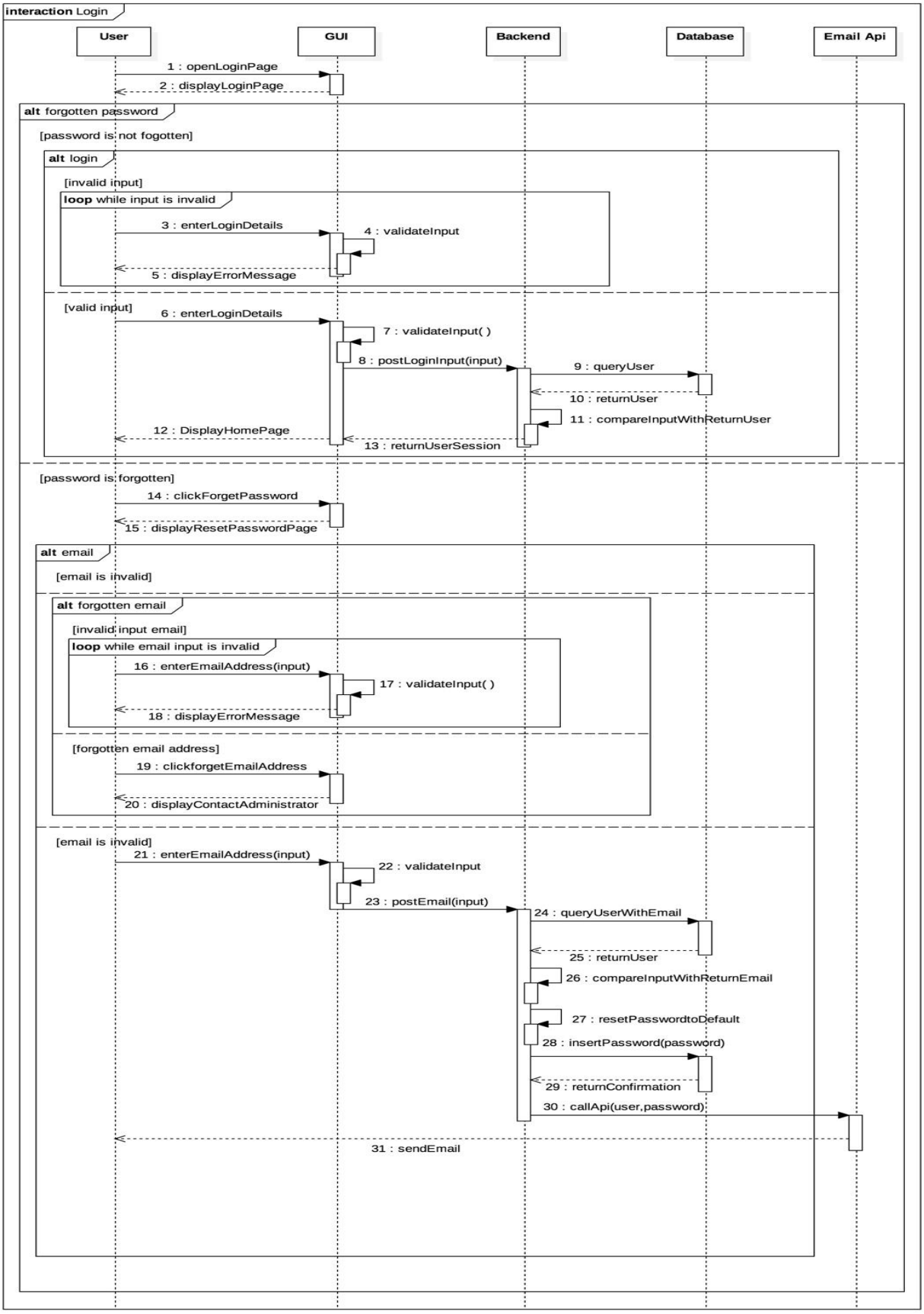


Figure 5.7 - Sequence Diagram User Login

5.2.3 Conclusion

The activity diagrams are derived from user stories and can be in some case really similar. However, designing the activity diagram reinforced the quality of the system and ensured that activities were present in the flow. Activity diagrams can also help discover other alternative flows and are easier to interpret (visual representation) compared to user stories. Sequence diagram is good at depicting sequence of messages flowing between objects but are not as intuitive as activity diagrams. Activity diagram is good at illustrating the flow between activities, especially for conditional structures such as loop and concurrency. The client could see that the complex flows have been clearly identified and depicted during this process and was really satisfied.

5.3 Interface Design

The interface design is essential during the development of the system to ensure the satisfaction of the user. It is important to consider which methodology will be used during the design. Sketches, wireframes and sitemaps will be used during this process to reinforce and improve the user experience (UX). It will be essential to get feedback from the users during multiple iterations. Design decisions will be made with the client to match the existing system.

5.3.1 Human Computer Interaction (HCI) considerations

HCI is a field of study that emerged in early 1980s, focusing on the design of computer technology, and principally on the interaction between humans and computers. It embraces numerous fields such as computer science, human-factors engineering and cognitive science. HCI design approaches are applied to user interface designs to develop a user-friendly, intuitive and efficient website to improve the user experience. It is a very important concept in the development process to produce user friendly and functional systems. To summarise, HCI is concentrates on how to improve and facilitate the user's view, navigation and use of the website and make it as easy as possible for the user.

5.3.2 Design Methodology

Diverse methodologies derived from human-computer interaction design have emerged since early 1980s. User centred design, is an iterative design approach that focuses on user needs and requirements to produce highly usable and accessible systems. It is based upon a clear understanding of the users, tasks and environment and aim to capture the “whole” user experience. Iterations of the UCD approach includes normally four phases:

- Designer should first understand the context in which the system is used.
- The user's requirements are stated.
- The design phase.
- The evolution phase.

The project will mainly focus on this approach during the design of the software, involving the users during numerous iterations, to get constructive feedback and constantly improve the product and the user experience.

5.3.3 Sketches

Sketches at early stage of the design, are valuable exercises offering a visual context that can provoke further ideas, provide more thinking and potential solutions. Sketches should be simple and rough, and they ought to stimulate ideas without worrying about quality and provide a good way of sharing, discussing, criticising them with the client.

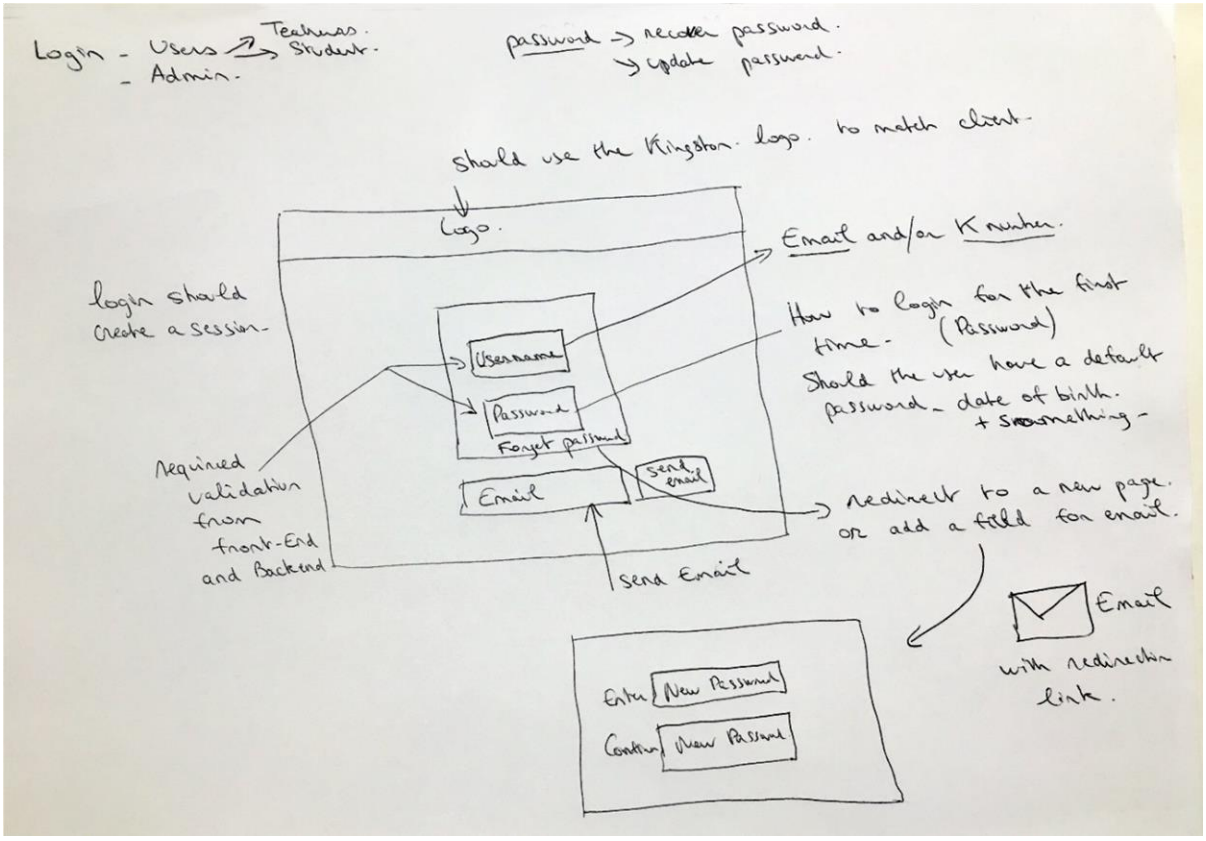


Figure 5.8 – Sketch for login

The sketch in Figure 5.8 was used to extract information about login page and get feedback from the client. This sketch shows users' (Lecturers/Students) and administrators login page. They will be using the same page to login to the system, but depending on the type of user, they will be redirected to two different pages. After revising and discussing the sketch with the client, it was agreed that users will be using their email address to login to the system. If the users forget their passwords, they will have the opportunity to reset it by validating their email address and will receive an email to reset it. An API will be used to send automatic emails. If a user forgets the password and email address, they will have to contact the administrator. Sketches were essential to ease the communication and interaction with the client and to agree the main decisions of the user interface. The sketches can be found in the Appendix J.

5.3.4 Wireframes

Wireframes is a more complex and detailed approach than sketches as it should include much more details. Fonts, logos, images and colours should be disregarded at this stage as wireframes focus only on representing the major components, layout, structure, content placement, and solving navigation and functionality problems. It is a vital process to avoid any confusions during the design phase and it provides a good view of the structure for the client.

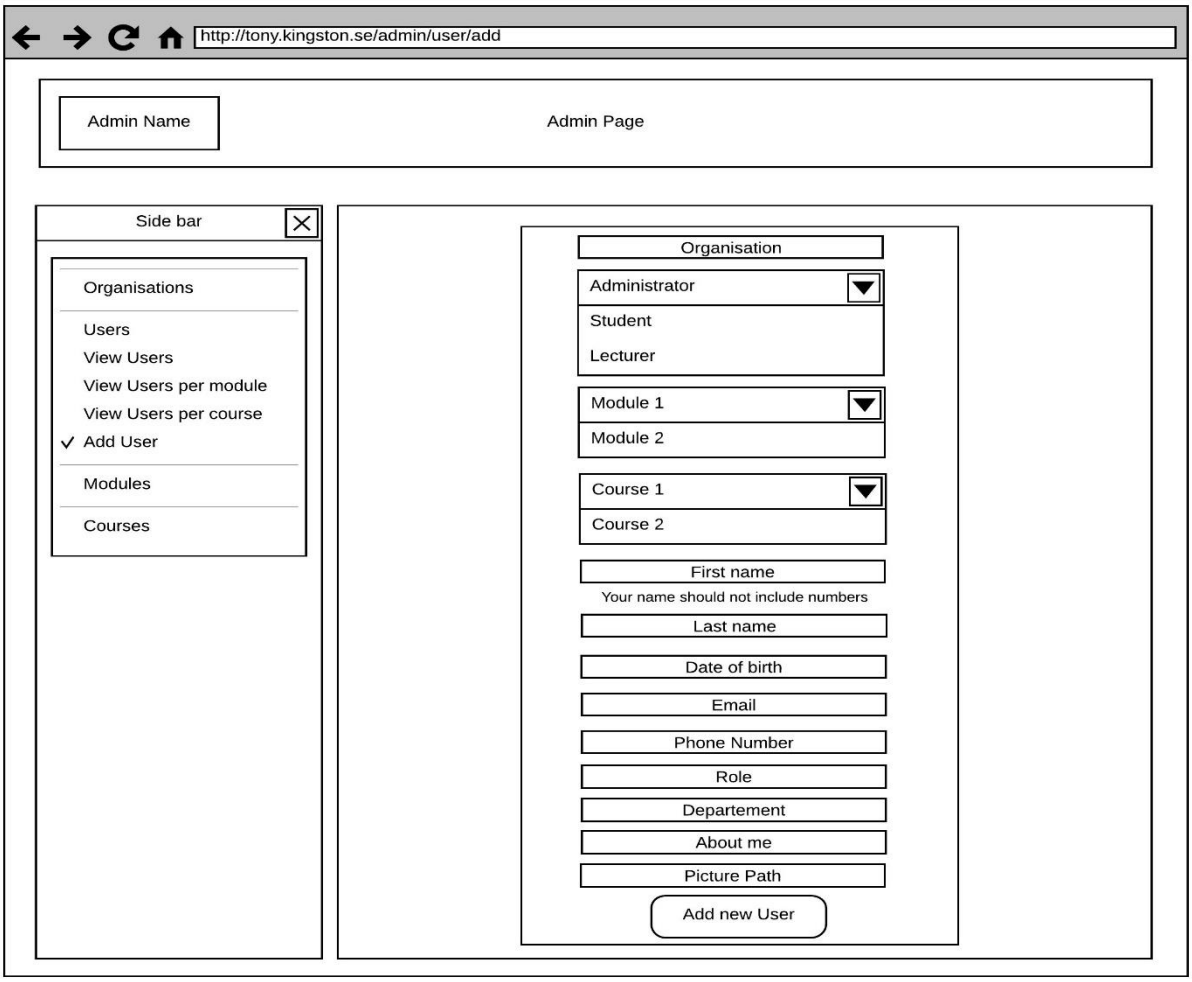


Figure 5.9 – Wireframe Add a user to the system

The wireframe Figure 5.9 represents the page "add user" and is only accessible by the administrator. This is the form that needs to be completed by the administrator to add a new user to the system.
The sidebar can be minimalised, by pressing the cross, to allow more space to view the page. The name of the administrator currently logged in will be displayed in the "Admin Name" box at the top left corner in the header. The current page has been opened by clicking on "Add user" menu in the sidebar. The fields in the form are mandatory to add a new user. The "Add new user" button will not be clickable until the fields are validated. A sensible feedback will be displayed underneath each field if they are not valid. Module and course dropdowns will only be available when the administrator wants to add a lecturer or a student to the system. Once the form is validated and the administrator clicked on the button "Add new user", the user will be inserted into the database, the form will be cleared out and a message will be displayed to confirm that the user has been added to the system. The other wireframes can be found in Appendix K.

Important decisions and agreements were made with the client during this process such as navigation, design and layout and will be discussed in the following sections. It was not

necessary to focus on the wireframes for mobile application because the software will only be used on desktop for the time being but will require to be compatible with modern browsers. Mobile application may be implemented later and will require more wireframes.

5.3.5 Navigation

Sitemaps

After creating sketches and wireframes, it is easier to plan the website's page architecture. The sitemaps will provide an overview for the client and help organise, clarify the pages that need to be implemented as well as exclude unnecessary pages. A sitemap will also clarify the site's goals before designing and creating any contents, and ensure that the content of the pages is not duplicated. It is necessary to avoid redundant steps and combine them when possible to improve and ease the navigation of the user. A sitemap should be understandable to the stakeholders to ensure everyone is on the same page. These two sitemaps in Figure 5.10 and 5.11 have been designed to represent main webpages and their associations with each other, starting from the home page.

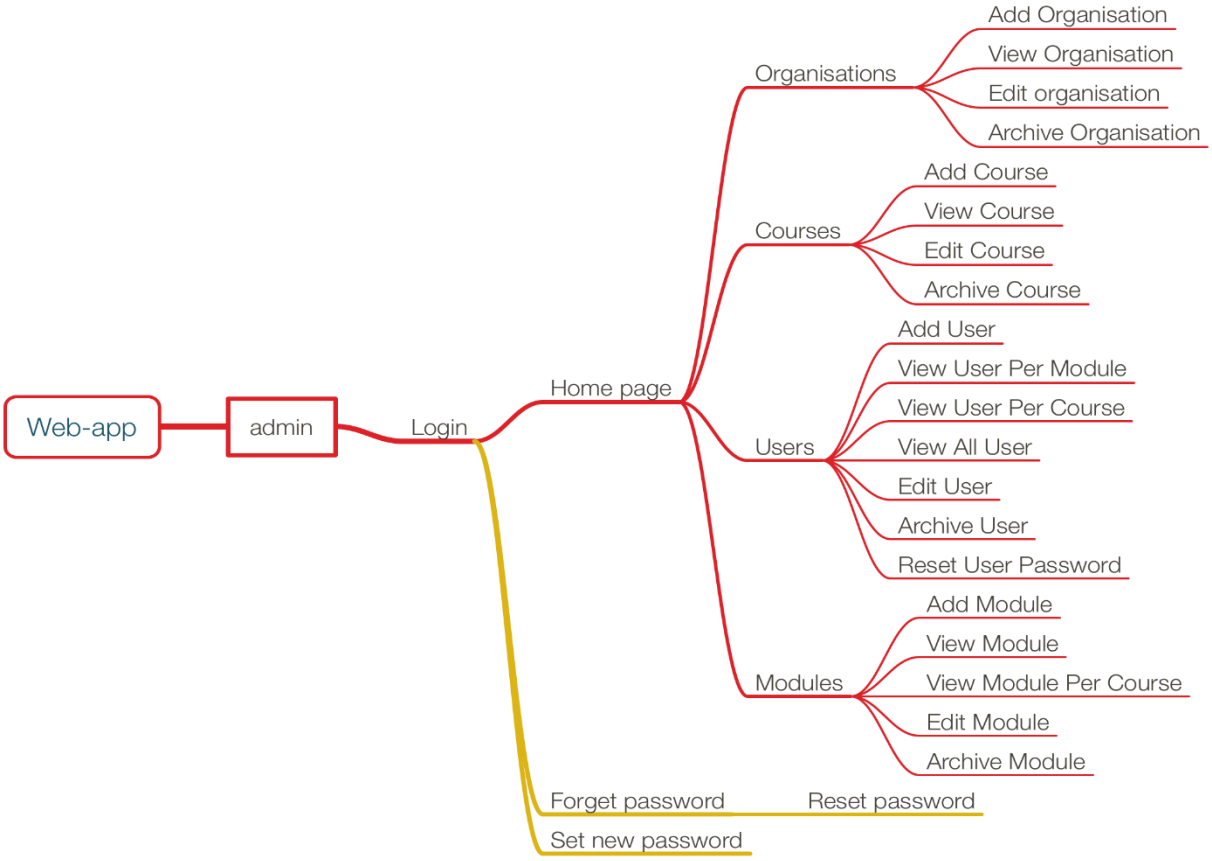


Figure 5.10 - Sitemap Administrator

The sitemap in Figure 5.10 represents the necessary pages for the administrator of the system. Starting from the left, if an administrator cannot login to the system, three pages shown in yellow will be available. If the administrators forget the password, they will need to click on "reset password" to be redirected to reset their password. If the administrators can login to the system, they will be redirected to the home page. The main menus: organisations, courses, users and modules will be represented in the sidebar and are not actual pages. The subsections on the left of the tree represent the administrator's pages and have derived from the main requirements validated by the client. It is important to mention that four essential pages for every main section are directly derived from the four basic functions of the CRUD (create, read, update and delete/archive) and will be directly in correlation with the way the interface interacts with the backend and the database to create, read, update or archive the data.

Archiving has been chosen as opposed to deleting because the client wants to keep a record of the data.

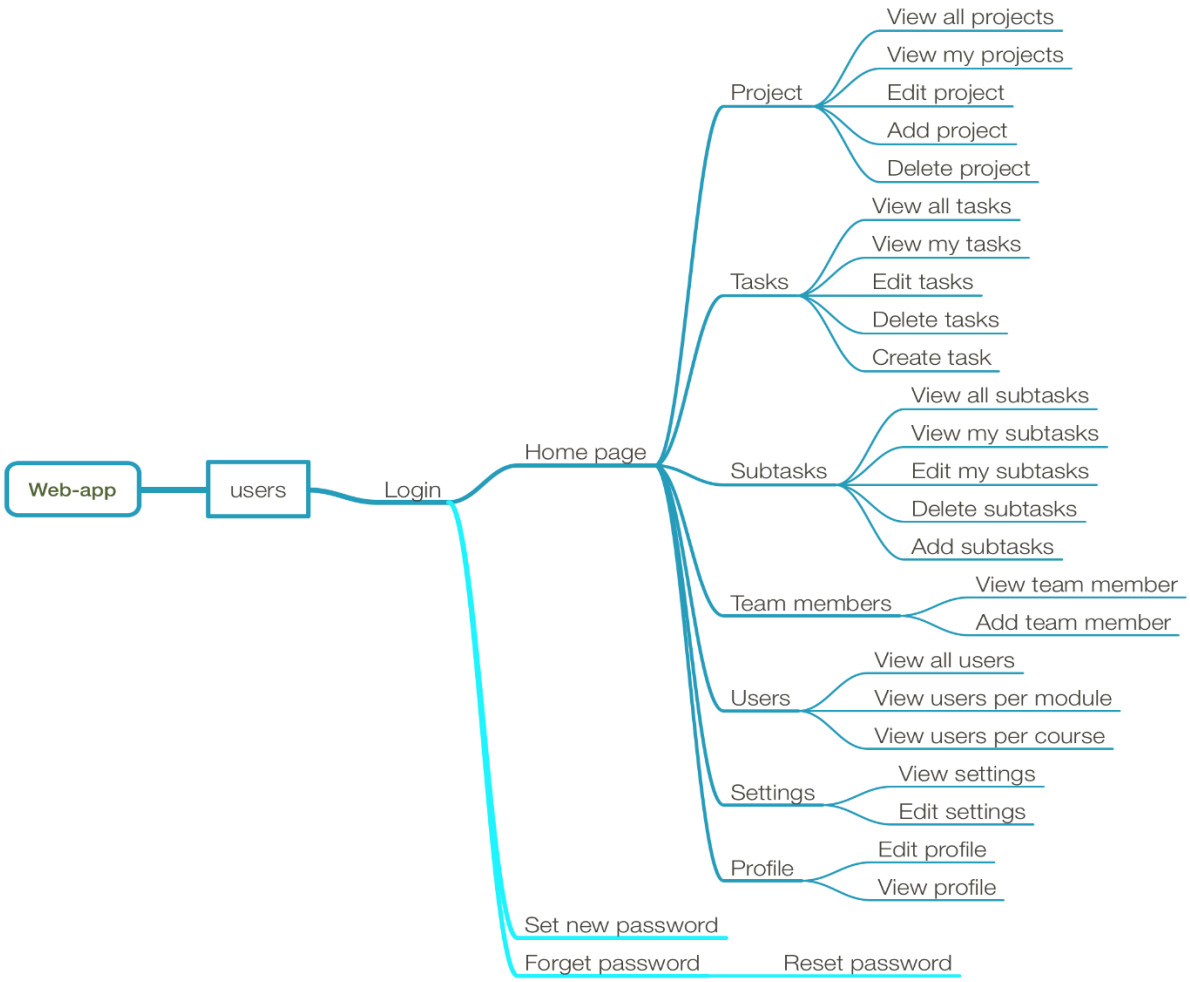


Figure 5.11-Sitemap Users

The sitemap Figure 5.11 represents the necessary pages for the users (Lecturers/Students) of the system. Starting from the left of the sitemap, if a user cannot login to the system, three pages shown in light blue will be available to the user. If the users forget their password, they will be redirected to the reset password page. If the users can login to the system, they will be redirected to the home page. The main menus, projects, tasks, subtasks, team members, users, settings and profile are not actual pages. The subsections on the left of the tree represent the user's pages and have been derived from the main requirements validated by the client. The main pages from every section are directly derived from the four basic functions of the CRUD and will be directly in correlation with the way the interface interacts with the backend and the database to create, read, update or archive the data. The pages "delete project", "delete tasks" and "delete subtasks" are only named that way to do not confuse the user but the projects, tasks and subtasks will be archived and not completely removed from the system.

Conclusion

Both sitemaps have only few pages in common so it was agreed with the client to completely separate the user from the administrator pages, as it will be easier to maintain, except for the login page that can be used across the system.

5.3.6 Design decisions

This section will discuss in more details the choices made for the layout, colours, images, fonts and accessibility of the website. The product is directed to Kingston University and should take in consideration the existing design.

Site layout

The layout of the website will need to be consistent, and user friendly. It should be kept as simple as possible to improve the user experience. The top navigation bar will only provide the possibility to access the profile and details of the user, a search bar to look through the entire website and a logout button that should always be visible to the users. The sidebar will give access to main functionalities of the website. Only main sections will be visible when opening the sidebar but the users will be able to expand each section to access other functionalities. The user will be able to toggle the sidebar to leave more space and have a better view of the content of the page. The footer will only be shown on the login page, to leave more space for the main content. The layout should resize and adapt according to the size of the screen.

Colour, images and fonts

The colours of the website have been chosen according to the client. The main colours of the client's current website are blue and white. The colour blue represents trust, security, stability, peace and calmness. For a better user experience, it is important to exclude excessive and unnecessary colours. The logo should be matching the dominant colour of the website so that the audience can remember it. Only one font-family will be used for the entire website: "Segoe UI, Roboto," Helvetica Neue", Arial and sans-serif. The size of the font will be different according to the importance of the content. Icons representing main menu functionalities should be clearly understandable and memorable for the user. The buttons should be consistent across the website and should have enough contrast to be clearly visible. The shape of the buttons will be different depending on their action. As an example, the submit button will have a green colour that contrast with the red colour of a delete button.

Accessibility

This section will mainly focus on people who are colour-blind. It is essential to have the right contrast and colours for colour-blind people to enable them to see clearly the content of the website. By using the colours blue, white as well as black for the font across the website and keeping it minimal I can ensure that colour-blind people will be able to have a good user experience on the website.

Conclusion

The decisions concerning design of the user interface were made according to the requirements of the client. Furthermore, interviews were conducted with students and lecturers to get feedback on the design. The user interface should be responsive, easy to access, easy to use and consistent. The software will be only implemented for desktop for now and a mobile application may be developed later.

5.4 Conclusion

This chapter focused on the design of the software and took into consideration the requirements from the client and their existing website as well as the recommendations from the users. The efficient and logical database model has been designed to support data, avoid any redundancy and ensure data integrity and consistency. It is essential to spend more time on database model design since it is the heart of the project. The database will be built, populated and tested in the implementation phase over multiple iterations and the design may be revisited to improve the database. A data dictionary has been designed to ease the view of the current database and keep track of any changes. During the design of the database, modifications have been made to ensure that lecturers can be assigned to multiple courses and modules and students can now be assigned to only specific modules of a course.

Activity and sequence diagrams were also designed to understand the workflow and the interaction between objects of the system on the main pages. This process was essential to

discover alternative flows and the interaction between the user interface, the backend and the database.

Multiple sketches and many wireframes were designed during the process to get feedback from the client and ensure a great user experience.

The design is now sufficient to start developing the database and the user interface and will be revisited continuously during the implementation phase for further improvements and requirement changes.

6 Implementation

This chapter will reflect the journey of converting the preliminary design into a working software which users can interact with. The implementation chapter will prove the necessary steps during the agile scrum implementation phase and the following will be discussed:

- Crucial tools and technologies used during the implementation of the product.
- A clear vision of the system's architecture to be implemented.
- The implementation of:
  - ✓ database using MySQL with the support of phpMyAdmin.
  - ✓ front-end using angular 5 with TypeScript.
  - ✓ back-end using the Slim framework with PHP and MySQL.
- The Security concern during the development of the software.

The aim is to clearly show how the system will be developed and how various modern technologies interact with each other. The project management system should be dynamic, flexible and supported by modern browsers on desktop application. It may be possible to view certain pages on mobile devices but for now the aim is to be fully supported as a desktop application. A mobile application may be considered later.

6.1 Tools, Technologies and Architecture

6.1.1 Tools

IntelliJ IDEA

The product will be entirely developed using IntelliJ IDE (integrated development environment). This solution is not free however, it provides the following advantages:

- Refactoring the codes with IntelliJ is reliable and efficient.
- Code completion and renaming suggestions.
- Code navigation which allows jumping to a class or declaration in the code directly.
- Built in tools and integration.
- It supports version control systems such as GitHub.

Connection to multiple databases within the IDE. Various editors have been considered during the development of the product such as visual code studio or PhpStorm, but IntelliJ offers much more possibilities. The front end will be developed using various programming languages such as TypeScript, Html and CSS and the back-end will be developed using PHP and MySQL. IntelliJ supports these languages, so the codes can be accessed and implemented in one single IDE. This will save a lot of time for the developers.

GitHub

GitHub is an open source repository service that hosts the source code of the project and supports various programming languages. It offers the following:

- The possibility to safely store the codes
- Keep track of the various changes made by the developers during the implementation phase.
- Track the changes in the code across multiple versions of the system
- Reverting the codes if they are not functional and easily deploy the new codes to the webhost.

The codes are stored locally on the machine but using GitHub is a safe and easy way to back them up to avoid the loss of the source code. The front-end and the back-end codes will be stored in two different repositories as they are completely independent from each other.

XAMPP

XAMPP (Cross-Platform (X), Apache (A), MySQL (M), PHP (P) and Perl (P)) will be used to set up a local web server to test the programming codes written in PHP for the back-end before deploying them to production on the webhost. XAMPP also provides a local database that allows the developers to test the queries and can be directly connected with IntelliJ IDEA. MAMP is an alternative to XAMPP and it offers the same functionalities, but it is a Mac-only product and developers may need to work on different platforms.

FileZilla

Filezilla, is an open source software that allows to transfer files from a local computer to a remote computer. Filezilla can run on windows, macOS and Linux. It supports FTP, SFTP (SSH file transfer protocol) and FTPS (File transfer Protocol - Secure) protocols. It allows to easily transfer files between the local machine to the web hosting. Once the codes have been fully tested on the local machine, they will be uploaded directly to the web hosting Bluehost and will be available to the public. Bluehost offers the possibility to upload and download contents but is it not as efficient so it will not be used for the project.

6.1.2 Technologies

Front-end

Angular is a TypeScript-based open-source front-end web application platform lead by the Angular team at Google and by a community of individuals and corporations. Angular 5 has been used to develop the user interface by implementing components in TypeScript, HTML and CSS. Angular 5 gives a complete integration of bootstrap for responsive pages and it has a built-in server that can be run locally on the machine and every change made in the code can instantly be seen in the browser without even saving the new lines of codes.

Angular 5 has been chosen over other frameworks as it offers the following attributes for the user interface:

Consistency: The overall framework is based on components and services that interact with each other. Regardless of what component are written, the overall structure is always the same.

Reusability: Components of a similar nature are self-sufficient. Developers can reuse them across different section of the application.

Readability: New developers who have been recently joined the project can read the codes more easily and become quickly efficient because of the encapsulation.

***Productivity*:** Once you learn how to write a component or a service it is easy to follow the same pattern to write additional ones following the same guidelines and code structure.

Maintainability: Components are easily decoupled from each other and they can easily be replaced by improved components.

***Modularity*:** Angular 5 is about organising code into modules. It gives a way to organise application functionality and divide it up into features and reusable sections.

***Unit-test friendly*:** Angular 5 is built with testability in mind and various tools can be used to test the codes such as Karma, Jasmine and many others. Components are independent and they simplify the unit tests and quality assurance.

Back-end

The back-end will be completely independent from the front-end. Slim framework is a PHP micro framework that supports writing quick, simple but powerful REST API (representation state transfer - application programming interface). Slim is a dispatcher that receives HTTP requests, invokes the appropriate call-back routine and returns an HTTP response. Slim

framework holds routes that will respond to specific request such as GET, POST, PUT, DELETE, HEAD, PATCH and OPTIONS methods.

Slim framework supports the following main features:

Route matching: In old PHP implementation method, routes are often created by plain PHP files in public web directory. With slim a main single controller, which among other functionalities, specifies the routes of the application and maps them to right callable.

Dependency injection container: Slim App object comes with a built-in ($app->getContainer()) that can be leverage to define relationship between classes of the application.

Middleware support: Middleware is a code that can be hook between Slim and the callable. It accepts three arguments and can do whatever is appropriate with these objects.

Slim framework is designed specifically for fast development of APIs and web application. Symfony and Laravel have also been considered but they are much more complex frameworks. Since the client wants the product to be developed within a short time frame, Slim framework will provide the necessary functionalities and can be quickly implemented.

Database and Webhosting

Bluehost web hosting solution provides mainly a domain, an efficient and secure file management system with backup and a database. This solution is not free but will save considerable amount of time and help concentrate on the actual development of the project.

MySQL, Bluehost integration, a full-featured relational database management system (RDBMS) will be used to implement the database. MySQL is simple to use and scalable with a lot of support. Alternatively, Bluehost offers the possibility to use PostgreSQL. PhpMyAdmin, also part of the Bluehost integration, will handle the administration of MySQL. It is one of the most popular web interfaces for MySQL and it is a powerful and stable solution that has been used for years.

6.1.3 System Architecture

Overall Architecture

The Figure 6.1 shows the planned and defined architecture of the system agreed with the client. The graphical user interface is implemented with angular 5 and the integration of Bootstrap for responsiveness of the web pages. The back-end, front-end and MySQL database are hosted on Bluehost using a unique server. MailChimp API and SlackAPI will be used to send various types of email notifications to confirm the changes made by the users in the project management system.

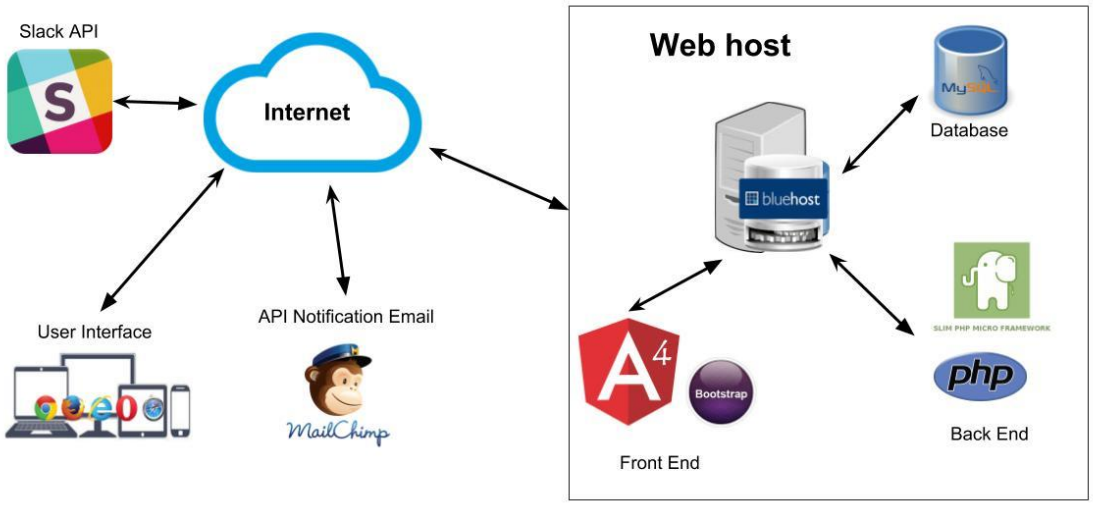


Figure 6.1 – System Overview

When a user is interacting with the graphical interface, a GET HTTP request will be executed when the administrator clicks on "View All Users" button shown in the Figure 6.2.

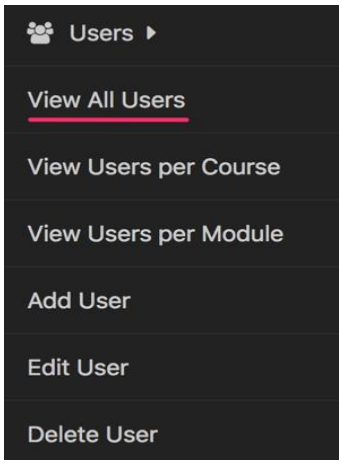


Figure 6.2 - User interface - Sidebar

A GET HTPP request will be directly made to the Slim API to retrieve the users of the system shown in figure 6.3.

```
@Injectable()
export class UserService {

  public isUserLoggedIn;
  public users;
  public currentUser;
  public userApi = 'http://slim.kingstonse.org/view/user';

  constructor(private http: HttpClient) {
    this.isUserLoggedIn = false;
  }

  getAllUsers(): Observable<any> {
    return this.http.get(this.userApi);

  }
```

Figure 6.3 - Angular 5 - HTTP GET request

Once the request has been made to the API, the controller will distribute the right route to retrieve the users of the system shown in Figure 6.4.

```php
<?php

header( string: 'Access-Control-Allow-Origin: *');
header( string: "Access-Control-Allow-Headers: Origin, X-Requested-With, Content-Type, Accept");
header( string: 'Access-Control-Allow-Methods: GET, POST, PUT');


require ('../../vendor/autoload.php');

$settings = ['settings'=> ['displayErrorDetails'=>true,],];

$app = new \Slim\App($settings);


require_once('../routes/login.php');
require_once('../routes/myname.php');
require_once ('../routes/dbconnect.php');
require_once ('../routes/user.php');
require_once('../routes/module.php');
require_once('../routes/organisation.php');
require_once('../routes/course.php');


$app->run();
```

Figure 6.4 - Slim routes

The function shown in Figure 6.5 will be called, a connection to the database will be made and the query will be executed.

```php
//view users

$app->get( pattern: '/view/user', function () {

    require_once('dbconnect.php');
    $connection = connect_db();
    $query = "SELECT * FROM `users`";
    $result = $connection->query($query);

    while ($row = $result->fetch_assoc()) {
        $data[] = $row;
    }
    if (isset($data)) {
        header( string: 'Content-Type: application/json');
        return json_encode($data);
    }
});
```

Figure 6.5 - Function - view users

The query will retrieve the rows from the users table shown in Figure 6.6.

Showing rows 0 - 5 (6 total, Query took 0.0006 seconds.)

```
SELECT * FROM users
```

Profiling [Inline] [Edit] [Explain SQL] [Create PHP Code] [Re

Show all | Number of rows: 25 Filter rows: Search this table

Sort by key: None

+ Options

	user_id	organisation_id	user_type_id	user_first_name	user_last_name	user_password	user_email	user_phone_number
Edit Copy Delete	1	1	1	Tony	mace	Tony	tony@kingston.ac.uk	07506959992
Edit Copy Delete	2	1	1	James	Robert	James	james@kingston.ac.uk	07547584848
Edit Copy Delete	3	1	2	Graeme	Jones	Graeme	graeme@kingston.ac.uk	07484838883
Edit Copy Delete	43	1	1	Radio	Head	Radiohead1234	radiohead@gmail.com	07506959991
Edit Copy Delete	44	1	2	Lecturer	Lecturer	Lecturer1234	lecturer@gmail.com	07506959994
Edit Copy Delete	45	1	2	Johnn	Smithh	Tony1234	tonymace@gmail.com	0750569499494

Figure 6.6 - Query the users

After the query is executed, the function shown in the Figure 6.6 will return a JSON object of all the users of the system shown in Figure 6.7 to the front-end as a response to the request.


```
slim.kingstonse.org/view/user
Apps  Previous project  Learn CSS Layout  Xdebug: Support; Ta...  apache - Xdebug on...  Update Mac Os X's...  Debugging with XDe...  Myapp  typescript - How do...  Simple Edit Form An...
[{"user_id":"1","organisation_id":"1","user_type_id":"1","user_first_name":"Tony","user_last_name":"mace","user_password":"Tony","user_email":"tony@kingston.ac.uk","user_phone_number":"07506959992","user_dep
am Tony from computer science","user_picture_path":null,"user_created_at":"2018-02-01 00:00:00","user_updated_at":null,"user_deleted_at":null,"user_date_of_birth":null},
{"user_id":"2","organisation_id":"1","user_type_id":"1","user_first_name":"James","user_last_name":"Robert","user_password":"James","user_email":"james@kingston.ac.uk","user_phone_number":"07547584848","user
am James from computer science","user_picture_path":null,"user_created_at":null,"user_updated_at":null,"user_deleted_at":null,"user_date_of_birth":null},
{"user_id":"3","organisation_id":"1","user_type_id":"2","user_first_name":"Graeme","user_last_name":"Jones","user_password":"Graeme","user_email":"graeme@kingston.ac.uk","user_phone_number":"07484838883","
am Graeme jones computer science and art","user_picture_path":null,"user_created_at":null,"user_updated_at":null,"user_deleted_at":null,"user_date_of_birth":null},
{"user_id":"43","organisation_id":"1","user_type_id":"1","user_first_name":"Radio","user_last_name":"Head","user_password":"Radiohead1234","user_email":"radiohead@gmail.com","user_phone_number":"075069599
01-18 05:20:48","user_updated_at":null,"user_deleted_at":null,"user_date_of_birth":"1981-11-25 00:00:00"},
{"user_id":"44","organisation_id":"1","user_type_id":"2","user_first_name":"Lecturer","user_last_name":"Lecturer","user_password":"Lecturer1234","user_email":"lecturer@gmail.com","user_phone_number":"07506959
01-18 05:43:23","user_updated_at":null,"user_deleted_at":null,"user_date_of_birth":"1965-11-23 00:00:00"},
{"user_id":"45","organisation_id":"1","user_type_id":"2","user_first_name":"Johnn","user_last_name":"Smithh","user_password":"Tony1234","user_email":"tonymace@gmail.com","user_phone_number":"075056949949
01-31 12:07:33","user_updated_at":null,"user_deleted_at":null,"user_date_of_birth":"1981-12-24 00:00:00"}]
```


Figure 6.7 - JSON object - all users

The details of the users will be displayed in the HTML table shown in figure 6.8.

```
<table class="table table-responsive table-sm">
  <tr>
    <th>First name</th>
    <th>last name</th>
    <th>Password</th>
    <th>Email</th>
    <th>Phone number</th>
    <th>Department</th>
  </tr>

  <tr *ngFor = "let item of users">

    <td>{{item.user_first_name}}</td>
    <td>{{item.user_last_name}}</td>
    <td>{{item.user_password}}</td>
    <td>{{item.user_email}}</td>
    <td>{{item.user_phone_number}}</td>
    <td>{{item.user_department}}</td>
  </tr>
</table>
```

Figure 6.8 - HTML table

The administrator of the system will now be able to view the users in the graphical interface shown in Figure 6.9 (Graphical user Interface).

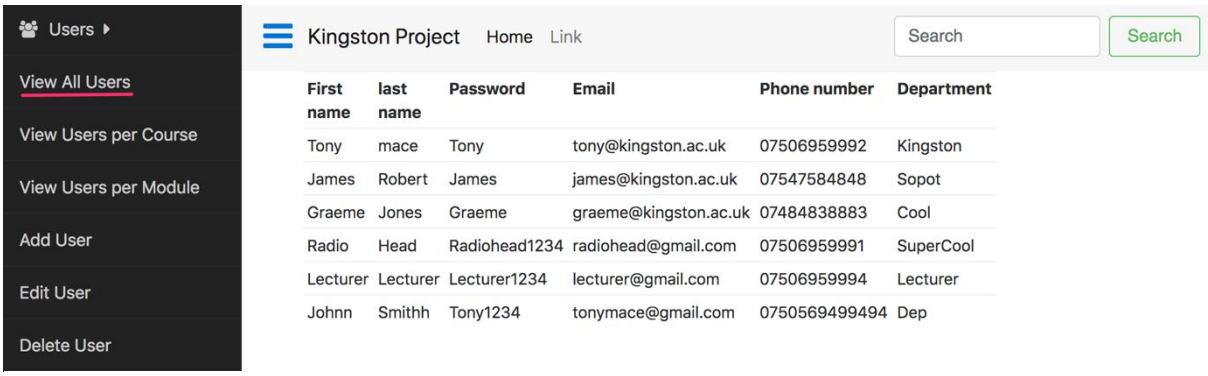


Figure 6.9 Graphical User interface

Front-end Architecture

Angular 5 is using a common architectural pattern Model-View-Controller. This pattern divides an application into three interconnected components roles. This model is represented in the Figure 6.10 by the file “course.service.ts” that is responsible for maintaining the states

and managing the data of the courses for the system. The model responds from the requests of the view as well as instructions made by the controller.

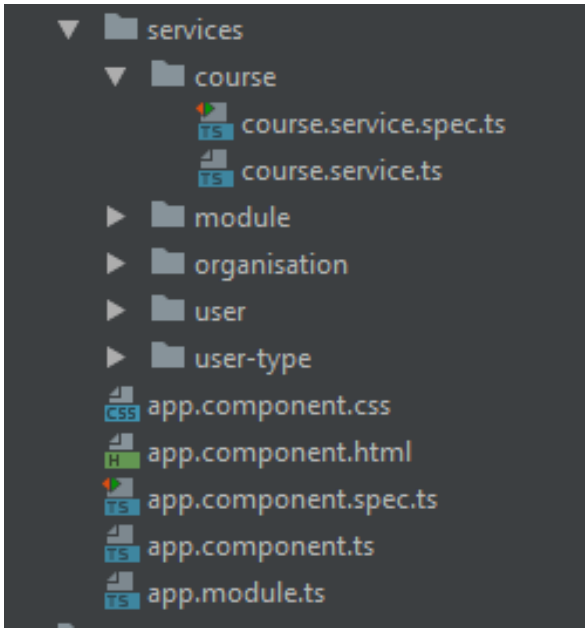


Figure 6.10 - Angular 5 Services

In Angular 5 the view and the controller are within the same components. The view is represented by the file “view-course.components.html” and the controller is represented by the file” view-course.components.ts” shown below in Figure 6.11. The view is responsible for displaying the graphical user interface to the user. The controller is responsible to handle the end user interaction, manipulating the model as well as choosing the view to be displayed to the user.

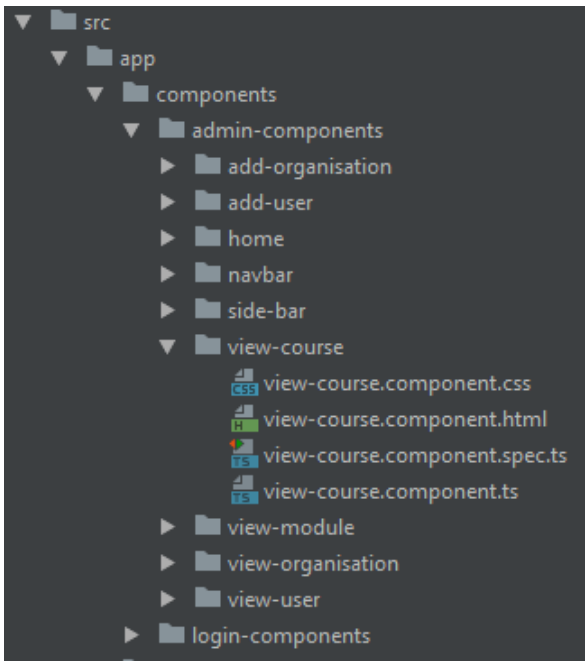


Figure 6.11 - Angular components

Back-end Architecture

Slim is a super lightweight framework, ideal for rapid development. It will be used as a REST API. REST differs from SOAP (Simple Object Access Protocol) because it is based on resources and actions, rather than methods and data types. A resource is just the URL referencing the object on which the action is performed – for example, /view/course. An action is one of the four HTTP verbs:

- Get - Retrieve
- Post - Create
- Put - Update
- Delete - Remove

The requests made to the API are directed to the "index.php" file which acts as a controller, in a publicly accessed folder shown below in Figure 6.12. The controller will, depending on the request, redirect to the right file.

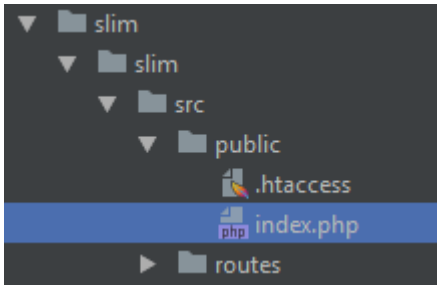


Figure 6.12 - Slim controller

The routes are stored in an independent folder called "routes" shown in the Figure 6.13. The folder and the files contained within the folder are only accessed privately.

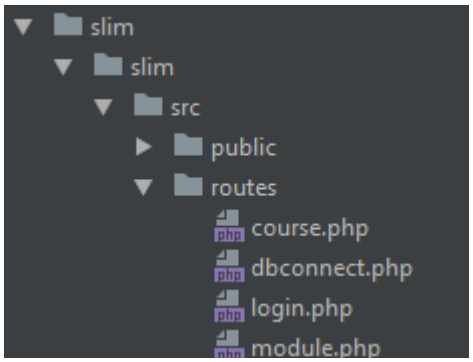


Figure 6.13 - Slim routes

Depending on the request the right function will be called. As an example, if the "GET request http://slim.kingstonse.org/view/course" is made, the function shown below in Figure 6.14 will be called within the file.

```
$app->get( pattern: '/view/course',function(){

    require_once('dbconnect.php');

    $connection = connect_db();

    $query = "SELECT * FROM course";

    $result = $connection->query($query);

    while($row = $result->fetch_assoc()){
        $data[] = $row;
    }
    if (isset($data)){
        header( string: 'Content-Type: application/json');
        return json_encode($data);
```

Figure 6.14 - Function view courses

6.1.4 Conclusion

It is essential to choose the appropriate tools and technologies, and to have a clear understanding of the architecture of the system before developing the application. The front-end and back-end are completely independent from each other and they enforce maintainability and reusability of the system. Once it has been validated by the client, the database, heart of the system, will be implemented in first instance.

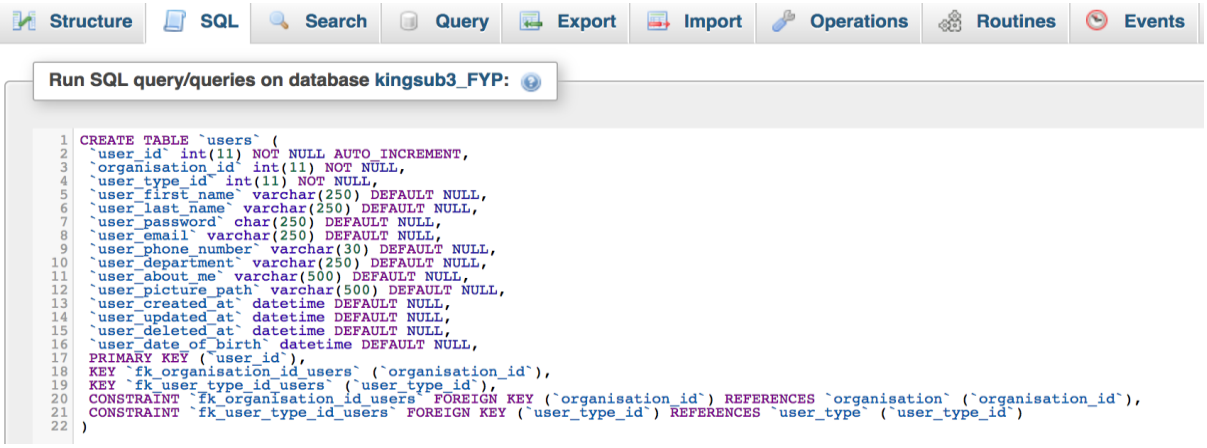
6.2 Database Implementation

6.2.1 Introduction

MySQL database has been chosen over many other solutions has it is open source, reliable, compatible with all major hosting providers, cost effective, and easy to manage. PhpMyAdmin also a free software, will be used to handle the administration of MySQL over the web. XAMPP will allow to build and manage the database locally to test the efficiency and the accuracy of the database before to be exported on Bluehost web hosting. The tables and constraints need to be implemented and the database need to be populated to validate the design.

6.2.2 Creating tables and constraints

The tables have been created using phpMyAdmin web interface and the "CREATE TABLE statement show in the Figure 6.15. During this process the constraints are generated such as Primary Keys, Foreign Keys, mandatory fields as well as the datatype for each field. It is important to consider appropriate datatype size during the creation of the tables to save space, improve cache efficiency and the performance of the database.




```
CREATE TABLE `users` (
 `user_id` int(11) NOT NULL AUTO_INCREMENT,
 `organisation_id` int(11) NOT NULL,
 `user_type_id` int(11) NOT NULL,
 `user_first_name` varchar(250) DEFAULT NULL,
 `user_last_name` varchar(250) DEFAULT NULL,
 `user_password` char(250) DEFAULT NULL,
 `user_email` varchar(250) DEFAULT NULL,
 `user_phone_number` varchar(30) DEFAULT NULL,
 `user_department` varchar(250) DEFAULT NULL,
 `user_about_me` varchar(500) DEFAULT NULL,
 `user_picture_path` varchar(500) DEFAULT NULL,
 `user_created_at` datetime DEFAULT NULL,
 `user_updated_at` datetime DEFAULT NULL,
 `user_deleted_at` datetime DEFAULT NULL,
 `user_date_of_birth` datetime DEFAULT NULL,
 PRIMARY KEY (`user_id`),
 KEY `fk_organisation_id_users` (`organisation_id`),
 KEY `fk_user_type_id_users` (`user_type_id`),
 CONSTRAINT `fk_organisation_id_users` FOREIGN KEY (`organisation_id`) REFERENCES `organisation` (`organisation_id`),
 CONSTRAINT `fk_user_type_id_users` FOREIGN KEY (`user_type_id`) REFERENCES `user_type` (`user_type_id`)
)
```

Figure 6.15 Create table statement for users table

In the Figure 5.16, the twenty tables and the relationships extracted from the ERD diagram during the design are now implemented.

Structure | SQL | Search | Query | Export | Import | Operations | Routines | Events | Triggers | Designer

Table	Action	Rows	Type	Collation	Size	Overhead
assignee	Browse Structure Search Insert Empty Drop	6	InnoDB	latin1_swedish_ci	16 KiB	-
content	Browse Structure Search Insert Empty Drop	4	InnoDB	latin1_swedish_ci	32 KiB	-
course	Browse Structure Search Insert Empty Drop	3	InnoDB	latin1_swedish_ci	16 KiB	-
goal	Browse Structure Search Insert Empty Drop	6	InnoDB	latin1_swedish_ci	32 KiB	-
module	Browse Structure Search Insert Empty Drop	8	InnoDB	latin1_swedish_ci	16 KiB	-
module_course	Browse Structure Search Insert Empty Drop	8	InnoDB	latin1_swedish_ci	48 KiB	-
notification	Browse Structure Search Insert Empty Drop	0	InnoDB	latin1_swedish_ci	16 KiB	-
objective	Browse Structure Search Insert Empty Drop	6	InnoDB	latin1_swedish_ci	32 KiB	-
organisation	Browse Structure Search Insert Empty Drop	2	InnoDB	latin1_swedish_ci	16 KiB	-
project	Browse Structure Search Insert Empty Drop	2	InnoDB	latin1_swedish_ci	32 KiB	-
project_comment	Browse Structure Search Insert Empty Drop	6	InnoDB	latin1_swedish_ci	32 KiB	-
task	Browse Structure Search Insert Empty Drop	6	InnoDB	latin1_swedish_ci	64 KiB	-
task_comment	Browse Structure Search Insert Empty Drop	4	InnoDB	latin1_swedish_ci	32 KiB	-
team	Browse Structure Search Insert Empty Drop	2	InnoDB	latin1_swedish_ci	16 KiB	-
users	Browse Structure Search Insert Empty Drop	6	InnoDB	latin1_swedish_ci	48 KiB	-
users_course	Browse Structure Search Insert Empty Drop	8	InnoDB	latin1_swedish_ci	48 KiB	-
users_module	Browse Structure Search Insert Empty Drop	7	MyISAM	latin1_swedish_ci	3.1 KiB	-
users_team	Browse Structure Search Insert Empty Drop	2	InnoDB	latin1_swedish_ci	48 KiB	-
user_type	Browse Structure Search Insert Empty Drop	3	InnoDB	latin1_swedish_ci	16 KiB	-
workflow_step	Browse Structure Search Insert Empty Drop	8	InnoDB	latin1_swedish_ci	32 KiB	-
20 tables	Sum	97	MyISAM	latin1_swedish_ci	595.1 KiB	0 B

Figure 6.16 - Complete database structure

6.2.3 Eliciting Data requirements

It is important to have relevant data to populate and test the database. A meeting has been agreed with the client to elicit the key information about the data that will be held in each table. This process was essential to agree the naming convention of the tables and the fields, as well as the number of row that needed to be inserted in each table. By inserting large amount of data in multiple tables, it will be possible to verify the efficiency of the database with the most complicated queries. The constraints that have been discussed with the client are as follow:

- A project should not be deleted but only archived.
- A task should not be deleted but only archived.
- Tasks and subtasks have exactly the same data type.
- Twenty thousand students are present over four campuses.
- The university teaches one hundred fifty different courses.
- Each course may have between three to five modules.
- Only lecturers, students and admin will be interacting with the project management software.

6.2.4 Methods in populating data

The database will be entirely populated using random data. To ease this extensive process many tools are available to generate data and Mockaroo shown in Figure 6.17 is one of them.

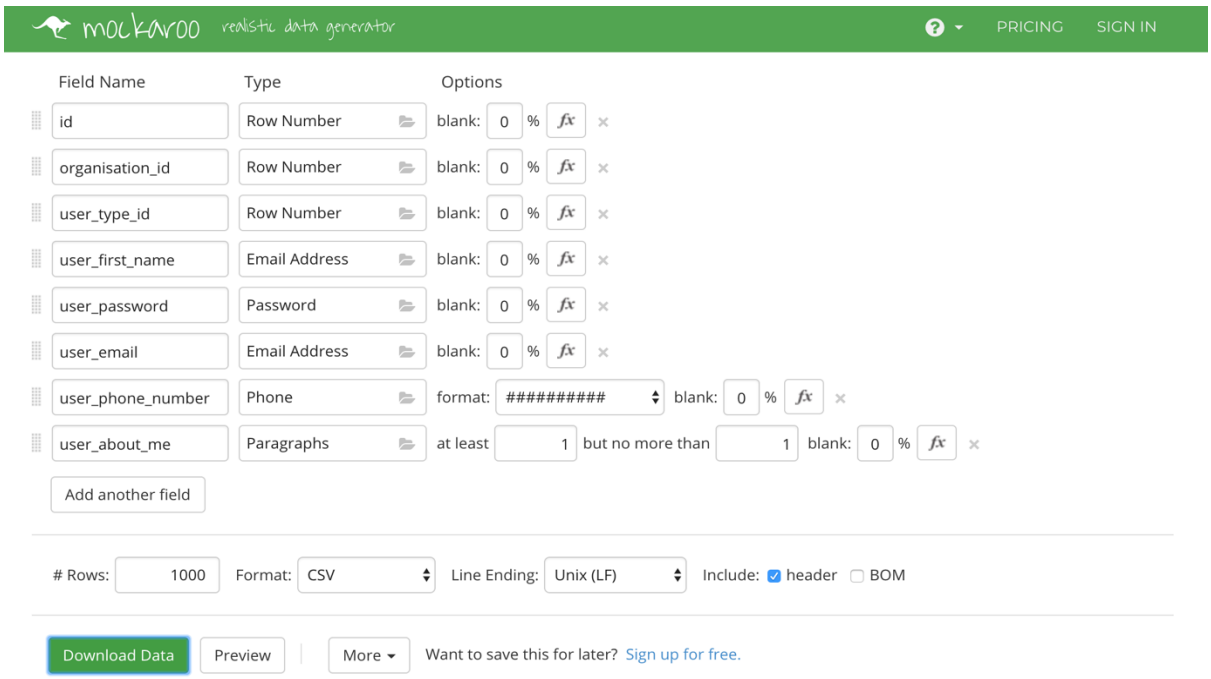


Figure 6.17 - Mockaroo data generator

Only the name and the type of the fields need to be entered to generate thousands of rows exported as a csv file shown in the Figure 6.18. When the csv file has been generated it can be imported into the database using the web interface phpMyAdmin to insert automatically the relevant data. Once the database has been fully tested and validated the random data will be cleared.

organisation	user_type_id	user_first_na	user_passwo	user_email	user_phone_	user_about_me
1	1	jzoephel0@r	l0q6FaupKgC	abullers0@d	2066446794	Mauris enim leo, rhoncus sed, vestibulum sit amet, cursus id, turpis. Integer aliquet, massa id lobortis convallis
2	2	gson1@pen.	gFCNVVA	tpeachey1@	2639001779	Praesent blandit. Nam nulla. Integer pede justo, lacinia eget, tincidunt eget, tempus vel, pede.
3	3	acrispe2@th	JKwkJUFXD2	hmactrustrie	8071200458	Sed sagittis. Nam congue, risus semper porta volutpat, quam pede lobortis ligula, sit amet eleifend pede libero
4	4	mwife3@ucl	Y0KkxP	grobbings3@	3181668069	Phasellus in felis. Donec semper sapien a libero. Nam dui.
5	5	dbiner4@ove	xOJo0Mqp	btruscott4@	1331839553	In sagittis dui vel nisl. Duis ac nibh. Fusce lacus purus, aliquet at, feugiat non, pretium quis, lectus.
6	6	fglossup5@s	XEqYoc	tgreated5@a	3772507817	Praesent blandit. Nam nulla. Integer pede justo, lacinia eget, tincidunt eget, tempus vel, pede.
7	7	mpardal6@u	OtBfbZ	jalelsandrow	2107783471	Aenean fermentum. Donec ut mauris eget massa tempor convallis. Nulla neque libero, convallis eget, eleifend
8	8	abandt7@up	A2H5LAjVFCI	lpaggitt7@e	3751526159	Phasellus in felis. Donec semper sapien a libero. Nam dui.
9	9	lskatcher8@	VuLcBjJ7M3c	bheningham	9147351169	Mauris enim leo, rhoncus sed, vestibulum sit amet, cursus id, turpis. Integer aliquet, massa id lobortis convallis
10	10	aeagell9@hc	dhGueHZQzA	mdickin9@tr	6273746581	Duis bibendum, felis sed interdum venenatis, turpis enim blandit mi, in porttitor pede justo eu massa. Donec d
11	11	jmundalla@t	bKda25v67	whendrona@	7577071661	Donec diam neque, vestibulum eget, vulputate ut, ultrices vel, augue. Vestibulum ante ipsum primis in faucibu
12	12	tedmondson	ssqztCH8q0q	cgowansonb	4005436856	In quis justo. Maecenas rhoncus aliquam lacus. Morbi quis tortor id nulla ultrices aliquet.
13	13	amynottc@a	3NoWofrMD	wgradyc@ec	7826895430	Nullam porttitor lacus at turpis. Donec posuere metus vitae ipsum. Aliquam non mauris.
14	14	rredingtond@	ZAaMsdom	lhutchcraftd(	9078429788	Cras non velit nec nisi vulputate nonummy. Maecenas tincidunt lacus at velit. Vivamus vel nulla eget eros elem
15	15	dbenedicke@	17UNQwIGo	fmarcume@	8717656888	In quis justo. Maecenas rhoncus aliquam lacus. Morbi quis tortor id nulla ultrices aliquet.
16	16	jklimochkinf(	R3MpSUejPF	vcowdreyf@	7641006607	Vestibulum quam sapien, varius ut, blandit non, interdum in, ante. Vestibulum ante ipsum primis in faucibus o
17	17	sarkowg@oc	kHgzysOdZB	bbayliieg@ab	4787092342	Sed ante. Vivamus tortor. Duis mattis egestas metus.
18	18	nobeneyh@b	XL7vyuXOJG7	mpriddeyh@	6442845780	Quisque porta volutpat erat. Quisque erat eros, viverra eget, congue eget, semper rutrum, nulla. Nunc purus.
19	19	cmainwaring	wjc5dmCSIZI	aricarti@see	3132532337	Vestibulum ac est lacinia nisi venenatis tristique. Fusce congue, diam id ornare imperdiet, sapien urna pretium
20	20	nsawkinsj@u	OBK8oKhH	bcawdellj@e	7378359052	Nam ultrices, libero non mattis pulvinar, nulla pede ullamcorper augue, a suscipit nulla elit ac nulla. Sed vel en
21	21	rkachellerk@	9OGgPNOUF	clindlk@ibm	8891075424	Duis consequat dui nec nisi volutpat eleifend. Donec ut dolor. Morbi vel lectus in quam fringilla rhoncus.
22	22	ecabanel@n	e1I7c7yb	mpaulsonl@	2487887025	Suspendisse potenti. In eleifend quam a odio. In hac habitasse platea dictumst.
23	23	docriganem@	2Ws5gsVnZ	mbrownettm	9869927831	Maecenas ut massa quis augue luctus tincidunt. Nulla mollis molestie lorem. Quisque ut erat.
24	24	tdonneelyn@	aslESIz3iVd	slawfuln@gc	5617801292	Nulla ut erat id mauris vulputate elementum. Nullam varius. Nulla facilisi.
25	25	estapfortho@	tuiJkeq	bdicarlio@ye	9357725991	Curabitur at ipsum ac tellus semper interdum. Mauris ullamcorper purus sit amet nulla. Quisque arcu libero, ru
26	26	dhaulkhampi	qePwXvSM	ekeilingp@b	7766927813	In quis justo. Maecenas rhoncus aliquam lacus. Morbi quis tortor id nulla ultrices aliquet.
27	27	bitzhayekq@	2GPDbYfdvY	ichildesq@d	8509664262	Aenean fermentum. Donec ut mauris eget massa tempor convallis. Nulla neque libero, convallis eget, eleifend
28	28	ctooleyr@16	2M0DI2wxSF	nivanishevr@	4831712764	Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Proin risus. Praesent lectus.
29	29	geasthams@	1CAyDUp	abellchambe	5442353618	Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Vivamus vestibulum sa
30	30	sgowant@cc	e41YVJyb	gwaret@arti	4591098321	In hac habitasse platea dictumst. Etiam faucibus cursus urna. Ut tellus.

Figure 6.18 - Random data in csv file

6.2.5 Validation of populated data

It is essential to verify that the data has been imported correctly into the database. PhpMyAdmin would give a sensible feedback once the operation is successfully completed as shown in Figure 6.19.

Figure 6.19 Importing data

To ensure that the data are present in the table, it is possible to count the number of rows as shown in Figure 6.20 and compare them with the number of rows imported previously in the table.

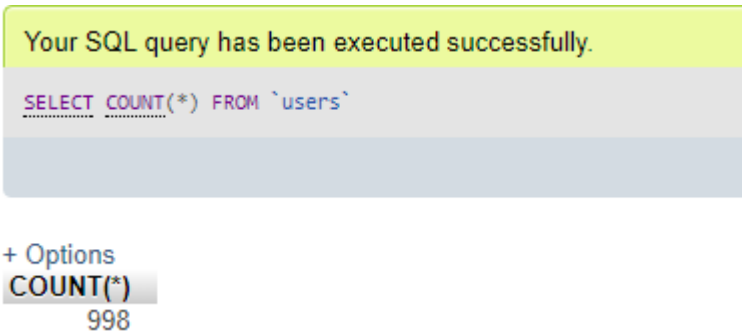


Figure 6.20 - Query the rows

6.2.6 Querying the database

Once the data have been imported into the database, it is important to consider the most complex queries of the project management system (see Appendix L). Some queries may need to join more than five tables to retrieve the relevant data of the system. Many users will be using the system, so the queries need to be optimised to improve the response time of the request made by the API. As a result, it will also improve the user experience because the database will be able to process the data in a more efficient way and give a quick response. To improve the efficiency of the queries, subqueries will be avoided and only join queries will be used during the implementation of the API as join queries usually execute faster. This section will be also revisited and expanded within the implementation of the API.

6.2.7 Conclusion

The database has now been implemented and populated with random data. The entities, relationships and constraints have been clearly identified and implemented. Some changes may be required during the implementation of the front-end, back-end and the testing phase. The client may also require new changes and further iterations will be needed to accommodate the changes as well as constantly improving the efficiency of the database. The database is now efficient and scalable. The frontend and the back-end can now be implemented.

6.3 Back-end implementation

6.3.1 Introduction

Once the database has been implemented, the back-end will be developed using slim framework. Slim is a PHP micro framework that helps writing quickly, simple and powerful web applications and APIs. It is a dispatcher that receives HTTP request and invoke the suitable callback routine and returns an HTTP response. Slim will be used as a REST API. A REST API is designed to use existing protocols and can be used over nearly any protocol although it usually operates with HTTP when used for Web APIS. Unlike SOAP, REST is not constrained to XML, but can also return XAML, JSON, YAML or any other format depending on the client requirements.

The client has agreed that the responses from the API should be returned in JSON objects because of the following reasons:

- JSON requires less tags than other formats.
- JSON is transportation-independent.
- Methods can also be part of a JSON object.
- JSON is easier to read than another format.

The API will be responsible for retrieving the data from the database depending on the request made when a user is interacting with the graphical interface

6.3.2 Setting up Slim framework

Slim framework required a web server in order to be installed. Apache is fast, reliable, secure, open source and one of the most widely used web server software. Apache will be directly installed as part of the XAMPP solution on the local machine and it is also available on Bluehost webhost for production.

The following steps are necessary to install slim framework.

- Install XAMPP including Apache.
- Installing slim with composer. Composer is a tool to manage PHP dependencies. It allows to declare the libraries the project will depend on and install and update them automatically.
- Local Apache configuration.(.htaccess)
- Production Apache configuration.(.htaccess)
- The steps describe above can be found in the Appendix M in more details.

It is important to take the time to configure Apache web server. The .htaccess file is a configuration file that is executed by the web server when located in the root directory. The file can be used to alter the configuration of the web server to enable and disable functionalities and features. It is used for basic redirection functionality or more advanced functions such as content password protection.

```
# Use PHP5.6 as default
AddHandler application/x-httpd-php56 .php

<IfModule mod_rewrite.c>

RewriteCond %{HTTP_HOST} ^slim\.kingstonse\.org$ [NC]
RewriteRule ^((?!slim/src/public/).*)$ /slim/src/public/$1 [L,NC]

RewriteEngine On
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^(.*)$ /index.php [QSA]

</IfModule>
```

Figure 6.21 .htaccess file

The figure 6.21 represents the .htaccess file that has been edited on the webhost. Starting from **<ifModule mod_rewrite.c>,** this line is present to protect the user from getting a 500 error if the server does not have mod_rewrite enabled. If this line is not currently present the message shown in the Figure 6.22 will be displayed when making a request to the API.

Internal Server Error

The server encountered an internal error or misconfiguration and was unable to complete your request.

Please contact the server administrator, webmaster@slim.kingstonse.org and inform them of the time the error occurred, and anything you might have done that may have caused the error.

More information about this error may be available in the server error log.

Additionally, a 500 Internal Server Error error was encountered while trying to use an ErrorDocument to handle the request.

Apache Server at slim.kingstonse.org Port 80

Figure 6.22 Internal error 500

The following two lines
RewriteCond %{HTTP_HOST} ^slim\.kingstonse\.org$ [NC]

RewriteRule ^((?!slim/src/public/).*)$ /slim/src/public/$1 [L,NC]

are present to redirect the request from the subdomain slim.kingstonse.org directly to the public folder of the slim framework that contains the index.html file. The RewriteEngine On will enable the rewrite engine. The following two lines

- **RewriteCond %{REQUEST_FILENAME} !-f**
- **RewriteCond %{REQUEST_FILENAME} !-d**

Will pass the requests not referring directly to files in the filesystem to index.php
RewriteRule ^(.*)$ /index.php [QSA]

6.3.3 Routes

The Slim framework's router has been built on top of the Nikic/fastroute component. Nikic/fastroute is a library that gives a fast implementation of a regular expression based on router. It is quick and stable. The application routes can be define using proxy methods on the \Slim\App instance. Slim gives methods for the most common methods. A route can be added as show in the Figure 6.23 and will handle only GET HTTP requests with the method get (). It takes two arguments which are the route pattern and the route callback. Routes can also handle POST, PUT, DELETE, OPTIONS, PATH and ANY HTTP requests. It is also possible to handle multiple type of request for the same route. In the Figure 6.23, the callback $response (HTTP response) is overwritten by a customised response "This course does not have any module" when the query does not return any rows. This response is directly returned to the frontend when no modules where found for a specific course.

```
$app->get( pattern: '/return/specific[/{id:.*}]', function ($request, $response, $args) {

    $id = explode( delimiter: '/', $request->getAttribute('id'));

    $ids = implode( glue: ',', $id);

    require_once('dbconnect.php');

    $connection = connect_db();

    $query = "SELECT module.* FROM module
              INNER JOIN module_course
              ON module.module_id = module_course.module_id
              INNER JOIN course
              ON module_course.course_id = course.course_id
              WHERE course.course_id IN ($ids)";

    $result = $connection->query($query);

    while ($row = $result->fetch_assoc()) {

        $data[] = $row;
    }
    if (isset($data)) {
        header( string: 'Content-Type: application/json');
        return json_encode($data);
    } else {

        header( string: 'Content-Type: application/json');
        $response = 'This course does not have any module';
        return $obj = json_encode($response);
    }
});
```

Figure 6.23 – Route

6.3.4 Request

The slim app's route is given a PSR 7 request object that represents the current HTTP request that has been made by the system. The request object implements the PSR 7 ServerRequestInterface with which it can inspect and manipulate the HTTP request method, headers and body. The PSR 7 request object is injected into the slim application routes as the first argument to the route callback shown in the Figure 6.24.

```
$app->get( pattern: '/view/user/{id}', function (ServerRequestInterface $request,ResponseInterface $response) {

    require_once('dbconnect.php');

    $response = json_encode( value: 'No users for this module');


    $connection = connect_db();
    $id = $request->getAttribute( name: 'id');
```

Figure 6.24 Request object

The HTTP request has a body. The application consumes JSON objects and it is possible to retrieve the data using the PSR 7 request object's getParsedBody ( ) method to parse the HTTP request body into a native PHP format as shown in Figure 6.25.

```
$course_name = $request->getParsedBody()['courseName'];
$course_description = $request->getParsedBody()['courseDescription'];
```

Figure 6.25 getParsedMethod

6.3.5 Response

The slim app’s routes are given a PSR 7 response object that represents the current HTTP response to be returned to the client. The response object implements the PSR 7 ReponseInterface with which it can inspect and manipulate the HTTP response status, headers

and body. The PSR 7 response object is injected into the slim application routes as the second argument to the route callback shown in the Figure 6.25. The response is overridden by a custom response when the query does not return any row.

```
$app->get( pattern: '/view/user/{id}', function (ServerRequestInterface $request,ResponseInterface $response) {

    require_once('dbconnect.php');

    $response = json_encode( value: 'No users for this module');


    $connection = connect_db();
    $id = $request->getAttribute( name: 'id');

    $query = "SELECT * FROM users
              INNER JOIN users_course
              ON users.user_id = users_course.user_id
              INNER JOIN course
              ON users_course.course_id = course.course_id
              WHERE course.course_id = $id";

    $result = $connection->query($query);
    while ($row = $result->fetch_assoc()) {
        $data[] = $row;
    }
    if (isset($data)) {
        header( string: 'Content-Type: application/json');
        return json_encode($data);
    } else {

        return $response;
```

Figure 6.25 Response object

6.3.6 Conclusion

The API is now implemented and the request made by the user interacting with the graphical interface will return the right data as a JSON object. It is now easy to modify and expand the API and, query and returned the appropriate data to the front-end. Once the back-end has been implemented, various test will be necessary using postman to ensure that the data returned are proper.

6.4 Front-end Implementation

6.4.1 Introduction and Design Patten

The front-end is entirely implemented using Angular 5. Angular 5 offer a full integration of bootstrap to build a responsive website and improve the user experience. Angular 5 is using the MVC pattern. The model, the view and the controller are low coupled and favour code reuse which can reduce the implementation time. The code is also easier to maintain and changes can be easily made. The **Model** is the central component of the pattern and expresses the application's behaviour. It directly manages the data, the logic and the rules of the application. The **View** is a visual representation of the outputs to the user. The **Controller** accepts the inputs and outputs and converts them into commands for the model and the view.

6.4.2 Setting up Angular 5

To start the implementation of the front-end using angular, it is necessary to set up the development environment. Node.js and Npm must be installed on the local machine before to install Angular client. Node and Npm will be installed using the command "brew install node". The Node.js and Npm version can be verified using the command "node –v" and "npm –v". Node.js is a server that allows Angular to run locally on the machine to constantly test and verify the progress of the implementation. Npm is a package manager for Node.js packages. Once Node.js and Npm are installed it is now time to install Angular client with the following command "npm install –g @angular/cli". Angular client is a command line interface tool that allow you to create an Angular project, add files, and perform a variety of ongoing development tasks such as testing, bundling and deployment. After installing Angular client, the command

“ng new my-project” is used to generate a new project and skeleton application. When the project structure is generated, the node_modules folder shown in figure 6.26 is also created and has the packages that may be necessary to build the application. Some of the packages may be removed later if they are not used.

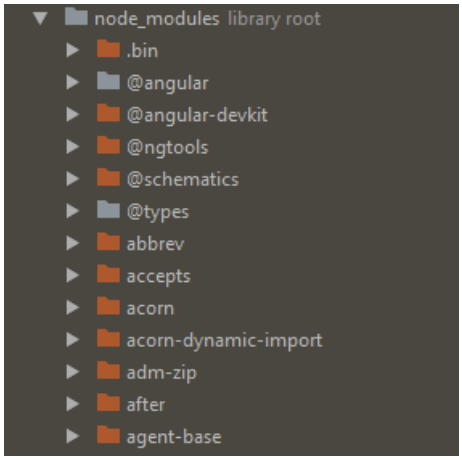


Figure 6.26 - Node modules

The command “ng serve” will be used to launch the server. This command will launch the server, watch the files and rebuilds the app every time changes are made to the codes.

6.4.3 Bootstrap Integration

The decision was made with the client to use bootstrap to implement a responsive application and improve the user experience. To integrate bootstrap in an Angular project it is necessary to first install the bootstrap package with the command “npm install boostrap” and to edit the configuration file “. angular-cli.json”. Only the following lines shown in Figure 6.27 were added to Angular to fully integrate bootstrap after installing the package.

```
"styles": [
  "../node_modules/bootstrap/dist/css/bootstrap.min.css",
  "../node_modules/font-awesome/css/font-awesome.css",
  "styles.css"
],
"scripts": [
  "../node_modules/jquery/dist/jquery.slim.min.js",
  "../node_modules/tether/dist/js/tether.min.js",
  "../node_modules/popper.js/dist/umd/popper.min.js",
  "../node_modules/bootstrap/dist/js/bootstrap.min.js",
  "../node_modules/jquery/dist/jquery.min.js"
```

Figure 6.27 - Bootstrap setup

Many of the components used by Bootstrap require the use of JavaScript to function. Specifically, they require jQuery, Popper.js and Tether.js and need to be imported with bootstrap.min.js.

6.4.4 Component

Angular is using components to build an application. They are basic building blocks of the application as shown in Figure 6.28. Each component defines the following:

- Any necessary imports needed by the component.
- A component decorator, which includes the properties that allow to define the template, CSS styling and animations.
- A class, which is where the component logic is stored.

Figure 6.28 - Angular component

The file “add-user.component.spec.ts” shown in the Figure 6.28 is a unit test for the source files. By convention for Angular applications, every TS file will have a spec.ts file. They are run using the Jasmine JavaScript test framework through the Karma task runner. By default, Angular client just includes the “app.component.ts” file. Component can be generated using the command “ng generate component add-user”. A single page can be implemented using multiple components depending on the functionality that is needed. Every single component is low coupled and it becomes simple to refractor the codes or make changes without affecting the entire application.

6.4.5 Page structure

It is important to mention that angular do not require to use the html and body tags to build an html page. The html and body tags are automatically generated by the Angular application when the production app is build. As shown in Figure 6.29, the content of a classic html page which normally sit inside the body tags is directly inserted in the html file. The “<app-side-bar>” and “</app-side-bar>” are the opening and closing tags for the top and side navigation bar component. Only these two tags are necessary to include the two navigation bars on the pages. The side navigation bar component includes the top navigation bar component but can easily be decoupled if necessary.

```
<app-side-bar></app-side-bar>

<div class="container">

  <form [formGroup]="userForm" (ngSubmit)="onSubmit(userForm.value)" novalidate>


    <div class="form-group row">

      <label for="userType" class="col-2 col-form-label">UserType</label>

      <div class="col-10">

        <!--[ngClass]="{-->

        <!--'has-danger': userForm.controls.userType.invalid && userForm.controls.userType.untouched,-->
        <!--'has-success': userForm.controls.userType.valid && userForm.controls.userType.dirty-->
        <!--}"><-->

        <select #userType class="form-control" formControlName="userType" (change)="checkUserType(userType.value)"
                type="text" id="userType" required>

          <option *ngFor="let userType of userTypes" [value]="userType.user_type_id">

            {{userType.user_type_role}}

          </option>

        </select>

      </div>

    </div>


    <div formGroupName="courseModule">

      <div *ngIf="userTypeChange == 1    " class="form-group row">

        <label for="courseStudent" class="col-2 col-form-label">Course</label>

        <div class="col-10">

          <select class="form-control" formArrayName="course" (change)="sendCourseStudent($event)" type="text"
                  id="courseStudent" required>

            <option value="undefined" disabled>Please, select the module</option>

            <option *ngFor="let course of courses" [value]="course.course_id"> {{course.course_name}}</option>
```

Figure 6.29 - Page structure

It is also normally necessary to have a reference to an external style sheet file directly in the HTML page if CSS style need to be applied. It is usually executed with the following:

<link rel="stylesheet" type="text/css" href="mystyle.css">.

In Angular the CSS file is automatically generated and linked for each component that has been created, in the TS file as shown in Figure 5.30.

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

Figure 6.30 - CSS reference

The structure of the html file is minimalist because most of the logic is present in the TS file as mentioned previously.

6.4.6 Expandable Menu

Now that bootstrap has been setup and the main structure of the pages is completed, the expandable sidebar can be implemented as shown in Figure 6.31. It will be possible for the user to show or collapse the side navigation bar by clicking the “burger menu” as well as expanding the menu by clicking the main menus (e.g. Users).

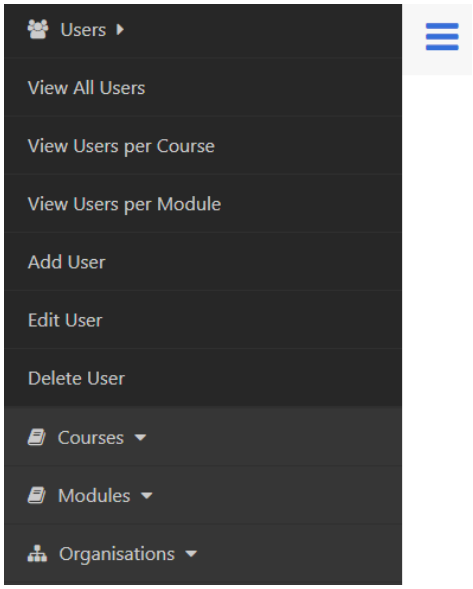


Figure 6.31 – Expandable sidebar

This expandable menu was achieved by using the data-toggle attribute shown in Figure 6.32 at the first line. When the link is actually clicked the <div class=" collapse" id=" menu1> that contains the submenus will be either displayed or hidden. When the user is clicking the "Users" menu shown in Figure 6.31, he will be able to see the submenus for this section.

```
<a href="#menu1" class="list-group-item d-inline-block collapsed" data-toggle="collapse" data-parent="#sidebar"
   aria-expanded="false"><i class="fa fa-users">
</i> <span class="d-none d-md-inline">Users</span> </a>
<div class="collapse" id="menu1">
  <a routerLink="/view/user" class="list-group-item" data-parent="#menu1">View All Users</a>
  <a href="view/usercourse" class="list-group-item" data-parent="#menu1">View Users per Course</a>
  <a href="#" class="list-group-item" data-parent="#menu1">View Users per Module</a>
  <a routerLink="/add/user" class="list-group-item" data-parent="#menu1">Add User</a>
  <a href="edit/user" class="list-group-item" data-parent="#menu1">Edit User</a>
  <a href="#" class="list-group-item" data-parent="#menu1">Delete User</a>
</div>
```
Figure 6.32 – Sidebar html

It is important for the user to have the possibility to collapse the sidebar when necessary to be able to leave more space and clearly view the full content of the page. Good feedback was given by the tester to ensure the implementation of this functionality to improve the user experience.

6.4.7 Service

In some instance it is necessary to reuse the same code across multiple components. Data may also need to be shared between components or data may need to be fetch from the database. In Angular a service is a function that allows to access it's define properties as well as methods. This is also a good practise to keep the code organised. To create a service, the command line "ng generate service user" need to be applied. The service also need to be added to the providers property of the NgModule decorator found in src/app/app.module.ts, as shown in the Figure 6.33. They are added to the root scope and they will be injected to the required component when necessary.

```
@NgModule({
  declarations: [
    AppComponent, FooterComponent, HeaderComponent, LoginFormComponent, HomeComponent, NavbarComponent,
    AddUserComponent, ViewUserComponent, SideBarComponent, ViewOrganisationComponent, ViewModuleComponent, ViewCourseComponent,
    AddOrganisationComponent, AddModuleComponent, AddCourseComponent, ViewUserCourseComponent, EditUserComponent
  ],
  imports: [
    BrowserModule,
    RouterModule.forRoot(appRoutes),
    FormsModule,
    HttpClientModule,
    AngularFontAwesomeModule,
    MalihuScrollbarModule.forRoot(),
    ReactiveFormsModule
  ],
  providers: [UserService, OrganisationService, ModuleService, UserTypeService, CourseService, AuthenticationGuard],
  bootstrap: [AppComponent]
```

Figure 6.33 - NgModule

The generic function getAllUsers() present in the UserService call the API and return the users (JSON object) of the system as shown in Figure 6.34.

```
import {Injectable} from '@angular/core';
import {Observable} from "rxjs/Observable";
import {HttpClient} from "@angular/common/http";

@Injectable()
export class UserService {

  public isUserLoggedIn;
  public users;
  public currentUser;
  public userApi = 'http://slim.kingstonse.org/view/user';

  constructor(private http: HttpClient) {
    this.isUserLoggedIn = false;
  }
  getAllUsers(): Observable<any> {
    return this.http.get(this.userApi);
  }
```

Figure 6.34 - UserService

The function getAllUsers can be reused across any other components when required. As an example, it is necessary to retrieve the users in the view-user.component.ts in order for the administrator to view the details of the users. In order to do so, it is necessary to import the UserService at the top of the component code as shown in Figure 6.35.

```
import {UserService} from '../../../services/user/user.service';
```

Figure 6.35 - Import UserService

Once it has been imported, the UserService should also be imported as a dependency injection in order to use it as shown in Figure 6.36.

```
constructor(private router: Router, private user: UserService) {
```

Figure 6.36 - Constructor View-user

Now the UserService can be used to access it's associated properties and methods. The method getAllUsers () in Figure 6.37 can then be called inside the User-view.component.ts using the UserService and the data can be retrieved and used.

```
retrieveUsers() {

  this.user.getAllUsers().subscribe( next: data => {
    this.users = data;
```

Figure 6.37 - Call the method in UserService

This process can be repeated for any other componenents to reuse various services and it's properties and methods. By doing so, duplicated codes are avoided which also ensure that if a bug is present it would not be spread over multiple components.

6.4.8 Form

Forms will be used mainly for the administrators, to setup the project management software. This is an essential part of the system. The inputs will need to be accurate and it is necessary to go through validation when the administrator will submit the forms. The following needs to be added to the database by completing the forms in order for the system to be fully setup:

- Organisation to the system
- Users to the system
- Courses to the system
- Modules to the system

It is an essential development phase of the system before to introduce the implementation of the dashboard for the users. When developing a form, it is important to create a data-entry experience that will guide the users through the workflow. The form "Add a user" was certainly the most difficult form and required complex implementation. Before starting implementing the form it was essential to confirm and clarify certain points witht the client. The following was agreed:

- A Lecturer can be assigned to one or mulitple courses and one or multiple modules from mulitple courses..
- A student can only be assigned to one course but not necessary all the modules belonging to this course.
- The date of birth should not allow users to be under sixteen.
- A password confirmation field should be present for the password input.
- All the fields within the form should be mandatory.

In Angular 5, the model for the form is composed of instances of FormGroups, FormArrays and FormControls as shown in Figure 6.38 . FormGroups can be nested in FormGroups. The userForm is an instance of a FormGroup class and this reprensent the form itself. A FormControl is necessary for each template form control present in the add-user.component.html. The formGroup courseModule is composed of two FormArrays course and module. The FormArrays for modules and courses can expend or contract depending on the number of modules and courses assigned to secturers and students.

```
  this.userForm = this.formBuilder.group( controlsConfig: {
    userType: new FormControl(),
    organisation: new FormControl(),
    firstName: new FormControl( formState: '', Validators.pattern( pattern: '[a-zA-Z]+')), // in
    lastName: new FormControl( formState: '', Validators.pattern( pattern: '[a-zA-Z]+')),// inpu
    dateOfBirth: new FormControl(),
    courseModule: this.formBuilder.group( controlsConfig: {
      course: this.formBuilder.array( controlsConfig: [], Validators.required),
      module: this.formBuilder.array( controlsConfig: [], Validators.required),
    }),
    password: this.formBuilder.group( controlsConfig: {
      passwordInput: new FormControl( formState: '', Validators.pattern( pattern: '(?=.*\\d)(?=.
      passwordConfirm: new FormControl( formState: '', Validators.pattern( pattern: '(?=.*\\d)(?
    },  extra: {validator: this.passwordMatchValidator}),
    email: new FormControl( formState: '', Validators.email),
    phoneNumber: new FormControl( formState: '', Validators.pattern( pattern: '^[0-9()-]+$')),
    department: new FormControl()
  })
};

passwordMatchValidator(password: FormGroup) {

  return password.get('passwordInput').value === password.get('passwordConfirm').value
    ? null : {'mismatch': true};

}
```

Figure 6.38 - UserForm

Once the form has been structured it is important to link the form model to the form template. The <form> element present in the add-user.component.html page need to be bind to the top level property userForm using the formGroup directive as shown in Figure 6.39.

```
<form [formGroup]="userForm" (ngSubmit)="onSubmit(userForm.value)" novalidate>
```

Figure 6.39 - Html Form

It is also necessary to map each form control using the formControlName or formArrayName directive as shown in Figure 6.40 for each control.

```
<select #userType class="form-control" formControlName="userType" (change)="checkUserType(userType.value)"
        type="text" id="userType" required>
```

Figure 6.40 - FormControlName

The value property of the userForm model returns the values of the controls as an object. The tag shown below in Figure 6.41 added at the bottom of the form in the add-user.component.html page is used to output debug information about the form.

```
<pre>Form value: {{ userForm.value | json }}</pre>
```

Figure 6.41 - Pre-Tag

Now that each form control in the template is mapped to the form controls in the model, when the user select or enter an input, the form values are updated in real time.The Figure 6.42 show the state of the userForm returned values for adding an admnistrator before and after completion of the form.

UserType
Organisation
First Name
Last Name
Date of Birth dd/mm/yyyy
Password
Password Confirmation
Email
Phone number
Department
Submit

```
Form value: {
  "userType": null,
  "organisation": null,
  "firstName": "",
  "lastName": "",
  "dateOfBirth": "",
  "courseModule": {
    "course": [],
    "module": []
  },
  "password": {
    "passwordInput": "",
    "passwordConfirm": ""
  },
  "email": "",
  "phoneNumber": "",
  "department": ""
}
```

UserType Admin
Organisation Kingston University
First Name Tony
Last Name Mace
Date of Birth 25/11/1981
Password ••••••••
Password Confirmation ••••••••
Email tonymace1981@gmail.com
Phone number 07555555555
Department department
Submit

```
Form value: {
  "userType": "3",
  "organisation": "1",
  "firstName": "Tony",
  "lastName": "Mace",
  "dateOfBirth": "1981-11-25",
  "password": {
    "passwordInput": "Test1234",
    "passwordConfirm": "Test1234"
  },
  "email": "tonymace1981@gmail.com",
  "phoneNumber": "07555555555",
  "department": "department "
}
```

Figure 6.42 - UserForm before and after completion

The form should readjust depending on the type of user that need to be added to the system without even refreshing the page. When no user type is selected, the form will be as shown in figure 6.43. This form is by default set to add an administrator to the system.

UserType
Organisation
First Name
Last Name
Date of Birth dd/mm/yyyy
Password
Password Confirmation
Email
Phone number
Department

Figure 6.43 - Blank Form

The dropdown userType has been populated from the row user_type_role in the table user_type. The data from the user_type table have been retrieved as a JSON object (Figure 5.44) by calling the API as shown in the Figure 6.45.

[{"user_type_id":"1","user_type_role":"Student"},
{"user_type_id":"2","user_type_role":"Lecturer"},
{"user_type_id":"3","user_type_role":"Admin"}]

Figure 6.44 JSON object user type

```
retrieveUserTypes() {
  this.userTypeService.getAllUserTypes().subscribe( next: data => {
    this.userTypes = data;
```

Figure 6.45 - API call user type

Once the user types have been retrieved, the dropdown in the html page add-user-component.html can be populated. The select tag and the option tags are used to generate the dropdown menu. In Angular 5 the *ngFor directive is used to generate a loop. As shown in the Figure 6.46, “let userType of userTypes” will loop through the object userTypes and display only the user_type_role for each element in the object.

```
<div class="container">

  <form [formGroup]="userForm" (ngSubmit)="onSubmit(userForm.value)" novalidate>


    <div class="form-group row">

      <label for="userType" class="col-2 col-form-label">UserType</label>

      <div class="col-10">

        <select #userType class="form-control" formControlName="userType" (change)="checkUserType(userType.value)"
                type="text" id="userType" required>

          <option *ngFor="let userType of userTypes" [value]="userType.user_type_id">

            {{userType.user_type_role}}

          </option>

        </select>

      </div>

    </div>
```

Figure 6.46 - ngFor

The dropdown organisation has been populated from the row organisation_name from the table organisation. The data from the organisation table were retrieve as a JSON object (Figure 6.47) by calling the API as shown in the Figure 6.48.

[{"organisation_id":"1","organisation_name":"Kingston University","organisation_address":"Penrhyn Rd
2EE","organisation_type":"University","organisation_phone_number":"+447506959991","organisation_c
Kingdom","organisation_created_at":null,"organisation_updated_at":null,"organisation_deleted_at":null}
{"organisation_id":"52","organisation_name":"Tony","organisation_address":"53 rue de la
pomme","organisation_type":"University","organisation_phone_number":"+444344343434","organisatio
06:28:42","organisation_updated_at":null,"organisation_deleted_at":null}]

Figure 6.47 - JSON object organisation

```
retrieveOrganisations() {
  this.organisationService.getOrganisations().subscribe( next: data => {
    this.organisations = data;
```

Figure 6.48 - Call API organisation

As soon as the user type “Student” is selected the dropdown “Course” shown in Figure 6.49 is added to the form. The administrator of the system can now select the appropriate course for the student.

Figure 6.49 Dropdown course

In Angular 5, it is possible to enable or disable a form control when needed. This is achieved by doing the following code in Figure 6.50. When the user type admin is selected the course and module form will be disabled and when the user type lecturer or student is selected, it will be enabled.

```
this.userForm.controls['courseModule'].enable();
this.userForm.controls['courseModule'].disable();
```

Figure 6.50 - Enable and disable FormControl

When the courseModule form group (course and module controlForm) has been enabled it is necessary to display the html dropdown “Course”. To do so, the system should keep track of the userType that has been selected in the dropdown userType, by instantiating a variable as shown in Figure 6.51.

```
userTypeChange: number = 0;
```

Figure 6.51 - Variable userTypeChange

The event (change) will trigger the function checkUserType with the value of the user_type_id shown in Figure 6.52 when a the selection is triggered.

- The user_type_id for student is 1 by default.
- The user_type_id for lecturer is 2 by default.
- The user_type_id for admin is 3 by default.

```
<select #userType class="form-control" formControlName="userType" (change)="checkUserType(userType.value)"
        type="text" id="userType" required>

  <option *ngFor="let userType of userTypes" [value]="userType.user_type_id">

    {{userType.user_type_role}}
```

Figure 6.52 - UserTypeId value

The function present in the add-user.component.ts is called and the value is passed to the function. Depending on the user_type_id the userTypeChange will be set to the appropriate value and the courseModule form will be enabled or disabled as shown in Figure 6.53.

```
checkUserType(userType) {

  //console.log('userTypeID: ' + userType);

  if (userType == 1) {

    this.userTypeChange = 1;
    this.resetFormControlCourse();
    this.resetFormControlModule();
    this.modulesMatchingCourses = null;
    this.userForm.controls['courseModule'].enable();

  } else if (userType == 2) {

    this.userTypeChange = 2;
    this.resetFormControlCourse();
    this.resetFormControlModule();
    this.modulesMatchingCourses = null;
    this.userForm.controls['courseModule'].enable();

  } else {

    this.userTypeChange = 3;
    this.resetFormControlCourse();
    this.resetFormControlModule();
    this.modulesMatchingCourses = null;
    this.userForm.controls['courseModule'].disable();

  }
}
```

Figure 6.53 - CheckUserType function

In Angular 5, the ngIf directive is used to show or hide the div depending on the condition shown in the Figure 6.54. In this case, if the userTypeChange is equal to one the div class =”form-group row” will be displayed otherwise it will remain hidden.

```
<div formGroupName="courseModule">

  <div *ngIf="userTypeChange == 1   " class="form-group row">

    <label for="courseStudent" class="col-2 col-form-label">Course</label>

    <div class="col-10">

      <select class="form-control" formArrayName="course" (change)="sendCourseStudent($event)" type="text"
              id="courseStudent" required>

        <option value="undefined" disabled>Please, select the course</option>

        <option *ngFor="let course of courses" [value]="course.course_id"> {{course.course_name}}</option>

      </select>

    </div>

  </div>
```

Figure 6.54 - Div hidden

When the course has been selected, the event (change) will trigger the function sendCourseStudent with the $event parametre as shown in Figure 6.55 .

```
<div *ngIf="userTypeChange == 1   " class="form-group row">
  <label for="courseStudent" class="col-2 col-form-label">Course</label>
  <div class="col-10">
    <select class="form-control" formArrayName="course" (change)="sendCourseStudent($event)" type="text"
            id="courseStudent" required>
```

Figure 6.55 - Function sendCourseStudent

The function sendCourseStudent, shown in Figure 6.56, present in the add-user.component.ts is called and the course_id is passed to the function. The resetFormControlCourse and resetFormControlModule are called in order to reset the FormArray that could have been set for previously. A new FormControl is now added to the course.

```
sendCourseStudent(event) {

  this.resetFormControlCourse();
  this.resetFormControlModule();
  const courseControl = <FormArray>this.userForm.controls['courseModule'].get('course');
  courseControl.push(new FormControl(event.target.value, Validators.required));
  var courseID = this.userForm.get('courseModule').value.course;
  this.retrieveModuleFromCourse(courseID);
}
```

Figure 6.56 - Function sendCourseStudent

It is possible to verify in the JSON object that the course id has been added to the formControl as shown in Figure 6.57. The function retrieveModuleFromCourse is now called with the course_id.

```
"courseModule": {
  "course": [
    "1"
  ],
  "module": []
},
```

Figure 6.57 - Course FormControl value

The retrieveModuleFormCourse method shown in Figure 6.58 will call the API with the course_id in order to retrieve the modules for this course. The modulesMatchingCourses variable will contain the modules.

```
retrieveModuleFromCourse(courseIds) {
  return this.http.get( url: this.apiUrl2 + '/' + courseIds).subscribe( next: object => {
    this.selectionDataJson = object;
    this.modulesMatchingCourses = this.selectionDataJson;
  });
}
```

Figure 6.58 - Function retrieveModuleFromCourse

Now that the modules are returned by the API, they need to be populated in the html page using the *ngFor directive as shown in Figure 6.59 to display them.

```
<div *ngIf="userTypeChange == 1  || userTypeChange == 2" class="form-check form-group row">
  <div class="form-check" *ngFor="let module of modulesMatchingCourses">
    <label class="form-check-label" for="{{module.module_name}}">
      <input class="form-check-input" formArrayName="module" (change)="selectModule($event,module.module_id)"
             [value]="module.module_name"
             type="checkbox"
             name="module"
             id="{{module.module_name}}">
      {{module.module_name}}
```

Figure 6.59 - *ngFor Directive

The user is now able to select the modules that he wants to assign to the student as shown in Figure 6.60.

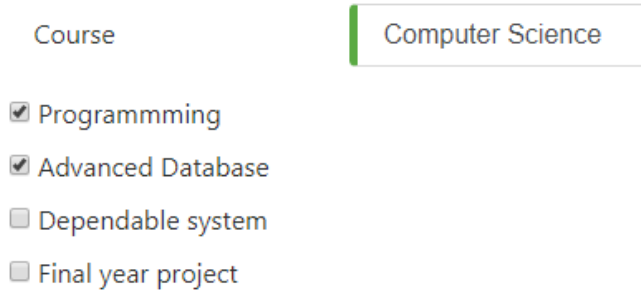


Figure 6.60 - Checkboxes modules

When the modules have been selected, the event will trigger the function selectModule with the $event and module_id parametre as shown in Figure 6.61. If the checkbox has been selected, the event.target is checked and a formControl will be added to the the formArray module. If the checkbox is then unchecked, the formControl will be removed from the formArray module.

```
selectModule(event, module_id: number) {
  const moduleControl = <FormArray>this.userForm.controls['courseModule'].get('module');
  if (event.target.checked) {
    // Add a new control in the arrayForm
    moduleControl.push(new FormControl(module_id));
  } else {
    // find the unselected element
    let i: number = 0;
    moduleControl.controls.forEach( callbackfn: (ctrl: FormControl) => {
      if (ctrl.value == event.target.value) {
        // Remove the unselected element from the arrayForm
        moduleControl.removeAt(i);
        return;
      }
      i++;
    });
  }
```

Figure 6.61 - Function selectModule

It is possible to verify in the JSON object that the module ids have been added to the formArray module as shown in Figure 6.62.

```
"courseModule": {
  "course": [
    "1"
  ],
  "module": [
    "1",
    "2"
  ]
```

Figure 6.62 - JSON object modules

The button submit is only available when the form has been completed. The attribute [disabled] shown in Figure 6.63 allow to control when a given input element should be enabled or disabled. The userForm is invalid until the FormControl are validated. When it is validated, the userForm.invalid becomes false and the button submit is now available.

```
<button type="submit" class="btn btn-success" [disabled]="userForm.invalid">Submit</button>
```

```
<button _ngcontent-c1 class="btn btn-success" type="submit" disabled>Submit</button>
```

Figure 6.63 - Submit button

When the user click the button submit, the function onSubmit shown in Figure 6.64 is called and an HTTP POST request is sent to the API to insert the user into the database. The form is complety reseted to add a new user if necessary.

```
onSubmit = function (dataForm) {
  if (this.userForm.valid) {
    this.http.post(this.addUserApi, JSON.stringify(dataForm),
      {
        headers: {
          'Accept': 'application/ json',
          'Content-Type': 'application/json'
        }
      }).subscribe
    (data => {
      //reset the form after submission
      this.userForm.reset();
      this.retrieveUsers();
      this.userForm.controls['courseModule'].disable();
      this.resetFormControlCourse();
      this.resetFormControlModule();
      this.modulesMatchingCourses = null;
      //print out the data return by the server
      console.log(data);
    }, err => {
      console.log(err);
      console.error("Could not be added");
    });

  }
};
```

Figure 6.64 - Form submission

6.5 Security Concerns

6.5.1 SQL injection

SQL injection is a code injection attack that can execute malicious SQL statement (also referred as a malicious payload) which attackers can use to gain access to resources or make changes to the data present in the database. They can also be used to drop the tables and retrieve sensitive data. It is typically used on web forms for user authentication or simple inputs. Several methods can be used to avoid the SQL injection.

The back-end was previously implemented using mysql_query as shown in Figure 6.65. The main arguments why mysql_query should not be used can be found in the Appendix N.

```
$app->get( pattern: '/view/usertype', function () {

    require_once('dbconnect.php');
    $connection = connect_db();
    $query = "SELECT * FROM user_type";
    $result = $connection->query($query);

    while ($row = $result->fetch_assoc()) {
        $data[] = $row;
    }
    if (isset($data)) {
        header( string: 'Content-Type: application/json');
        return json_encode($data);
    }
});
```

Figure 6.65 – Backend with MySQL

The back-end SQL will be refactored using PHP Data Objects(PDO) as shown in Figure 6.66 to ensure SQL injection will be avoided by using prepared statement and bound parameters.

```
$app->get( pattern: '/view/usertype', function () {

    require_once('dbconnect.php');
    $connection = connect_db();
    $query = $connection->prepare( query: "SELECT * FROM user_type");
    $query->execute();
    $result = $query->fetchAll(\PDO::FETCH_ASSOC);

    if (isset($result)) {
        header( string: 'Content-Type: application/json');
        return json_encode($result);
    }
});
```

Figure 6.66 – Backend with PDO

6.6 Conclusion

The implementation chapter is now completed and resume the main phases executed to build the SPMS. It was important to point out the different steps achieved to setup the local environment before deploying the working software on the webhost. Testing the code locally allows the developer to save time by avoiding deploying constantly the code to production to ensure it is fully working. Only part of the functional and non-functional requirements was implemented and by using the Agile methodology it will allow to introduce more iterations, to complete the project.

PDO has been introduced late during this process but it is essential to protect the data against SQL injection and consider one of the essential principal dependability properties which is security. The users do not want their personal data to be accessed by hackers.

An efficient and responsive interface has been implemented to ensure a good user experience. The navigation sidebar is fully responsive and can easily be hidden to view the full page especially on small desktop screen.

Decoupling the front end and back-end allows the developer to easily refractor the codes when changes are needed by introducing new requirements from the client.

Angular 5 is a relatively new technology and documentation is not always available so more time was needed to research and understand complex functionalities. It could have been avoided by using only PHP for the frontend. Slim framework on the other hand was relatively easy to implement but the structure following an MVC pattern can be improved over time.

7 Testing

Testing is intended to show that a program does what is it intended to do but also to discover program defects before it is put into use. In Agile, testing is not a separate phase, but an integral part of the software development, along with coding. Different methodologies can be applied to test the software. This chapter will focus mainly on the following testing methodologies:

- Functional testing
- Non-functional testing
- Acceptance testing
- Usability testing

During the testing phases, verification and validation should be considered to check that the software meets the specifications and that it also fulfils its purpose. Verification is about knowing if we are building the product right so the system should conform to its specification. Validation is about knowing if the product has been implemented correctly so that the system should do what the user really requires.

7.1 Functional testing

Functional testing also known as black-box testing is a software testing method in which the internal structure, design and implementation of the system is unknown for the user. The tests can be functional and non-functional but usually focuses more on functional testing. The advantages of black-box testing are that it does not required tester to be technical, it allows to verify contradictions in the actual system and the specifications, and the test cases can be designed as soon as the functional specifications are completed. Black-box testing focuses on the functionality of the system and it allows to find errors in the following categories:

- Incorrect or missing functions
- Interface errors
- Errors in data structures or external database access
- Behaviour or performance errors
- Initialisation and termination errors

Test cases

The test cases in table 7.1 were built around specifications and requirements of the system. This will prove how the functional testing was executed during the project. Each test will be referenced by its own identification id and the functional requirement it is associated to. The tester needs also to be able to see the expected result when the scenario has been completed. The requirements implemented need to be carefully tested using various inputs and different flows to ensure the correct outcome. The table 7.1 only proves test cases for a single requirement. The remaining cases can be found in the Appendix O.

ID Test	FR ID	SCENARIO	EXPECTED RESULT	RESULT
1	5	Users enter Department with unexpected characters (e.g. numbers)	The field should not be validated	PASS
2	5	Users select the admin user type in the dropdown menu	The courses should become available	PASS
3	5	Users select a course from the dropdown menu	The modules should become available	PASS
4	5	Users select the organisation from the dropdown menu	The organisation field should be validated	PASS
5	5	Users enter their first name with unexpected characters (e.g. numbers)	The field first name should not be validated	PASS
6	5	Users enter their last name with unexpected characters (e.g. numbers)	The field last name should not be validated	PASS
7	5	Users enter their passwords that does not contain minimum of 8 characters, one number, one uppercase and one lowercase.	The field password should not be validated	PASS
8	5	Users enter their confirmation password that does not match password input	The field password confirmation should not be validated	PASS
9	5	Users enter an email address without a '@' character or a dot.	The field should not be validated	PASS
10	5	Users enter a phone number that contains a letter	The field should not be validated	PASS
11	5	Users click the submit button before completing the form	The button submit should only be available when the form is completed	PASS
12	5	Users click the submit button after completing the form	The user should get a sensible feedback when the user is added to the system	PASS
13	5	Users enter date of birth and is not over 16 years old	The field should not be validated	PASS

Table 7.1 – Functional Testing – Test cases

The functional testing was primarily used to ensure that the software provides the same output as required by the end-users and the client. It is essential to certify that all requirements have been tested and that the input fields should prevent the users from entering inappropriate data that could cause the system to fail. The forms were fully tested starting with the dropdown menus populated with data directly extracted from the database. Angular 5 is using a powerful built-in function to better control inputs validation and restrict users to only enter specific values. Pattern Validator directives were also used to control the user inputs by giving a regex attribute to the function**.** Using Validators will reinforce inputs validation.

7.2 Non-functional testing

Non-functional testing aims to test the aspects of the system that may not relate to a specific function or user action such as scalability or security. The purpose of non-functional testing is

to increase usability, efficiency, maintainability and portability of the system. It also helps reducing production risk and optimising the way the system is installed, setup, executed, managed and monitored. Non-functional testing is about testing the way system operates rather than specific behaviours of the system. This process can start after completing the functional testing.

Test cases

Test cases are also necessary to ensure that the non-functional requirements are covered. Several non-functional test cases were created as shown in table 7.2 and will be tested against the system. It is important to correctly state non-functional requirements because they will affect the user experience. The cases will be identified by an identification ID as well as the non-functional requirements type such as security. The non-functional test cases can be found in the Appendix P.

ID	NFR AREA	SCENARIO	EXPECTED RESULT	RESULT
1	Response time	Logging in to the system	The users should be able to login to the system within 2 seconds	Pass
2	Compatibility	Using different browser to access the system	The system should support all modern browser	Pass
3	Response time	Inserting data to the database	The database should be responsive and data should be inserted within a second	Pass
4	Availability	The system should be available 24/7 with short break for maintenance	The system is always available	Pass
5	Backup	A backup of the database is executed every week	The database should be backed up every week	Pass

Table 7.2 – Non-Functional Testing – Test cases

Thorough non-functional testing has a great influence on the customer and user satisfaction with the product. During this process it was possible to perceive how the application perform under normal circumstances as well as examining how well the system is secure. Non-functional testing is an important aspect of quality and requires a strategy and planning.

7.3 Usability testing

Usability testing is also a black-box testing technique that is used to evaluate how easy the system is to use and learn and how convenient the system is to the end users. It reveals whether the users feel comfortable with the system according to different parameters such as the flow, layout and navigation. The tests are executed by real users or customers to measure how intuitive and usable the software is. Usability testing is effective because it gives the opportunity to watch potential users of the system and see what works well and what would need further improvements. Listening to the needs and concerns of the end-users is essential during this process to ensure their satisfactions but also to save time on development and redesign efforts.

7.3.1 End-users tasks

To carry out the usability testing SPMS, it was necessary to recruit end-users (shown in Table 7.3) from Kingston University as the system will be directly intended for them. Each user will be able to get familiar with the system, test the main functionalities and can give feedback to improve the usability of the system. Users will also be observed during this process to see how

they interact with the system and discover where complications can occur. To gather further feedback, the users will take part in an interview after completing the tasks. The SPMS will only be used with modern browsers on desktop computers and usability testing will not apply to mobile device.

Name/surname	Age	Role	Computer literacy	Browser
Name withheld	18	Student	Intermediate	Chrome
Name withheld	25	Student	Basic	Safari
Name withheld	34	Lecturer	Proficient	Mozilla
Name withheld	45	Lecturer	Proficient	Opera

Table 7.3 - Testers

7.3.2 Interviews

The testers were asked several questions as shown in Table 7.4

1. Do you think that the design of the system is consistent?
2. Is it easy to interact with the system?
3. Is the system responsive for your browser?
4. Do you think that extra feedback should be added to the system?
5. Do you see any functionalities that would need improvement?

Name withheld	1. Yes, the layout is very consistent 2. I did not have any issues. 3. Yes, even when I would expand the navigation bar. 4. The forms may need some more feedback. 5. The icons in the navigation sidebar should be more relevant
Name withheld	1. Yes. 2. Yes. 3. Yes, but the two “Hamburger menus” can be confusing. 4. I was not sure why the password was not valid. 5. The colour of the navigation sidebar should not be so dark.
Name withheld	1. Yes, it feels that we are always on the same page. 2. Yes. 3. Yes, and it is quick to navigate through pages. 4. I do not always know what should be the valid input for the forms. 5. The navigation sidebar should remain present after selection.
Name withheld	1. Yes 2. Yes 3. Yes 4. Yes, some forms need more feedback. 5. Some alerts should be displayed with the right feedback.

Table 7.4 – Interview Questions

7.3.3 Conclusion

The usability testing has been conducted and the critical functionalities of the system operate as intended. Overall, the testers were satisfied with the product and could complete their tasks. Some changes were made to give more feedback to the users when completing the forms following the comments that were gathered from the users during the interviews. It was essential to conduct usability testing because a poor user experience will have a negative impact on the product. The product has met user’s expectations and matches the business decision to real-world use.

7.4 Unit testing – Angular 5

Unit testing also called isolated testing is a software testing method where individual units of a software are tested. Its purpose is to validate each unit of the software and ensure that the

functions are working as expected. Test should be executed periodically and after changes are made to the source code. Test cases can be automatically generated in some cases but they are usually written manually. Number of testing libraries and framework can be used to reduce the amount of time it takes to write tests. Angular 5 focused on Jasmine test framework and Karma test runner. Jasmine is a JavaScript testing framework that supports software development practice called behaviour driven development (BDD). Jasmine attempts to describe tests in human readable format so that non-technical people could understand it. Karma is the tool that lets us spawn browsers and run jasmine tests inside of them from the command line.

Angular 5 - Unit test

In angular 5, when a component is generated, a spec.ts file as shown in Figure 7.5 is also created so the components will have their own test file. The test files are executed using jasmine JavaScript test framework trough the karma task runner when the command 'ng test' is performed. A single test file can be target using the same command followed by the test file name (e.g ng test add-user.component.ts).

```
import ...

describe( description: 'AddUserComponent',  specDefinitions: () => {
  let component: AddUserComponent;
  let fixture: ComponentFixture<AddUserComponent>;

  beforeEach(async( fn: () => {
    TestBed.configureTestingModule( moduleDef: {
      declarations: [ AddUserComponent ]
    })
    .compileComponents();
  }));

  beforeEach( action: () => {
    fixture = TestBed.createComponent(AddUserComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it( expectation: 'should create',  assertion: () => {
    expect(component).toBeTruthy();
  });
});
```

Figure 7.5 – Test file

The command "ng test" builds the application in watch mode and launches the karma test runner as shown in Figure 7.6. The output in the web browser can be easier to read than the console log. Test can be re-run by simply clicking the test row or the description of the test.

```
Karma v1.7.1 - connected                                          DEBUG
Chrome 64.0.3282 (Mac OS X 10.13.3) is idle

Jasmine  2.6.4                                                finished in 0.853s
xxxxxxxxx•xxxxxx••xxxxxxx

25 specs, 22 failures                                         raise exceptions
Spec List | Failures

AppComponent should create the app

Failed: Template parse errors:
'router-outlet' is not a known element:
1. If 'router-outlet' is an Angular component, then verify that it is part of this module.
2. If 'router-outlet' is a Web Component then add 'CUSTOM_ELEMENTS_SCHEMA' to the '@NgModule.schemas' of this component to suppress this message. (" [ERROR ->]<router-outlet></rou

"): ng:///DynamicTestModule/AppComponent.html@0:1
```

Figure 7.6 Karma

Every time a change is made in the code the test will run again and new results will be refreshed automatically in the browser. Each of the spec.ts file have the tests grouped in describe block.

To execute only one specific test with angular-client it is important to prefix the describe function name with f. (e.g. fdescribe) as shown in Figure 7.7. One test has been executed to ensure user-service is properly implemented as shown in Figure 7.9. The test has been executed running the command ng test as show in Figure 7.8. Other test will be conducted to ensure that the frontend is fully tested with unit test cases.

```
import { TestBed, inject } from '@angular/core/testing';

import { UserService } from './/user.service';
import { HttpClientTestingModule} from "@angular/common/http/testing";

fdescribe( description: 'UserService',  specDefinitions: () => {
  beforeEach( action: () => {
    TestBed.configureTestingModule( moduleDef: {
      providers: [UserService],
      imports: [HttpClientTestingModule]
    });
  });

  it( expectation: 'should be created', inject( tokens: [UserService],  fn: (service: UserService) => {
    expect(service).toBeTruthy();
  }));
});
```

Figure 7.7 - User service test

```
C:\xampp\htdocs\project\myapp>ng test
 10% building modules 1/1 modules 0 active09 03 2018 16:37:37.374:WARN [karma]: No captured browser, open http://localhost:9876/
09 03 2018 16:37:37.390:INFO [karma]: Karma v1.7.1 server started at http://0.0.0.0:9876/
09 03 2018 16:37:37.390:INFO [launcher]: Launching browser Chrome with unlimited concurrency
09 03 2018 16:37:37.390:INFO [launcher]: Starting browser Chrome
aptured browser, open http://localhost:9876/
09 03 2018 16:37:53.463:INFO [Chrome 64.0.3282 (Windows 10 0.0.0)]: Connected on socket 16NBXFZgiGuagCpAAAAA with id 1502218
Chrome 64.0.3282 (Windows 10 0.0.0): Executed 1 of 25 (skipped 24) SUCCESS (0.132 secs / 0.05 secs)
```

Figure 7.8 - Ng test command

```
Jasmine  2.6.4

• • • • • • • • • • • • • • • • • • • • • • • •

Ran 1 of 25 specs - run all

1 spec, 0 failures

  UserService
    should be created
```

Figure 7.9 - Test result

Conclusion

Unit test should be written before starting to implement specific objects or methods. They allow the developers to constantly run test against their codes and ensure that the methods or objects are still working after making changes. Unit testing helps improves the quality of codes. During the agile development phase, many bugs can be discovered at an early stage to avoid late fixes that are costlier.

7.5 REST API Testing

The REST API has been implemented using the slim framework and the requests need to be tested to ensure that the functionalities are working as expected. API testing is used to determine whether the API will return the appropriate response from a broad range of requests and react as expected to edge cases. For methods are mainly involved in API testing which are GET, POST, DELETE and PUT.

- The GET method used to extract information from the server using given URL. Using a GET request should only retrieve data and have no other effects.
- The POST request used to generate a new entity. It can also be used to send the data to the server.
- The PUT request used to create or update an entity.
- The DELETE request will delete the resource identified by the request-URL.

Different tools are available to test a REST API such as advanced Rest Client, Postman-Rest Client or Curl in Linux. Postman will be used to test the REST API as it is free, and one of the most used and reliable tool now available. It is possible to organise and save the requests that can be reused later. Postman also supports variety of authentication and allows you to create tests using both GUI and JavaScript.

7.5.1 Postman GET request

The REST API should retrieve the users of the system when the GET request is executed with the following URL http://slim.kingstonse.org/view/user as shown in Figure 7.10. The data should be retrieve as a JSON object and sent back to the frontend when the request is functioning.

```
$app->get( pattern: '/view/user', function () {

    require_once('dbconnect.php');
    $connection = connect_db();
    $query = "SELECT * FROM `users`";
    $result = $connection->query($query);

    while ($row = $result->fetch_assoc()) {
        $data[] = $row;
    }
    if (isset($data)) {
        header( string: 'Content-Type: application/json');
        return json_encode($data);
```

Figure 7.10 - GET request

The request has been sent using the GET method to retrieve the users of the system. The status 200 OK as show in Figure 7.11 indicates that the request was fulfilled. The data have been returned by the REST API and can be used on the frontend.

GET http://slim.kingstonse.org/view/user Params Send Save

Body Cookies Headers (10) Test Results Status: 200 OK Time: 399 ms Size: 971 B

Pretty Raw Preview HTML Save Response


```
[{"user_id":"1","organisation_id":"1","user_type_id":"1","user_first_name":"Tony","user_last_name":"mace","user_password"
:"Tony","user_email":"tony@kingston.ac.uk","user_phone_number":"07506959992","user_department":"Kingston"
,"user_about_me":"I am Tony from computer science","user_picture_path":null,"user_created_at":"2018-02-01 00:00:00"
,"user_updated_at":null,"user_deleted_at":null,"user_date_of_birth":null},{"user_id":"2","organisation_id":"1"
,"user_type_id":"1","user_first_name":"James","user_last_name":"Robert","user_password":"James","user_email"
:"james@kingston.ac.uk","user_phone_number":"07547584848","user_department":"Sopot","user_about_me":"I am James from
computer science","user_picture_path":null,"user_created_at":null,"user_updated_at":null,"user_deleted_at":null
,"user_date_of_birth":null},{"user_id":"3","organisation_id":"1","user_type_id":"2","user_first_name":"Graeme"
,"user_last_name":"Jones","user_password":"Graeme","user_email":"graeme@kingston.ac.uk","user_phone_number"
:"07484838883","user_department":"Cool","user_about_me":"I am Graeme jones computer science and art","user_picture_path"
:null,"user_created_at":null,"user_updated_at":null,"user_deleted_at":null,"user_date_of_birth":null},{"user_id":"43"
,"organisation_id":"1","user_type_id":"1","user_first_name":"Radio","user_last_name":"Head","user_password"
:"Radiohead1234","user_email":"radiohead@gmail.com","user_phone_number":"07506959991","user_department":"SuperCool"
,"user_about_me":null,"user_picture_path":null,"user_created_at":"2018-01-18 05:20:48","user_updated_at":null
,"user_deleted_at":null,"user_date_of_birth":"1981-11-25 00:00:00"},{"user_id":"44","organisation_id":"1","user_type_id"
:"2","user_first_name":"Lecturer","user_last_name":"Lecturer","user_password":"Lecturer1234","user_email"
:"lecturer@gmail.com","user_phone_number":"07506959994","user_department":"Lecturer","user_about_me":null
,"user_picture_path":null,"user_created_at":"2018-01-18 05:43:23","user_updated_at":null,"user_deleted_at":null
,"user_date_of_birth":"1965-11-23 00:00:00"},{"user_id":"45","organisation_id":"1","user_type_id":"2","user_first_name"
:"Johnn","user_last_name":"Smithh","user_password":"Tony1234","user_email":"tonymace@gmail.com","user_phone_number"
:"0750569499494","user_department":"Dep","user_about_me":null,"user_picture_path":null,"user_created_at":"2018-01-31 12
:07:33","user_updated_at":null,"user_deleted_at":null,"user_date_of_birth":"1981-12-24 00:00:00"}]
```


Figure 7.11 – JSON object

7.5.2 Postman POST request

The REST API should create a new user of the system when the POST request is executed with the following URL http://slim.kingstonse.org/add/user as shown in Figure 7.12. A new row should be inserted in the users table and a response from the REST API should confirm that a user has been added to the system.

```
$app->post( pattern: '/add/user', function ($request, $response) {

    require_once('dbconnect.php');
    $connection = connect_db();

    $validation = json_encode( value: 'The user has been added to the system');
    $error = json_encode( value: 'The user could not be added to the system');

    $array = $request->getParsedBody();

    $user_type = $array['userType'];
```

Figure 7.12 – REQUEST URL

The request has been sent using the POST method to add a user to the system. To test the request when adding a new user to the system, it is necessary to encode the data in a JSON object and add it to the body that will be sent through the request as shown in Figure 7.13. The status 200 OK as shown in Figure 7.13 shows that the request was fulfilled. Additionally, the response from the server “The user has been added to the system” also confirm that the request has been fulfilled.

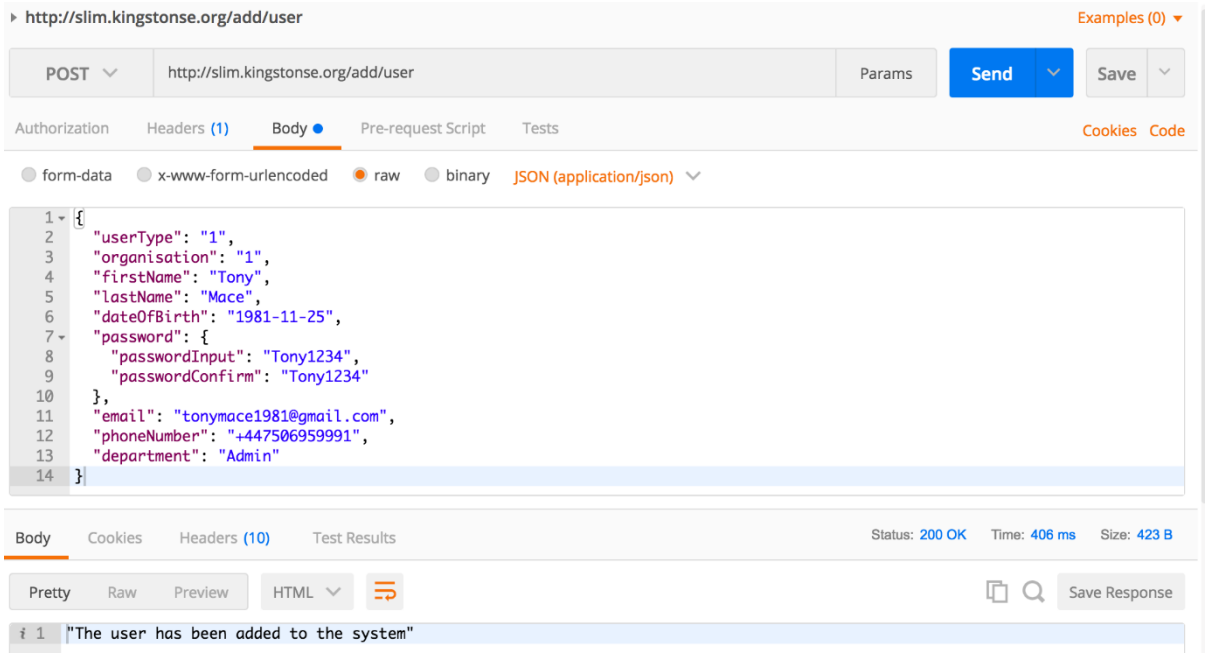


Figure 7.13 - Valid POST REQUEST

7.5.3 Postman PUT request

The REST API should update the details of a user of the system when the PUT request is executed with the following URL http://slim.kingstonse.org/edit/user as shown in Figure 7.14. The data should be retrieve as a JSON object and sent back to the frontend when the request is functioning.

```
$app->put( pattern: '/edit/user',function(ServerRequestInterface $request,ResponseInterface $response){

    require_once('dbconnect.php');
    $connection = connect_db();

    $validation = json_encode( value: 'The user has been updated');
    $error = json_encode( value: 'The user could not be update');

    $user_id = $request->getParsedBody()['userIdForm'];
    $user_first_name = $request->getParsedBody()['userFirstNameForm'];
    $user_last_name = $request->getParsedBody()['userLastNameForm'];
    $user_phone_number = $request->getParsedBody()['userPhoneNumberForm'];
    $user_Email = $request->getParsedBody()['userEmailForm'];
    $user_department = $request->getParsedBody()['userDepartmentForm'];

    $user_id = array_shift( &array: $user_id);
    $user_first_name = array_shift( &array: $user_first_name);
    $user_last_name = array_shift( &array: $user_last_name);
    $user_phone_number = array_shift( &array: $user_phone_number);
    $user_Email = array_shift( &array: $user_Email);
    $user_department = array_shift( &array: $user_department);
```

Figure 7.14 – PUT REQUEST

The request has been sent using the PUT method to edit a user of the system. To test the request when editing a new user of the system, it is necessary to encode the data in a JSON object and add it to the body that will be sent through the request as shown in Figure 7.15. The status 200 OK as shown in Figure 7.15 shows that the request was fulfilled. Additionally, the response from the server “The user has been updated” also confirm that the request has been fulfilled.

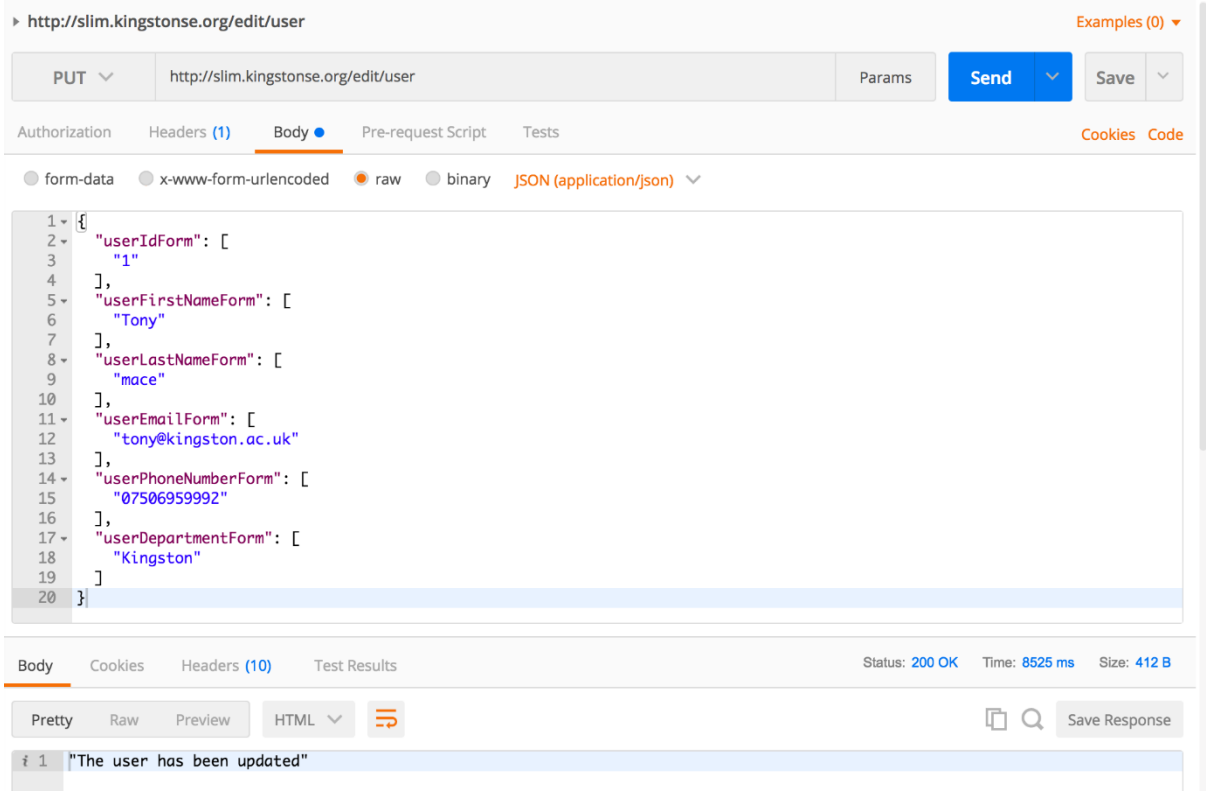


Figure 7.15 – Valid PUT REQUEST

7.5.4 Conclusion

The requests have been tested to ensure that the REST API is fully functional. The tests can be found in the Appendix Q. It is essential for the system to retrieve the right data to display to the user interface. It is also important to test that the REST API will insert and update properly the data into the database.

7.6 Conclusion

The functional testing has been completed to validate the implemented requirements. Some requirements will be implemented during future iterations and require the same process. The 63 test cases were completed and additional improvements will be made to ensure they are validated. The 25 test cases for non-functional testing were also completed to ensure that the system meets the quality expected by the client. Usability testing provided feedback from the users and validate the design of the system. Overall, the users were satisfied but further improvements will be carried out to get a high standard product. Unit testing has been conducted using Karma and Jasmine with Angular 5, to validate the implementation of the frontend. The REST API has been fully tested and is working has expected. It is responsive and can be easily expanded to handle more requests.

8 Critical Review

8.1 Introduction

This chapter is focusing on reviewing the project and discussing the problems encountered. The aims and objectives will be reviewed to see if the requirements have been validated. Furthermore, it is important to take into consideration legal, social and ethical aspects of the project and their implications.

8.2 Achievements

The aim of SPMS is to allow students and lecturers to work on common projects, keep track of the progress of each project, create and allocate tasks, share contents and give crucial feedback.

The first important phase of the project was to conduct extended researches and analyses, to identify stakeholders, extract the risks of the project and ensure the project can be conducted. Additional investigation was necessary to choose the appropriate management and development methodology, structure, plan and control the process of developing the project. Further research was conducted using user stories, use cases, use case diagram and interviews, to elicit and validate the requirements.

To fulfil client's requirements, a complex and secure MySQL database has been designed and implemented to support multiple campuses and organisations, and record and retrieve the large amount of data generated by the users. The database is now fully functional and can be expanded if necessary.

A REST API has been designed and implemented with Slim framework to ensure the front-end and the back-end are completely independent, so additional requirements would not need to refractor abundant code. The REST API is fully responsive, and the data can be efficiently retrieved or inserted into the database.

The user interface has been designed using sketches, wireframes, and sitemaps. The user interface has been developed using Angular 5 and bootstrap to ensure it is responsive, engaging, user friendly and can easily be refactored to accommodate any new requirements. Complex forms were implemented using Angular 5 to allow the administrator to setup the system for the users. The administrator is now able to see the essential data such as organisations, users, courses, modules of the system. The user interface for students and lecturers is only partially implemented as further iteration will be needed to ensure the quality of the product. Functional, non-functional and usability testing was necessary to validate the design and the implementation of the user interface.

Angular 5 and Slim framework are new technologies and by completing this project it was an opportunity to learn, compare and acquire new skills to develop future project. Angular 5 is a powerful but complex typescript tool to build web interface. Slim framework is a really simple way to build an efficient REST API.

The Agile methodology was the appropriate choice to ensure the communication was flowing between the clients and the developers. To improve the system, feedback and review of requirements meetings took place every week. The iterations process helped plan and organise the tasks in advance but also accommodate new tasks to complete the project.

8.3 Problems Encountered and Limitations

Bootstrap is a fully responsive framework allowing to implement web pages efficiently. The expandable sidebar needed menus and submenus as well as customised colours to satisfy the client. To implement the right expandable sidebar, it was necessary to deviate from the

Bootstrap structure and it required lots of overriding CSS styles or rewriting files. This process can be long and exhaustive.

Angular 5 is a powerful TypeScript framework but it has a limited documentation available to the public and it is necessary to do extensive research to implement specific functionalities. Angular 5 is also difficult to master and it is crucial to spend a lot of time learning the framework before to be able to implement simple web application. As an example, the form that has been implemented for the administrator to add users to the system contains more than 200 lines with complex functions calling multiple time the REST API to get the necessary data.

The Project management software is a large project targeting more than 20,000 users and the list of requirements is exhaustive. Considering that the person working on the project should handle mainly the following roles:

- System Analyst
- User Experience designer
- Software engineer
- Web developer
- Web designer
- Database administrator
- Quality assurance
- Software tester

It would have been more suitable to have other members in the team allocated to various tasks to ensure the project can be completed on time.

The time allocated to execute and complete the SPMS is limited. Unfortunately, only one person was allocated to complete the entire project and this person had other obligations while working on the project and would have to dedicate a large amount of time to improve his skills in completely different fields.

8.4 Future Work

The SPMS is not completed. Additional tasks are necessary to provide a fully working product to the client.

The administrator would require editing organisations, courses and modules. Considering that the administrator is already able to edit the users, it should be straight forward to implement these extra functionalities. The administrator would also need to be able to archive the users, organisations, modules and courses.

The front-end is partially implemented and a fully working dashboard would be needed for all users to enable or disable the settings and edit their profiles, to improve the user experience.

A notification system would also need to be implemented using the Mailchimp API to send emails. When a user is added to the system, a confirmation email is sent and the user can setup new password to access the system. Users should be notified via email when changes are made to a project they are associated to. (e.g. new comment on a task or a project).

As a possible extra feature, it was also suggested that any changes made to a project could directly be notified within Slack. Slack integration would allow students and lecturers to see in real time the changes made to a project or a task, and easily enable or disable some of the notifications if necessary.

8.5 Legal, Social and Ethical Aspects

The personal data stored in the database conform to the rules under Data Protection Act that applies for storing users ‘personal information and ensuring that the data will be fairly and lawfully used. The data should be only used for limited, specifically stated purposes and kept

for no longer than absolutely necessary. If the system is distributed outside the European Economic Area different rules may apply.

According to the definition on Open Source Initiative's website, open source code is source code that is readily available to users. If the source code is copyrighted or credited source code, the author needs to be credited. The system was implemented without any use of open-source code and did not require to credit any authors. The system was implemented without the use of any illegal software to develop the project management software.

8.6 Conclusion

The project has been an opportunity to work directly with a real client and get a proper inside of the skills required to complete an entire project. It also allowed to gain various technical skills using completely new technologies such as Angular 5 and Slim framework. It was also essential to manage appropriately the time available to meet the client deadline.

9 References

AltexSoft. (2018). *The Good and the Bad of Angular Development*. [online] Available at: https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-angular-development/ [Accessed 3 Jan. 2018].

Benbunan-Fich, R., Hiltz, S. R., & Turoff, M. (2003). *A comparative content analysis of face-to-face vs. asynchronous group decision making*. Decision Support Systems, 34(4), 457–469.

Blinklist. (2018). *Twiddla Review: Collaborate Using a Feature Rich Board*. [online] Available at: http://blinklist.com/reviews/twiddla [Accessed 1 Dec. 2017].

Blogs.umass.edu. (2018). *Twiddla – Online Tools for Teaching & Learning*. [online] Available at: https://blogs.umass.edu/onlinetools/twiddla/ [Accessed 3 Jan. 2018].

Bruffee, Kenneth A. *Collaborative Learning: Higher Education, Interdependence, and The Authority of Knowledge*. Baltimore: The Johns Hopkins University Press, 1993.

Caepnet.org. (2017). *Council for the Accreditation of Educator Preparation*. [online] Available at: http://caepnet.org/ [Accessed 10 Dec. 2017].

Citrix Blogs. (2018). *Introducing Talkboard: Drawing Your Way to Group Collaboration | Citrix Blogs*. [online] Available at: https://www.citrix.com/blogs/2013/09/12/introducing-talkboard-drawing-your-way-to-group-collaboration/ [Accessed 19 Dec. 2017].

Common Sense Education. (2018). *Wikispaces*. [online] Available at: https://www.commonsense.org/education/website/wikispaces [Accessed 3 Dec. 2017].

Contributor, U. (2018). *UnitedVoice: Tech That Turns Conversations Into Connection*. [online] Forbes. Available at: https://www.forbes.com/sites/united/2013/12/23/tech-that-turns-conversations-into-connection/#60dd43a6696e [Accessed 7 Dec. 2017].

Cornish WebServices, w. (2018). *Human Computer Interaction. Website Design. Internet Marketing, Website Design and Development*. [online] Cornishwebservices.co.uk. Available at: http://www.cornishwebservices.co.uk/website-design/human-computer-interaction.html [Accessed 19 Feb. 2018].

Coursetro.com. (2018). *Angular 5 Components Tutorial*. [online] Available at: https://coursetro.com/posts/code/106/Angular-5-Components-Tutorial [Accessed 2 Feb. 2018].

Craddock, A. (2017). *Organisational Transformation to the use of DSDM Atern for Project Management and Delivery*. [Online] http://nlighten.co.uk/. Available at: https://www.agileconference.org/wp-content/uploads/2011/06/Organisational-Transformation-to-the-use-of-DSDM-Atern-for-PM-and-Delivery.pdf [Accessed 12 Oct. 2017].

edshelf. (2018). *DebateGraph Reviews | edshelf*. [online] Available at: https://edshelf.com/tool/debategraph/ [Accessed 14 Dec. 2017].

edshelf. (2018). *DebateGraph Reviews | edshelf*. [online] Available at: https://edshelf.com/tool/debategraph/ [Accessed 1 Jan. 2018].

Edtech, H., EdTech? W., HyperDocs, H., Library, G., How., S., Schools, B., Edtech, W., Classroom, 5., Teachers, W., Classroom, U., Mayfield, T., Subjects, S. and Nepris, C. (2018). *7 Must Have Student-Collaboration Apps, Tools, and Resources - The Tech Edvocate*. [online] The Tech Edvocate. Available at: http://www.thetechedvocate.org/7-must-student-collaboration-apps-tools-resources/ [Accessed 1 Dec. 2017].

Fishbaugh, M. S. E. 1997. *Models of Collaboration. Boston: Allyn and Bacon.*

Getvoip.com. (2018). *The 10 Best Online Whiteboards with Realtime Collaboration | GetVoIP*. [online] Available at: https://getvoip.com/blog/2016/09/14/online-whiteboard-collaboration/ [Accessed 3 Feb. 2018].

Harvard Business Review. (2018). *Which Kind of Collaboration Is Right for You?* [online] Available at: https://hbr.org/2008/12/which-kind-of-collaboration-is-right-for-you [Accessed 10 Dec. 2017].

Herrington, A., & Herrington, J. (2006). *Authentic learning environments in higher education*. IGI Global.

Barker, L. J., Garvin-Doxas, K., & Jackson, M. (2002). *Defensive climate in the computer science classroom*. In, Proceedings of the 33rd SIGCSE technical symposium on Computer science education, pp. 43-47.

Ipad Educators. (2018). *Ipad Educators*. [online] Available at: http://www.ipadeducators.com/review-talkboard [Accessed 17 Dec. 2017].

Johnson, D. W., & Johnson, R. T. (1996). *Cooperation and the use of technology*. In D. Jonassen (Ed.), Handbookofresearchforeducationalcommunicationsandtechnology (pp. 785–812). London: MacMillan.

John-Steiner, V., R. J. Weber, and M. Minnis. 1998. *The challenge of studying collaboration. American Educational Research Journal* 35, no. 4: 773–83.

Kirschner, P. A., Martens, R. L., & Strijbos, J. W. (2004). *CSCL in higher education? A framework for designing multiple collaborative environments*. In J. W., Strijbos, P. A., Kirschner, & R. L., Martens (Eds.), What we know about CSCL: And implementing it in higher education (pp. 3–30). Boston, MA: Kluwer.

Lockyer, L., Patterson, J., & Harper, B. (2001). ICT in higher education: *Evaluating outcomes for health education.* Journal of Computer Assisted Learning, 17(3), 275–283.

Mason, R., & Romiszowski, A. (1996). Computer-mediated communication. In D. Jonassen (Ed.), Handbook of research for educational communications and technology (pp. 438–456). New York: Macmillan.

Montiel-Overall, P. (2018). *Toward a Theory of Collaboration for Teachers and Librarians.*. [online] Eric.ed.gov. Available at: https://eric.ed.gov/?id=EJ965627 [Accessed 19 Dec. 2017].

Okuogume, A., & Jäminki, S. (2011). *Experiences of integrating R&D&I in professional master degree programme: The case of competence-development-based education.* Hämeenlinna: Presented at the Combi Conference.

Orme, P. (2018). *Principles for Successful Button Design*. [online] Web Design Envato Tuts+. Available at: https://webdesign.tutsplus.com/articles/principles-for-successful-button-design--webdesign-6094 [Accessed 4 Jan. 2018].

Out-law.com. (2017). *Database rights: the basics*. [Online] Available at: https://www.outlaw.com/page-5698 [Accessed 12 Oct. 2017].

Pugach, M., and L. J. Johnson. 1995. *Collaborative practitioner's collaborative schools*. 1st ed. Denver, Colo.: Love.

Ray, R. (2018). *Free TalkBoard App for White Board Collaboration*. [online] SmallBizTechnology. Available at: http://www.smallbiztechnology.com/archive/2014/06/free-talkboard-app-for-white-board-collaboration-video-interview-with-chris-duncan-of-citrix.html/#.WmKNsai6-9I [Accessed 19 Dec. 2017].

Resta, P. and Laferrière, T. (2007). *Technology in Support of Collaborative Learning*. Educational Psychology Review, 19(1).

Resta, P. and Laferrière, T. (2007*). Technology in Support of Collaborative Learning*. Educational Psychology Review, 19(1), pp.65-83.

Roschelle, J. & Teasley S.D. (1995) *The construction of shared knowledge in collaborative problem solving*. In C.E. O'Malley (Ed), Computer-Supported Collaborative Learning. (pp. 69-197). Berlin: Springer-Verlag

Scholar.lib.vt.edu. (2018). *JTE v7n1 - Collaborative Learning Enhances Critical Thinking*. [online] Available at: http://scholar.lib.vt.edu/ejournals/JTE/v7n1/gokhale.jte-v7n1.html?ref=Sawos.Org [Accessed 17 Dec. 2017].

Shrage, M. 1990. Shared minds. New York: Random House.

Slim Framework. (2018). *Why Slim? What problem Slim solves better?* [online] Available at: http://discourse.slimframework.com/t/why-slim-what-problem-slim-solves-better/1692 [Accessed 7 Jan. 2018].

Sparxsystems.com.au. (2018). *Sparx Systems - UML 2 Tutorial - Activity Diagram*. [online] Available at: https://www.sparxsystems.com.au/resources/uml2_tutorial/uml2_activitydiagram.html [Accessed 16 Nov. 2017].

Students, H., If You Really Want to Design Useful Edtech, S., Tools, 8., Markets, H., Makeover, H., Like? W., Learning, T., Poverty, H., Teach, E., Educators, W., Time, R., learning, A. and The Tech Edvocate's List of 27 Assessment Apps, a. (2018). *7 Must Have Student-Collaboration Apps, Tools, and Resources - The Tech Edvocate*. [online] The Tech Edvocate. Available at: http://www.thetechedvocate.org/7-must-student-collaboration-apps-tools-resources/ [Accessed 10 Dec. 2017].

Techopedia.com. (2018). *What is Human-Computer Interaction (HCI)? - Definition from Techopedia*. [online] Available at: https://www.techopedia.com/definition/3639/human-computer-interaction-hci [Accessed 19 Jan. 2018].

The Interaction Design Foundation. (2018). *Introduction to the Essential Ideation Techniques which are the Heart of Design Thinking*. [online] Available at: https://www.interaction-design.org/literature/article/introduction-to-the-essential-ideation-techniques-which-are-the-heart-of-design-thinking [Accessed 15 Dec. 2017].

The Interaction Design Foundation. (2018). *What is Human-Computer Interaction (HCI)?* [online] Available at: https://www.interaction-design.org/literature/topics/human-computer-interaction [Accessed 19 Jan. 2018].

The Interaction Design Foundation. (2018). *What is User Centered Design?* [online] Available at: https://www.interaction-design.org/literature/topics/user-centered-design [Accessed 19 Dec. 2017].

The Mind Mapping Site. (2018). *Debategraph*. [online] Available at: http://www.mindmappingsite.com/mindmapping-web-tools/debategraph/ [Accessed 7 Jan. 2018].

Todaysmeet.com. (2018). *TodaysMeet - Give everyone a voice*. [online] Available at: https://todaysmeet.com/ [Accessed 19 Dec. 2017].

University, C. (2018). *Collaboration Tools - Eberly Center - Carnegie Mellon University*. [online] Cmu.edu. Available at: https://www.cmu.edu/teaching/technology/collaborationtools.html [Accessed 4 Dec. 2017].

Wahlin, D. (2018). *5 Key Benefits of Angular and TypeScript – Code with Dan Blog*. [online] Blog.codewithdan.com. Available at: https://blog.codewithdan.com/2017/08/26/5-key-benefits-of-angular-and-typescript/ [Accessed 19 Mar. 2018].

WhatIs.com. (2018). *What is collaboration? - Definition from WhatIs.com*. [online] Available at: http://whatis.techtarget.com/definition/collaboration [Accessed 17 Dec. 2017].

Wikispaces.com. (2018). *Wikispaces by TES - About*. [online] Available at: https://www.wikispaces.com/content/classroom/about [Accessed 10 Dec. 2017].

www.tutorialspoint.com. (2018). *Design Process and Task Analysis*. [online] Available at: https://www.tutorialspoint.com/human_computer_interface/design_process_and_task_analysis.htm [Accessed 19 Feb. 2018].

10 APPENDIX A - Interviews

Interview with stakeholders

- What is your name?
- What is your job title?
- What is your role in this organisation?
- Do you have a current solution to work on a common project? If yes, what solution are you using?
- How the group are formed to start working on a project? Is it an easy process?
- How many group projects are you currently working on?
- When working on a group project, how do you communicate with the team members?
- How do you share documentation with your colleagues? What solution are you using?
- Do you meet often with your colleagues when working on a common project?
- How are you currently keeping track of the progress of the team project?
- How many modules involved group projects?
- Do you have a current solution to create groups within your module?
- Do you have an easy way of tracking all the members of each single group?
- Are you able to easily view all the members of your module?
- Do you see any features that you would had to the system? What would be useful for you?

Interview one-to-one with client

- Do you think that roles can be updated for Team Leader and Team Member?
- Do you think that only Team Leader should be able to invite new team member?
- Do you think that only a Team Leader should be able to assign and unassigned a task?
- Would you allow team members to create tasks or should it be only done by the Team Leader? If Team Members can create a task, should it be validated by the Team Leader?
- Do you think that a task should be assigned to a single or multiple user?
- Do you think that contents could be shared for individual task or should the system have a specific section to share the content?
- Does notifications should be sent for every single or only particular changes made on a project?
- Should user have the option to customise the notifications for the project?
- Is it essential for the user to be able to customise the design of the interface?
- Do you see any features that you would had to the system? What would be useful for you?

11 APPENDIX B - User stories

User stories for team leaders.

As a/an	I want to	So that
Team leader	Invite Team members to work on a common project	I can build my Team for the project.
Team leader	Create a project	I can start inviting Team members.
Team leader	Archive a project	I can hide completed projects.
Team leader	Comment on the project	I can give feedback or essential information to other Team members.
Team leader	Add goals to a project	The Team can have a clear understanding on what the goals are.
Team leader	Add objectives to a project	The Team can have a clear understanding on what the objectives are.
Team leader	View overall progress of the project	I can ensure we meet our deadline.
Team leader	View project's Team members	I can easily view all the tasks they are assigned to.
Team leader	View all member of a course	I can invite new members to a project.
Team leader	View all members assigned to modules	I can invite new members to a project.
Team leader	Assign tasks to Team members	I can delegate the work and track the progress.
Team leader	Add a Team member	I can build a Team
Team leader	Un-assign a task	I can reassign the task to another Team member
Team leader	Create a task	I can assign a task to a Team member and delegate the work.
Team leader	View incomplete tasks	I can have a general overview of the progress of the project.
Team leader	View completed tasks	I can have a general overview of the progress of the project.
Team leader	Comment on task (subtasks)	I can give feedback, support and help the Team members within the project.
Team leader	View all project's tasks	I can have an overview of tasks assigned to each user and what needs to be achieved to complete the project.
Team leader	View status of a task (subtask)	I can see the progress of the task.
Team leader	Edit the status of a task (subtask)	I can update the status of the task.
Team leader	Be notified on any comments and status changes within the project.	I can constantly be updated in real time.
Team leader	Share content with all Team members	I can support and help Team members with their tasks.
Team leader	Download content from other Team members	I can view the content shared by the Team members.
Team leader	Upload and download the most common type of files as pdf or word document.	I can easily share contents with the Team and view any content shared.

User stories for team members

As a/an	I want to	So that
Team Member	Comment on the project	I can give a feedback or essential information to other Team members.
Team Member	View overall progress of the project	I can ensure we meet our deadline.
Team Member	View project's Team members	I can see with who I am working on the project.
Team Member	View all member of a course	I can invite new members to a project.
Team Member	View all members assigned to modules	I can invite new members to a project.
Team Member	View incomplete tasks	I can have a general overview of the progress of the project.
Team Member	View completed tasks	I can have a general overview of the progress of the project.
Team Member	Comment on task (subtasks)	I can give feedback, support and help the Team members of the project.
Team Member	View all project's tasks	I can have an overview of tasks assigned to each user and what needs to be achieved to complete the project.
Team Member	View status of a task (subtask)	I can see the progress of the task.
Team Member	Edit the status of a task (subtask)	I can update the status of the task.
Team Member	Be notified on any comments and status changes within the project.	I can constantly be updated in real time.
Team Member	Share content with all Team members	I can support and help Team members with their tasks.
Team Member	Download content from other Team members	I can view the content shared by the Team members.
Team Member	Upload and download the most common type of files as pdf or word document.	I can easily share contents with the Team and view any content shared.

12 APPENDIX C - Use cases

Use Case 1 - Create a project

Use case 1	Create project	
Goal in context	A user creates a project	
Preconditions	The user should be logged in	
Post Condition	The system is available, and the user has been added to the system	
Success End condition	The user creates a project and become a Team leader	
Failed End condition	The user is unable to create a project	
Primary, Secondary actors	Team leader, Team member	
Main Flow	**Step**	**Action**
Navigate to login page	1	The user opens the software interface on his browser to view the login page
Login to the system	2	The user must enter and validates his login details
Navigate to the side bar	3	The user navigates to the sidebar to find the project section
User press the “add a project button”	4	The user creates a project by pressing the appropriate button
Validation message	5	The user gets a validation message when the project has been created
Alternative Flow	**Step**	**Action**
User already logged in	1	The user is already logged in
Expend the sidebar	2	The user expends the sidebar to see the “create a new project” button
Navigate to the side bar	3	The user navigates to the sidebar to find the project section
User press the “add a project button”	4	The user creates a project by pressing the appropriate button
Validation message	5	The user gets a validation message when the project has been created
Priority		must
Performance		The new project should be created within 5 seconds
Frequency		

Use Case 2 – Invite Team Member

Use case 2	Invite Team member	
Goal in context	A Team leader invite a new Team member to join the Team	
Preconditions	The Team leader should be logged in	
Post Condition	The system is available, and the user has been added to the system, the Team leader has already created a project	
Success End condition	The Team leader invites a new Team Member	
Failed End condition	The Team leader could not invite a Team Member	
Primary, Secondary actors	Team leader, Team member	
Main Flow	**Step**	**Action**
Navigate to the side bar	1	The Team leader navigates to the sidebar to find the project section
Open current project	2	The Team leader open the current project by clicking the name of the project
Invite a new member	3	The Team leader press the “invitation” button
An invitation email is sent to the user	4	An email is sent to the future Team Member
The user validates the invitation	5	The user who received the email validate the invitation by pressing a redirection link
Alternative Flow		none
Priority		must
Performance		The email should be sent within 30 seconds for the user to validate the invitation
Frequency		

Use case 3 – Create a Task

Use case 3	Create task	
Goal in context	A Team leader creates a new task	
Preconditions	The Team leader should be logged in	
Post Condition	The system is available, and the user has been added to the system. The Team leader has already created a project.	
Success End condition	The Team leader create a new task	
Failed End condition	The Team leader could not create a new task	
Primary, Secondary actors	Team leader, Team member	
Main Flow	**Step**	**Action**
Navigate to the side bar	1	The Team leader navigates to the sidebar to find the project section
Open current project	2	The Team leader open the current project by clicking the name of the project
Add a workflow step	3	The Team leader adds a workflow step
Add a workflow step name	4	The Team leader will name the workflow step
Add a task	5	The Team leader will press the “add a task” button
Alternative		none
Priority		must
Performance		The task should be created within 2 seconds
Frequency		

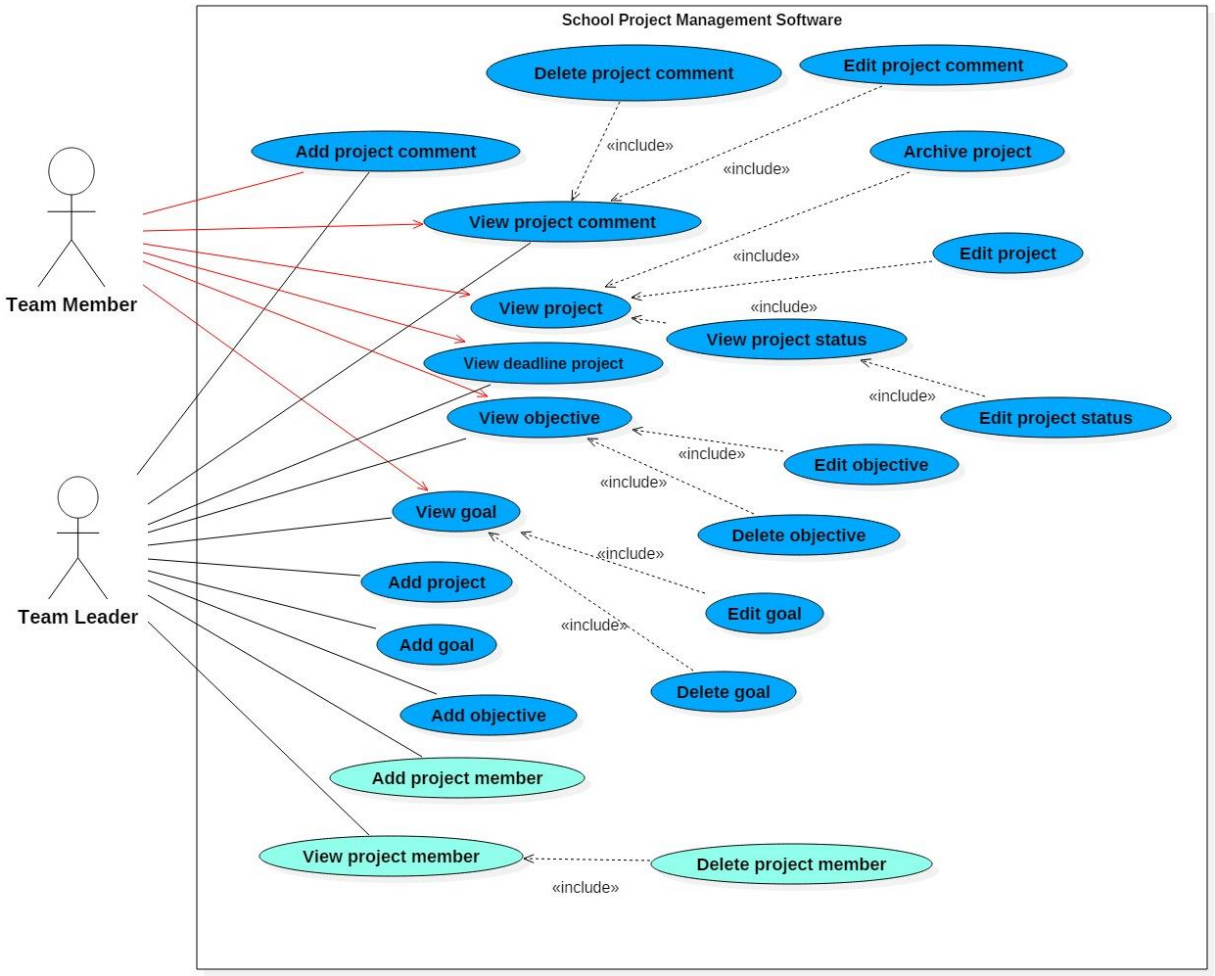
Use Case 4 – Assign a Task

Use case 4	Assign Task	
Goal in context	A Team leader assign a task to a Team member	
Preconditions	The Team leader should be logged in	
Post Condition	The system is available, and users have been added to the system. The Team leader has already created a project and a task. The project has a Team member.	
Success End condition	The Team leader assign a task to a Team member	
Failed End condition	The Team leader could not assign a task to a Team member	
Primary, Secondary actors	Team leader, Team member	
Main Flow	**Step**	**Action**
Navigate to the side bar to view the project	1	The Team leader navigates to the sidebar to find the project
Open current project	2	The Team leader open the current project by clicking the name of the project
Select the task	3	The Team leader select the task by clicking it.
Search for a Team member	4	The Team leader enter the name of the Team member in the field
Assign a Team member	5	The Team leader press the “assign” button
Alternative	**Step**	**Action**
Navigate to the side bar to view the task	1	The Team leader navigates to the sidebar to find the “view all task” section
Display all tasks	2	The Team leader presses the “view all tasks” button
Select the task	3	The Team leader select the task by clicking it.
Search for a Team member	4	The Team leader enter the name of the Team member in the field
Assign a Team member	5	The Team leader press the “assign” button
Priority		must
Performance		The task should be assigned to the team member within 2 seconds
Frequency		

13 APPENDIX D - Use case diagrams

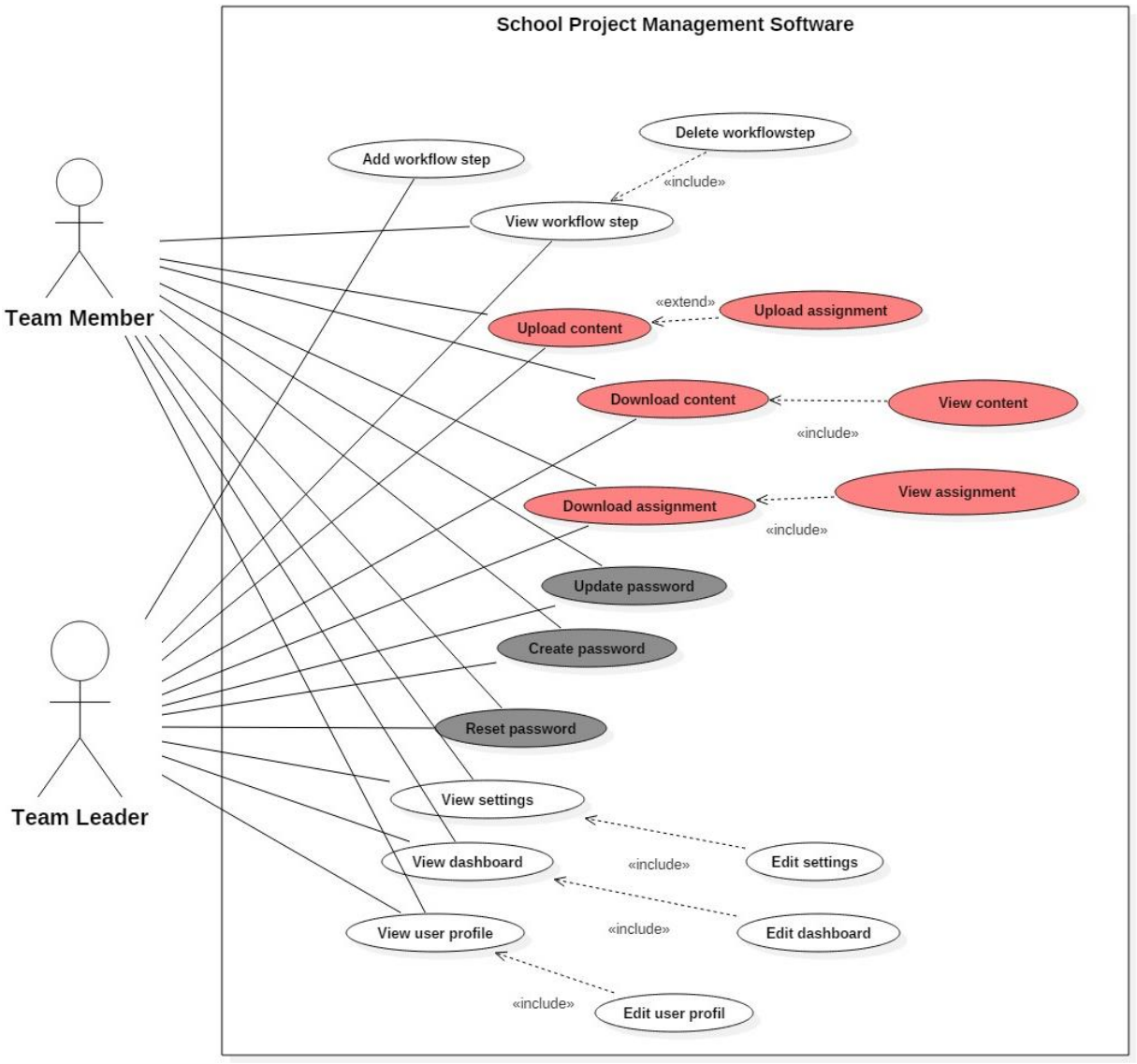
Use Case Diagram (Organisation, Course, Module and User)

Use Case Diagram (Project, Comment, Objective, Goal, Member and Status)



Use Case Diagram (Task, Task Comment and Deadline)

Use Case Diagram (Workflow, Assignment, Content, Password, Setting and Dashboard)



14 APPENDIX E – Non-functional requirements

- The system shall ensure that a user can login within 2 seconds.
- The system shall be available 24/7 with short break for maintenance during off-peak hours.
- The system shall send a notification email within the next 2 seconds after a change has been made on a project.
- The system shall send a notification email within 30 seconds after a change has been made on a project.
- The system shall be supported by all the modern browsers on desktop.
- The system shall provide a reliable database backup actioned automatically every week.
- The system shall encrypt all the sensitive data as passwords.
- The system shall send a confirmation email when a user logging in for the first time and validate it once the user has approved.
- The system shall support a minimum of 1000 user logged in at the same time.
- The system should support a total of 20000 users.
- The system shall only support passwords composed of capital letters, small letters and sign, to be validated.
- The system shall update the database and the user interface within a second when a change is made on a “Project”.
- The system shall ensure the user is logged out after 15 minutes of inactivation.
- The system shall provide a confirmation pop-up when the user profile has been updated and changes need to be saved.
- The system shall display an obvious and visible warning, and a confirmation field to prevent user from errors when pressing a delete field (e.g. deleted project).
- The system shall ensure that the data visible on the interface is only visible by the appropriate user. (privileges)
- The system shall be able to recover within the next hour if any down-time.

15 APPENDIX F - MoSCoW prioritisation

Administrator **Admin** Team member **TM** Team Leader **TL**

ID	Requirements	Description	Priority	User
1	Add organisation	The admin should be able to define an organisation for all the projects	MUST	Admin
2	Edit organisation	The admin should be able to edit the organisation already added	MUST	Admin
3	View organisation	The administrator should be able to view the organisation	MUST	Admin
4	Delete organisation	The administrator should be able to delete an organisation	SHOULD	Admin
5	Add user	The administrator should be able to add users to the system	MUST	Admin
6	Delete user	The administrator should be able to delete a user from the system	MUST	Admin
7	Edit user	The administrator should be able to edit the details of a user	MUST	Admin
8	View user	Admin, Team Member and Team Leader should be able to view the users. Team Member and Team Leader will be restricted to specific modules.	MUST	Admin TL, TM
9	Add project	The Team Leader should be able to create a new project.	MUST	TL, TM
10	Archive project	The Team Leader should be able to archive a project when it is completed.	SHOULD	TL, TM
11	Edit project	The team Leader should be able to edit a project.	MUST	TL
12	View project	The Team Leader and Team member should be able to view a project.	MUST	TL, TM
13	Add project member	The Team Leader should be able to add a project member to the existent project.	MUST	TL
14	Delete project member	The Team Leader should be able to remove a member form the project	MUST	TL
15	View project member	The Team Leader and the Team member should be able to view all the members working on the same project	MUST	TL, TM
16	Create password	The Team Leader and the Team member should be able to generate a password when logging in for the first time	MUST	TL, TM
17	Reset password	The Team Leader and the Team member should be able to reset their password if it is forgotten	MUST	TL, TM
18	Update password	The Team Leader and the Team member should be able to update their password for security reason	MUST	TL, TM
19	Add task comment	The Team Leader and the Team member should be able to add a comment to a task	SHOULD	TL, TM

ID	Requirements	Description	Priority	User
20	Edit task comment	The Team Leader and the Team member should be able to edit their own comments on a task	SHOULD	TL, TM
21	Delete task comment	The Team Leader should be able to delete a task	COULD	TL
22	View task comment	The Team Leader and the Team member should be able to view all the comments on all the tasks	SHOULD	TL, TM
23	Add project comment	The Team Leader and the Team member should be able to add comments to the project	SHOULD	TL, TM
24	Edit project comment	The Team Leader and the Team member should be able to edit their own comments only	SHOULD	TL, TM
25	View project comment	The Team Leader and the Team member should be able to view all the comments on a project	SHOULD	TL, TM
26	Delete project comment	The Team Leader and the Team member should be able to delete only their own comments	COULD	TL, TM
27	Create task	The Team Leader and the Team member should be able to create tasks, but validation may be needed by the team leader	MUST	TL, TM
28	Edit task	The Team Leader and the Team member should be able to edit the tasks	MUST	TL, TM
29	Delete task	The Team Leader should be able to delete task	MUST	TL
30	Assign task	The Team Leader should be able to assign task	MUST	TL
31	Un-assign task	The Team Leader should be able to un-assign task	MUST	TL
32	View task	The Team Leader and the Team member should be able to view all the tasks for their own project	MUST	TL, TM
33	Validate task	The Team Leader should be able to validate a task created by a Team member	COULD	TL
34	Add objective	The Team Leader should be able to add the objectives to the project	MUST	TL
35	Edit objective	The Team Leader should be able to edit the objectives to the project	MUST	TL
36	Delete objective	The Team Leader should be able to add the objectives to the project	MUST	TL
37	View objective	The Team Leader and the Team member should be able to view the objectives to the project	MUST	TL, TM
38	Add goal	The Team Leader should be able to add the goals to the project	MUST	TL
39	Edit goal	The Team Leader should be able to edit the goals to the project	MUST	TL
40	Delete goal	The Team Leader should be able to delete the goals to the project	MUST	TL
41	View goal	The Team Leader and the Team member should be able to view all the goals of the project	MUST	TL, TM

ID	Requirements	Description	Priority	User
42	View project status	The Team Leader and the Team member should be able to view the status of the project	SHOULD	TL, TM
43	Edit project status	The Team Leader should be able to edit the status of the project	SHOULD	TL
44	Edit task status	The Team Leader and the Team member should be able to edit the status of a task	SHOULD	TL, TM
45	View task status	The Team Leader and the Team member should be able to view the status of a task	SHOULD	TL, TM
46	Upload content	The Team Leader and the Team member should be able to upload content to a task to share with other members	MUST	TL, TM
47	Download content	The Team Leader and the Team member should be able to download content	MUST	TL, TM
48	View content	The Team Leader and the Team member should be able to view the content that has been downloaded	MUST	TL, TM
49	Upload assignment	The Team Leader and the Team member should be able to upload the assignment completed by the team	MUST	TL, TM
50	Download assignment	The Team Leader and the Team member should be able to download the assignment completed by the team	MUST	TL, TM
51	View assignment	The Team Leader and the Team member should be able to view the assignment completed by the team	MUST	TL, TM
52	Add modules	The administrator should be able to add modules to an existing course	MUST	Admin
53	Edit modules	The administrator should be able to edit the modules	MUST	Admin
54	Delete modules	The administrator should be able to remove the modules	MUST	Admin
55	Add course	The administrator should be able to add a new course	MUST	Admin
56	Edit course	The administrator should be able to edit a course	MUST	Admin
57	Delete course	The administrator should be able to delete a course	MUST	Admin
58	Add project deadline	The Team Leader should be able to add a deadline to the project	MUST	TL
59	View project deadline	The team Leader and the Team Member should be able to view the deadline of the project	MUST	TL, TM
60	Delete project deadline	The Team Leader should be able to remove the deadline of a project	SHOULD	TL

ID	Requirements	Description	Priority	User
61	Add task deadline	The Team Leader and the Team member should be able to add a deadline to a task	MUST	TL, TM
62	View task deadline	The Team Leader and the Team member should be able to view the deadline of a task	MUST	TL, TM
63	Delete task deadline	The Team Leader and the Team member should be able to remove the deadline of a task	SHOULD	TL, TM
64	View workflow step	The Team Leader and the Team member should be able to view the workflow step of the project	MUST	TL, TM
65	Add workflow step	The Team Leader should be able to add the workflow step of the project	MUST	TL
66	View dashboard	The team Leader and the Team Member should be able to view their own dashboard	COULD	TL, TM
67	Edit dashboard	The team Leader and the Team Member should be able to edit their own dashboard	COULD	TL, TM
68	View setting	The team Leader and the Team Member should be able to view their own settings	SHOULD	TL, TM
69	Edit setting	The team Leader and the Team Member should be able to edit their own settings	SHOULD	TL, TM

16 APPENDIX G - Data dictionary

relation name	attribute name	Data type	length	PK/FK	NOT NULL	other constraints
Organisation	organisation_id	number	20	PK	NOT NULL	AUTO INCREMENT
	organisation_name	varchar	250		NOT NULL	
	organisation_address	varchar	500		NOT NULL	
	organisation_type	varchar	250		NOT NULL	
	organisation_phone_number	varchar	30		NOT NULL	
	organisation_country	varchar	100		NOT NULL	
	organisation_created_at	datetime				
	organisation_updated_at	datetime				
	organisation_deleted_at	datetime				

relation name	attribute name	Data type	length	PK/FK	NOT NULL	other constraints
User_Type	user_type_id	number	20	PK	NOT NULL	AUTO INCREMENT
	user_type_role	varchar	250		NOT NULL	

relation name	attribute name	Data type	length	PK/FK	NOT NULL	other constraints
Course	course_id	number	20	PK	NOT NULL	AUTO INCREMENT
	course_name	varchar	250		NOT NULL	
	course_year	date			NOT NULL	
	course_description	varchar	1000		NOT NULL	
	course_created_at	datetime				
	course_updated_at	datetime				
	course_deleted_at	datetime				

relation name	attribute name	Data type	length	PK/FK	NOT NULL	other constraints
users	user_id	number	20	PK	NOT NULL	AUTO INCREMENT
	organisation_id	number	20	FK	NOT NULL	
	user_type_id	number	20	FK	NOT NULL	
	user_first_name	varchar	250		NOT NULL	
	user_last_name	varchar	250		NOT NULL	
	user_date_of_birth	date			NOT NULL	
	user_password	char	250		NOT NULL	
	user_email	varchar	250		NOT NULL	
	user_phone_number	varchar	30		NOT NULL	
	user_departement	varchar	250		NOT NULL	
	user_about_me	varchar	500			
	user_picture_path	varchar	500			
	user_created_at	datetime				
	user_updated_at	datetime				
	user_deleted_at	datetime				

relation name	attribute name	Data type	length	PK/FK	NOT NULL	other constraints
Users-module	user_id	number	20	FK	NOT NULL	
	module_id		20	FK	NOT NULL	

relation name	attribute name	Data type	length	PK/FK	NOT NULL	other constraints
Users_Course	user_id	number	20	CK	NOT NULL	
	course_id	number	20	CK	NOT NULL	

relation name	attribute name	Data type	length	PK/FK	NOT NULL	other constraints
Module	module_id	number	20	PK	NOT NULL	AUTO INCREMENT
	module_name	varchar	250		NOT NULL	
	module_year	Date			NOT NULL	
	module_description	varchar	1000		NOT NULL	
	module_created_at	datetime				
	module_updated_at	datetime				
	module_deleted_at	datetime				

relation name	attribute name	Data type	length	PK/FK	NOT NULL	other constraints
Module_Course	module_id	number	20	FK	NOT NULL	
	course_id	number	20	FK	NOT NULL	

relation name	attribute name	Data type	length	PK/FK	NOT NULL	other constraints
Team	team_id	number	20	PK	NOT NULL	AUTO INCREMENT
	team_description	varchar	1000		NOT NULL	
	team_created_at	datetime				
	team_updated_at	datetime				
	team_deleted_at	datetime				

relation name	attribute name	Data type	length	PK/FK	NOT NULL	other constraints
User_Team	user_id	number	20	CK	NOT NULL	
	team_id	number	20	CK	NOT NULL	

relation name	attribute name	Data type	length	PK/FK	NOT NULL	other constraints
Project	project_id	number	20	PK	NOT NULL	AUTO INCREMENT
	team_id	number	20	FK	NOT NULL	
	project_name	varchar	250		NOT NULL	
	project_description	varchar	1000		NOT NULL	
	project_creator	varchar	250		NOT NULL	
	project_due_date	datetime				
	project_duration	datetime				
	project_created_at	datetime				
	project_updated_at	datetime				
	project_archived_at	datetime				

relation name	attribute name	Data type	length	PK/FK	NOT NULL	other constraints
Goal	goal_id	number	20	PK	NOT NULL	AUTO INCREMENT
	project_id	number	20	FK	NOT NULL	
	goal_description	varchar	1000			
	goal_created_at	datetime				
	goal_updated_at	datetime				
	goal_deleted_at	datetime				

relation name	attribute name	Data type	length	PK/FK	NOT NULL	other constraints
Objective	objective_id	number	20	PK	NOT NULL	AUTO INCREMENT
	project_id	number	20	FK	NOT NULL	
	objective_description	varchar	1000			
	objective_created_at	datetime				
	objective_updated_at	datetime				
	objective_deteled_at	datetime				

relation name	attribute name	Data type	length	PK/FK	NOT NULL	other constraints
Project_Comment	project_comment_id	number	20	PK	NOT NULL	AUTO INCREMENT
	project_id	number	20	FK	NOT NULL	
	project_comment_description	varchar	1000		NOT NULL	
	project_comment_creator	varchar	250		NOT NULL	
	project_comment_created_at	datetime				
	project_comment_updated_at	datetime				
	project_comment_deleted_at	datetime				

relation name	attribute name	Data type	length	PK/FK	NOT NULL	other constraints
Workflow_Step	workflow_step_id	number	20	PK	NOT NULL	AUTO INCREMENT
	project_id	number	20	FK	NOT NULL	
	worflow_step_name	varchar	250			
	workflow_step_created_at	datetime				
	workflow_step_updated_at	datetime				
	workflow_step_deleted_at	datetime				

relation name	attribute name	Data type	length	PK/FK	NOT NULL	other constraints
Assignee	assignee_id	number	20	PK	NOT NULL	AUTO INCREMENT
	assignee_name	varchar	250		NOT NULL	
	assignee_created_at	datetime				
	assignee_updated_at	datetime				
	assignee_deleted_at	datetime				

relation name	attribute name	Data type	length	PK/FK	NOT NULL	other constraints
Task	task_id	number	20	PK	NOT NULL	AUTO INCREMENT
	parent_task_id	number	20	FK	NOT NULL	refer to parent task (subtask)
	project_id	number	20	FK	NOT NULL	
	assignee_id	number	20	FK		
	task_title	varchar	250		NOT NULL	
	task_creator	varchar	250		NOT NULL	
	task_status	varchar	250			
	task_description	varchar	1000			
	task_due_date	datetime				
	task_created_at	datetime				
	task_updated_at	datetime				
	task_deleted_at	datetime				

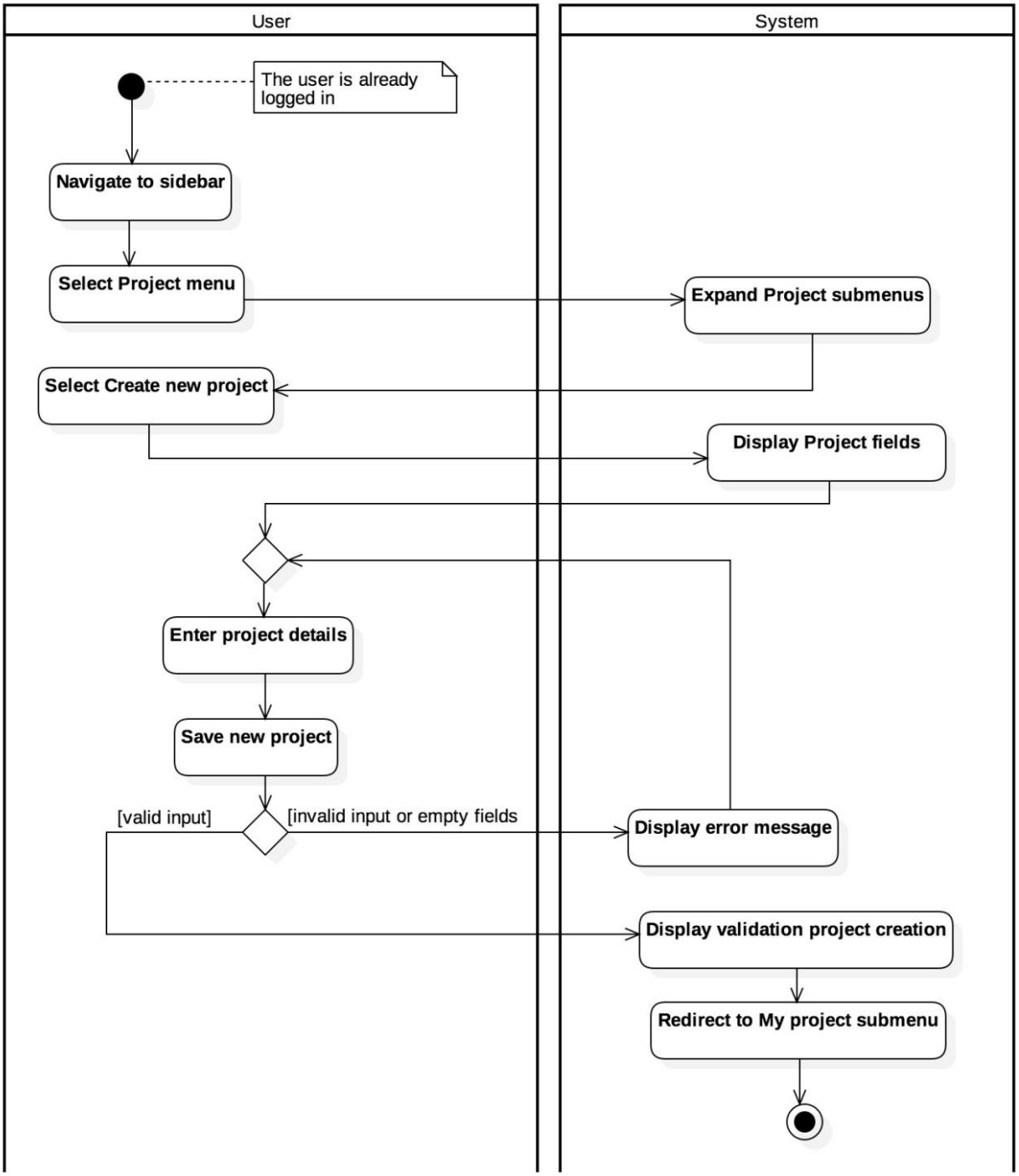
relation name	attribute name	Data type	length	PK/FK	NOT NULL	other constraints
content	content_id	number	20	PK	NOT NULL	AUTO INCREMENT
	task_id	number	20	FK	NOT NULL	
	content_path	varchar	500		NOT NULL	
	content_type	varchar	250		NOT NULL	
	content_creator	varchar	250		NOT NULL	
	content_created_at	datetime				
	content_updated_at	datetime				
	content_deleted_at	datetime				

relation name	attribute name	Data type	length	PK/FK	NOT NULL	other constraints
Task_Comment	task_comment_id	number	20	PK	NOT NULL	AUTO INCREMENT
	task_id	number	20	FK	NOT NULL	
	task_comment_description	varchar	1000			
	task_comment_creator	varchar	250		NOT NULL	
	task_comment_created_at	datetime				
	task_comment_updated_at	datetime				
	task_comment_deleted_at	datetime				

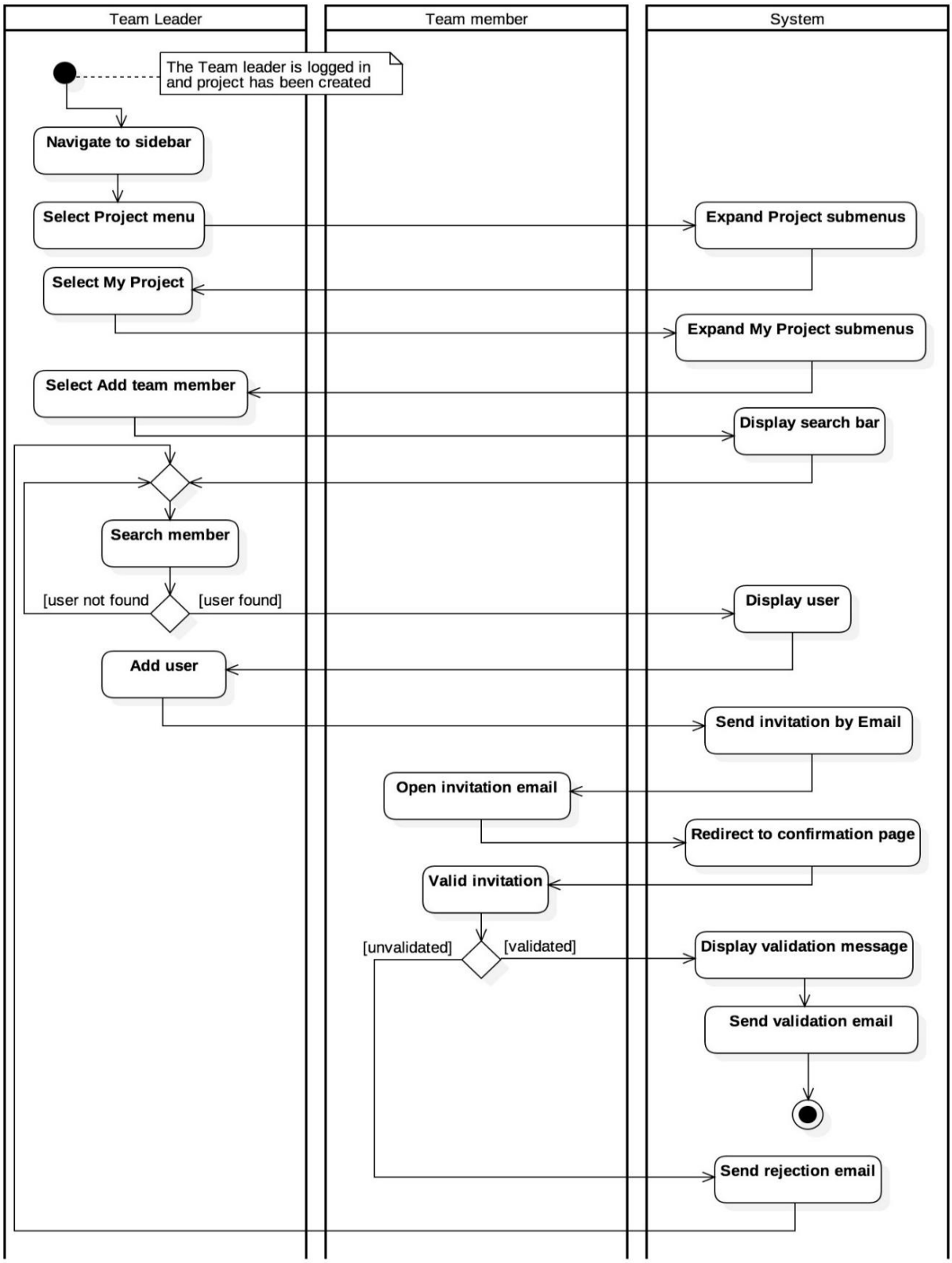
relation name	attribute name	Data type	length	PK/FK	NOT NULL	other constraints
Notification	notification_id	number	20	PK	NOT NULL	AUTO INCREMENT
	notification_type	varchar	250		NOT NULL	
	notification_description	varchar	1000			
	notification_created_at	datetime				

17 APPENDIX H - Activity Diagrams

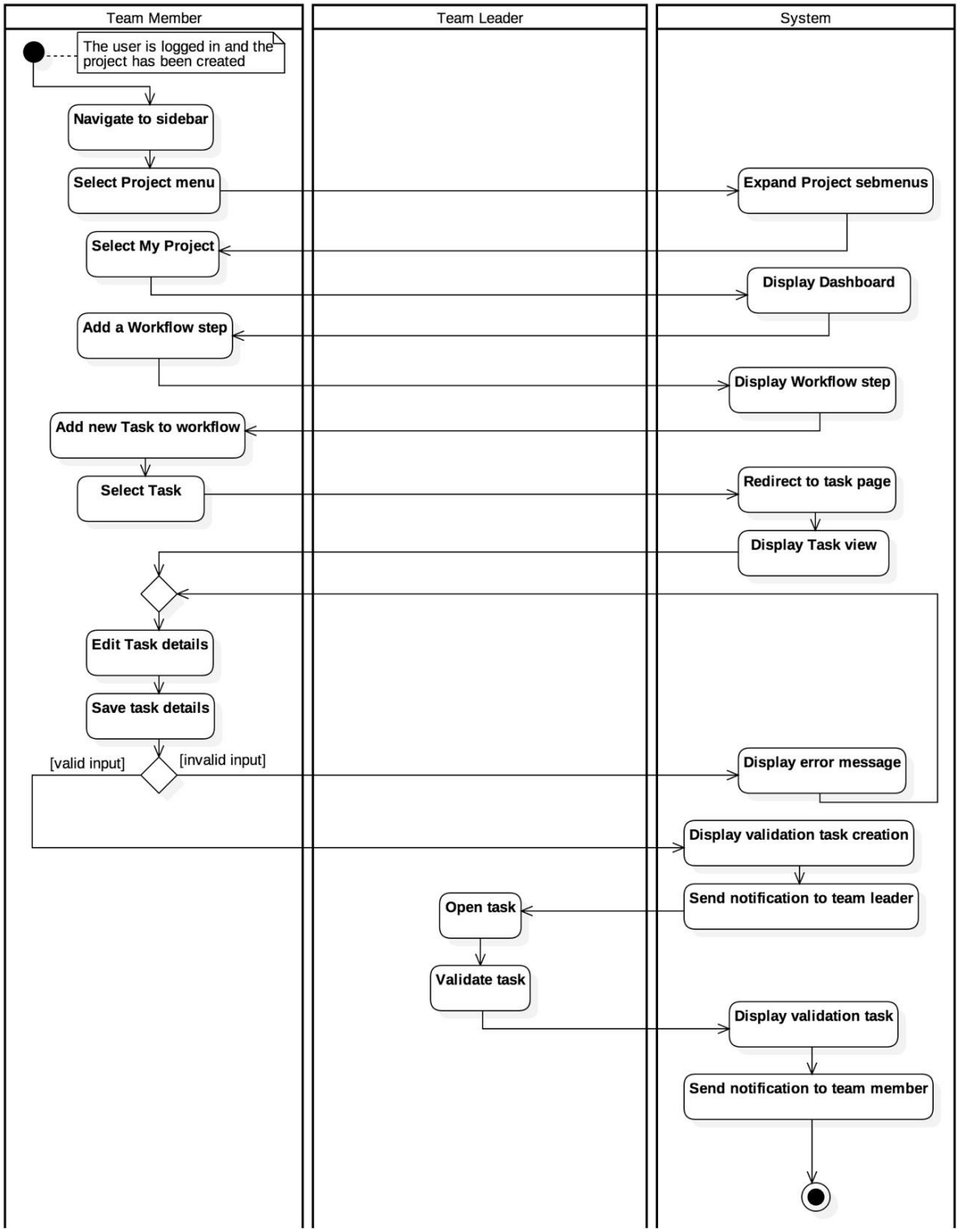
A user Create a project:



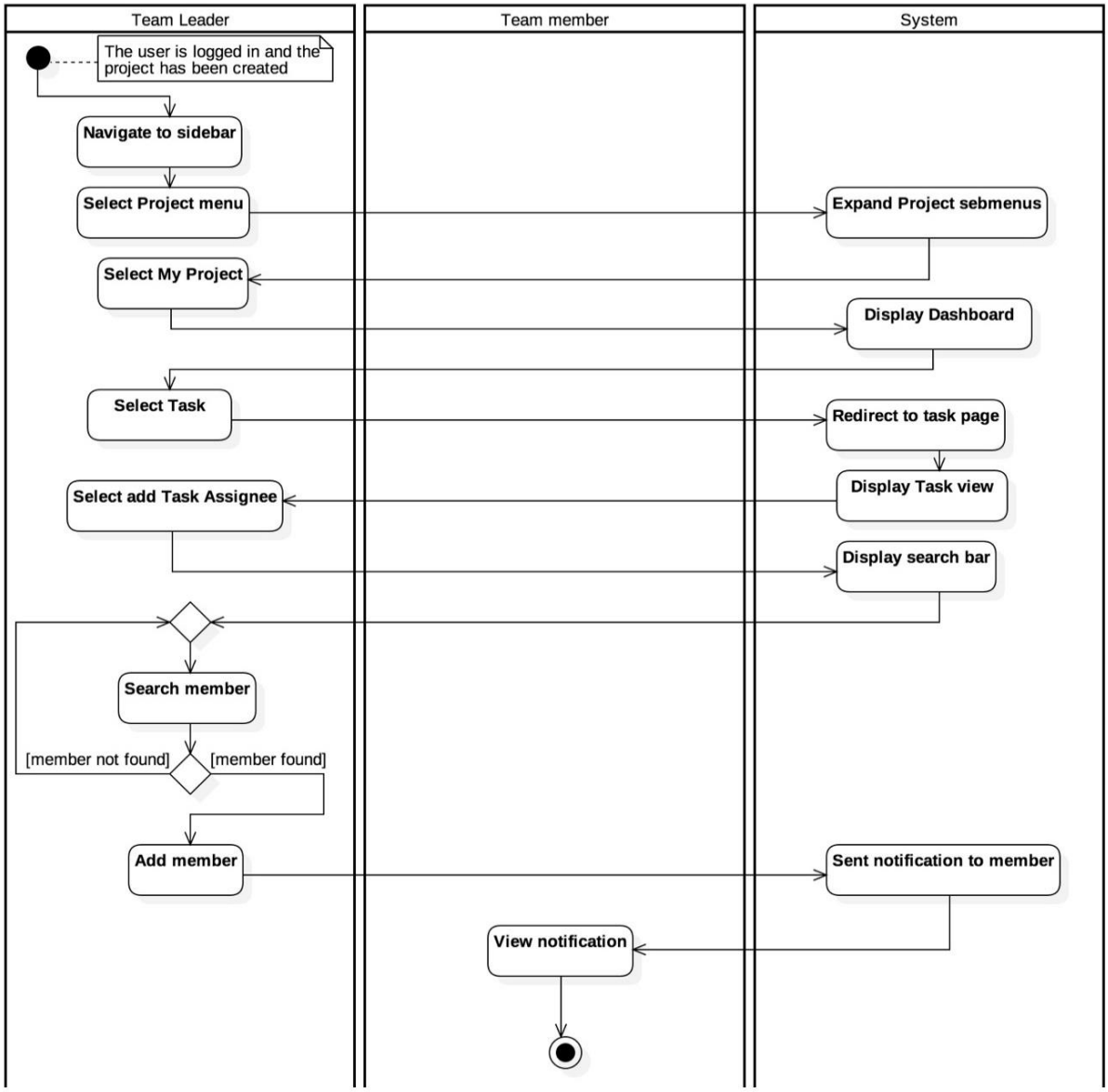
A user invites a member:



A user creates a task:

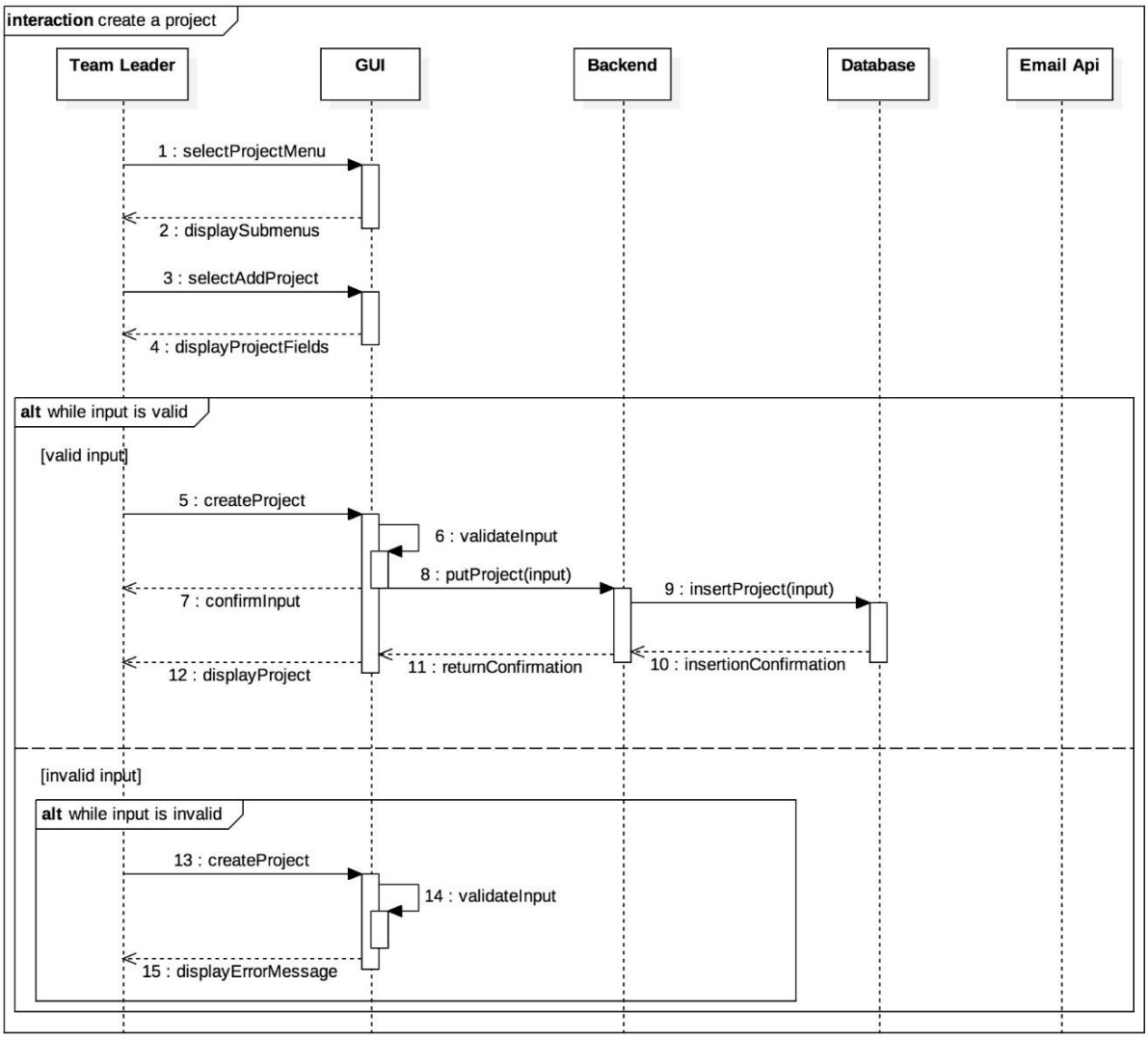


A user assigns a task:

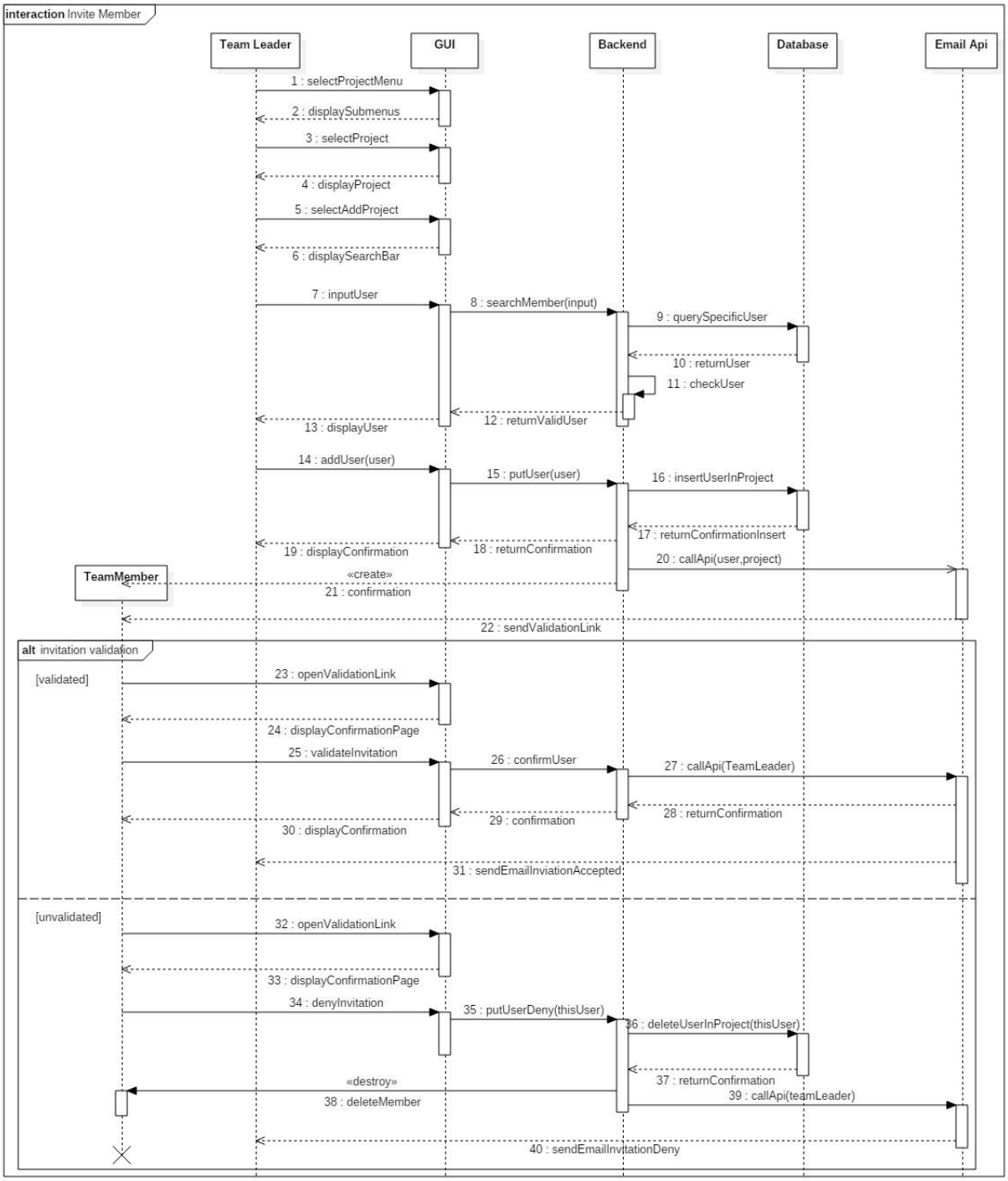


18 APPENDIX I - Sequence Diagrams

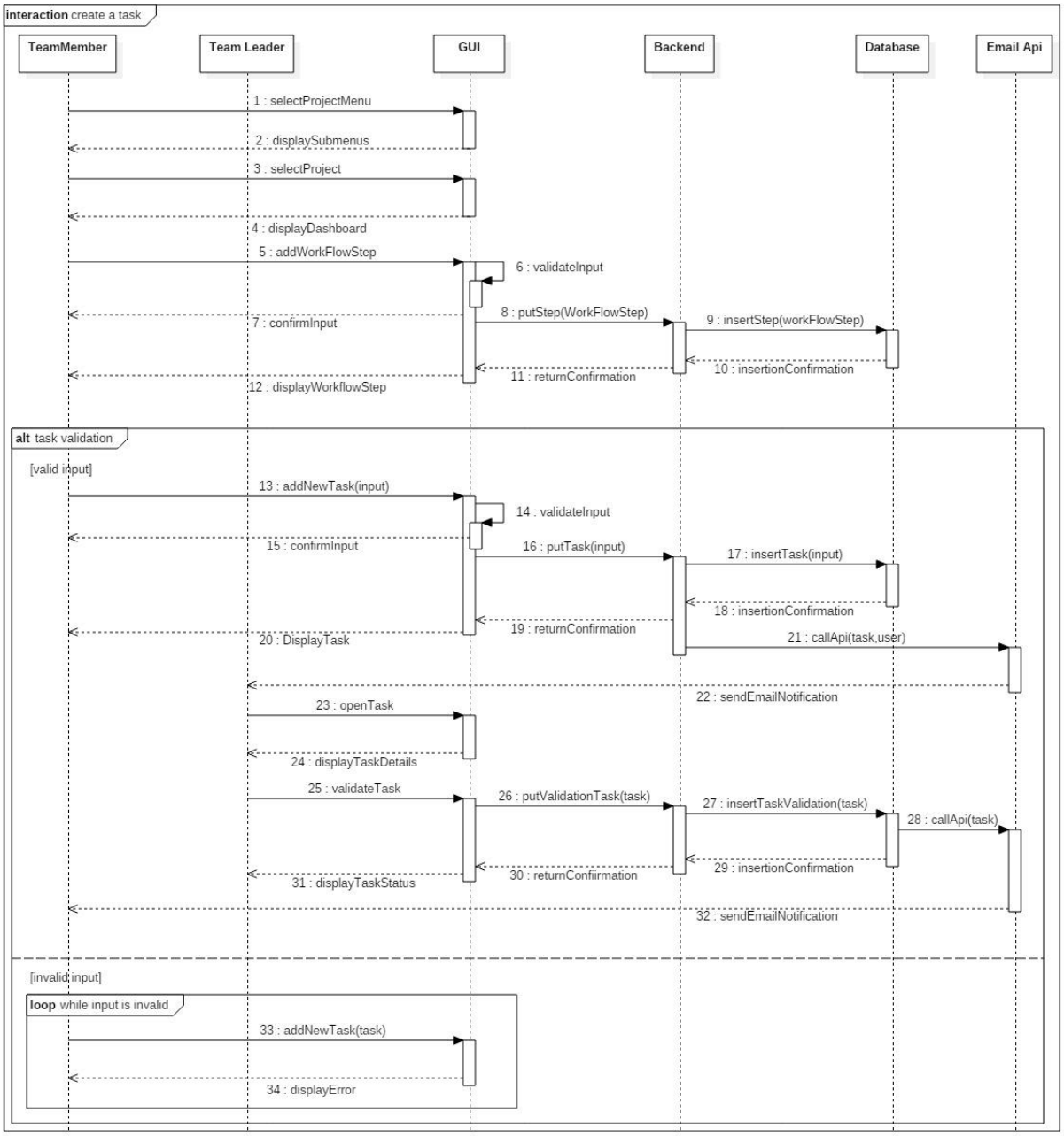
A user creates a project:



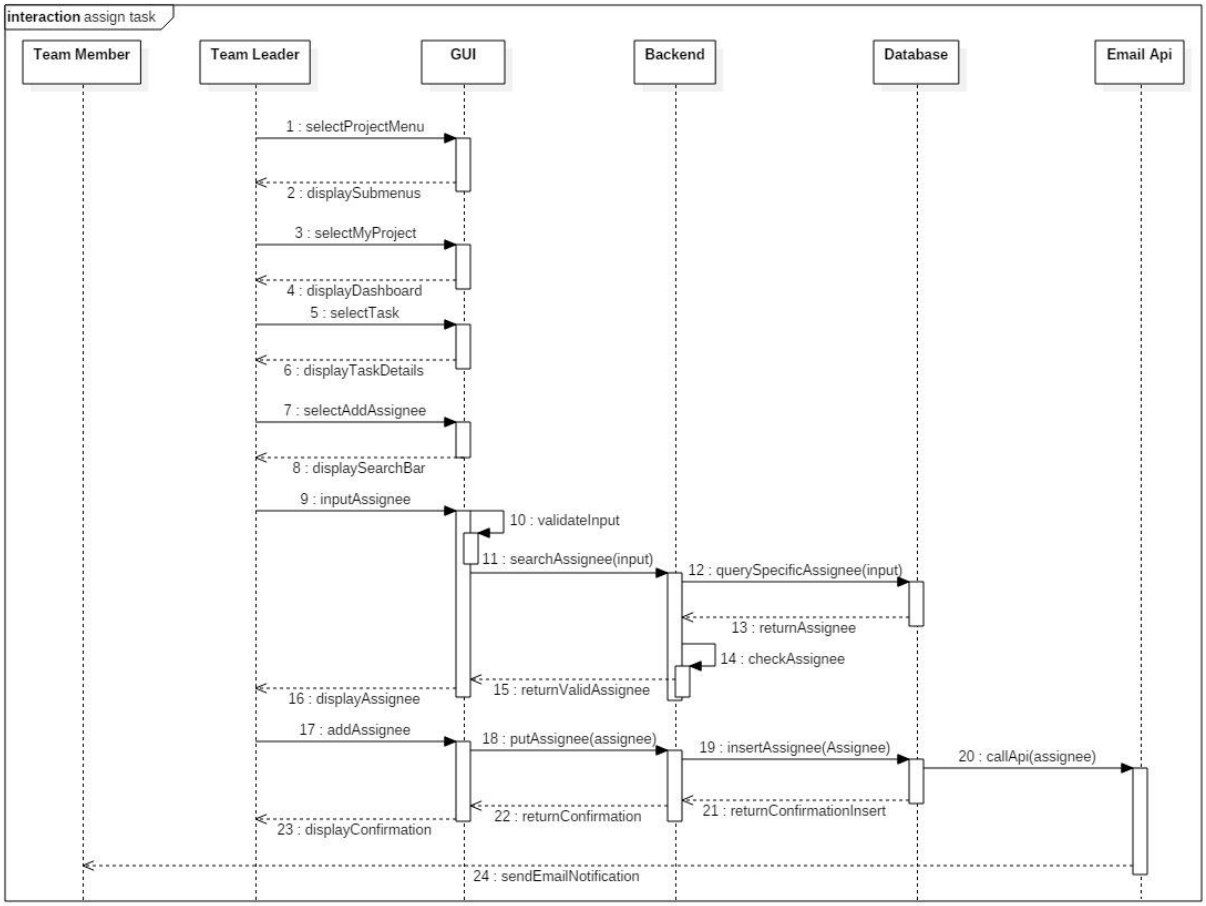
A user invites a member:



A user creates a task:

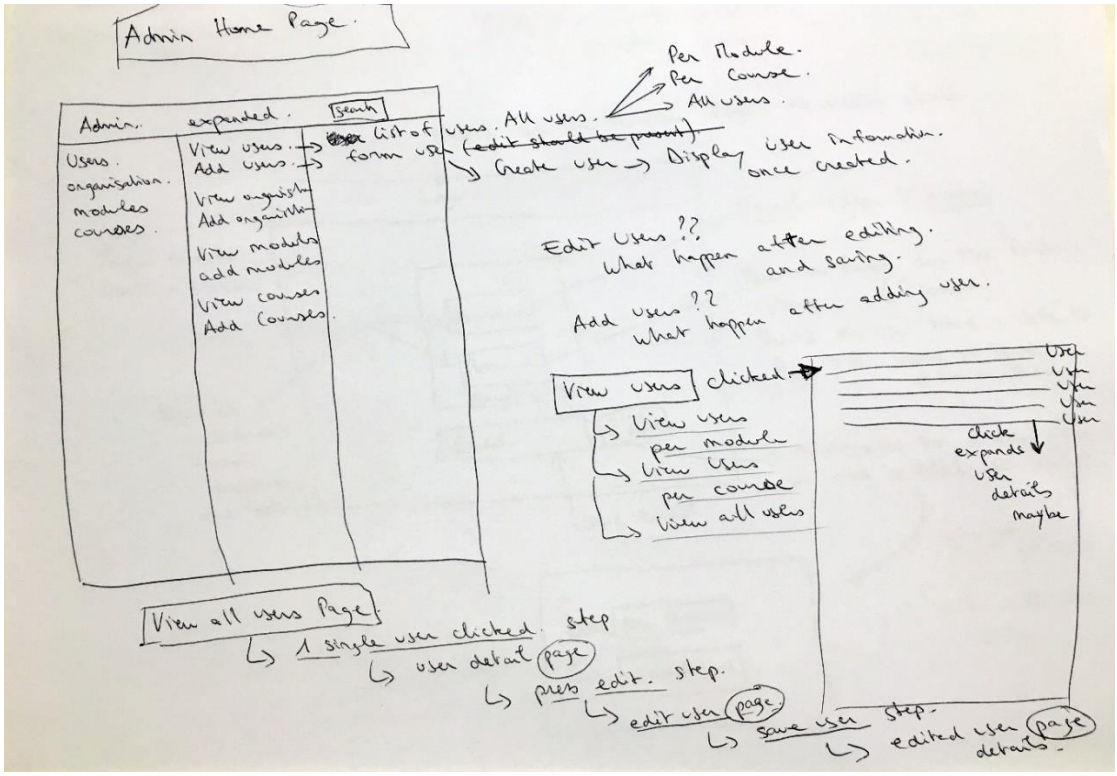


A user assigns a task:

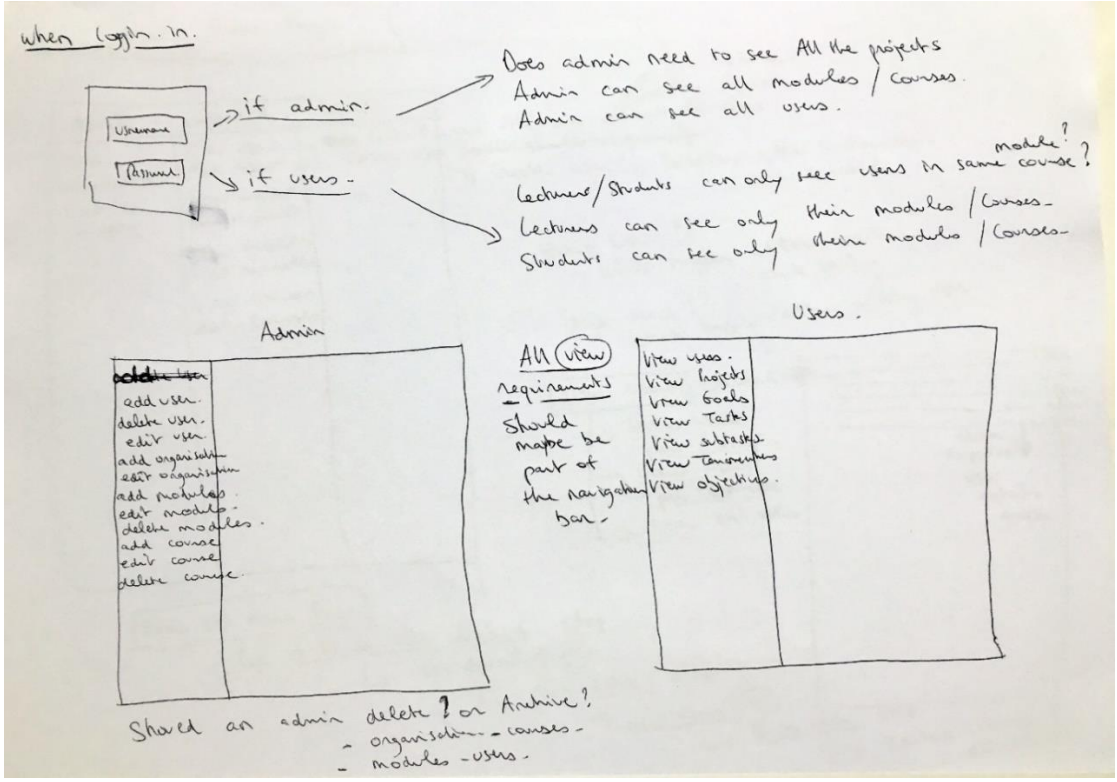


19 APPENDIX J - Sketches

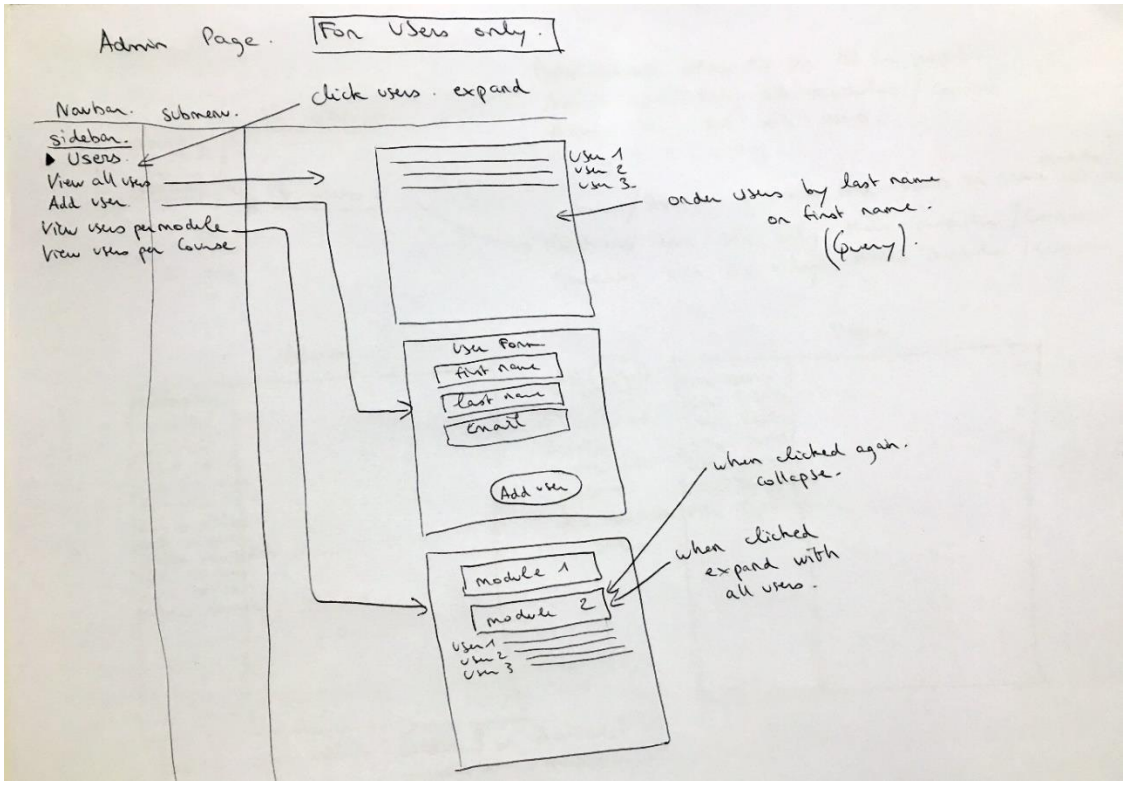
Administrator home page:



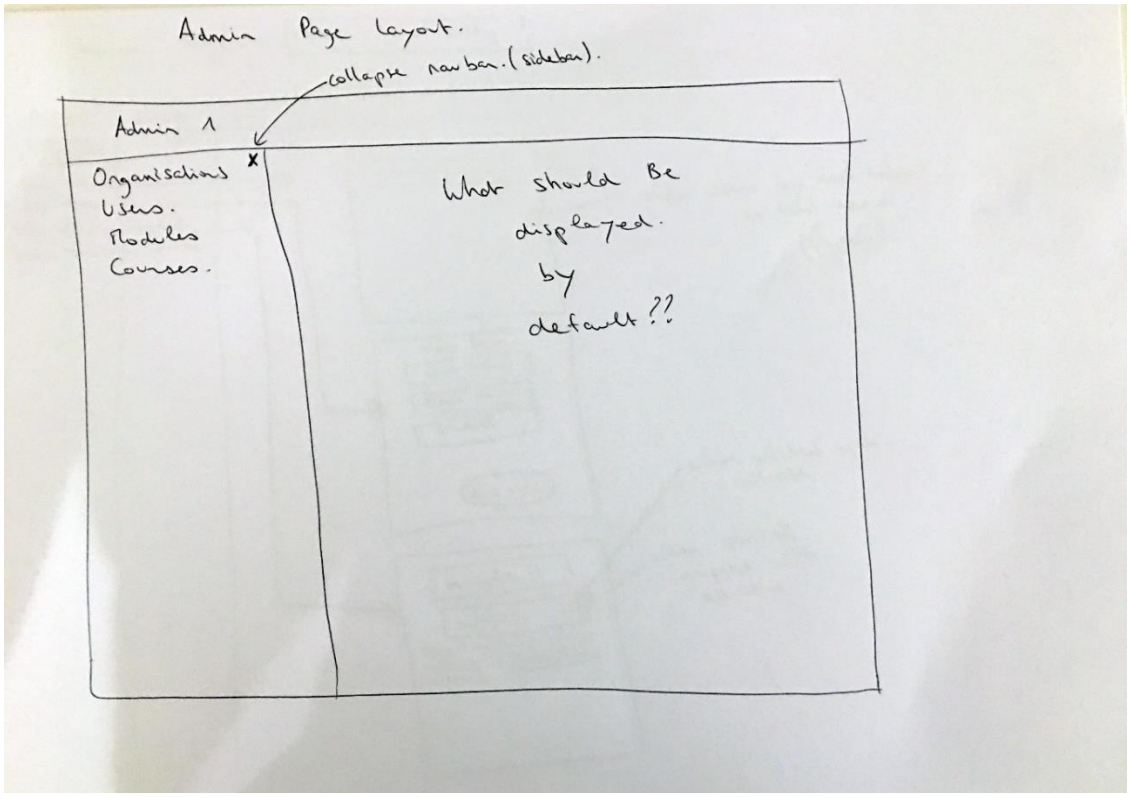
Users and administrators after logging in:



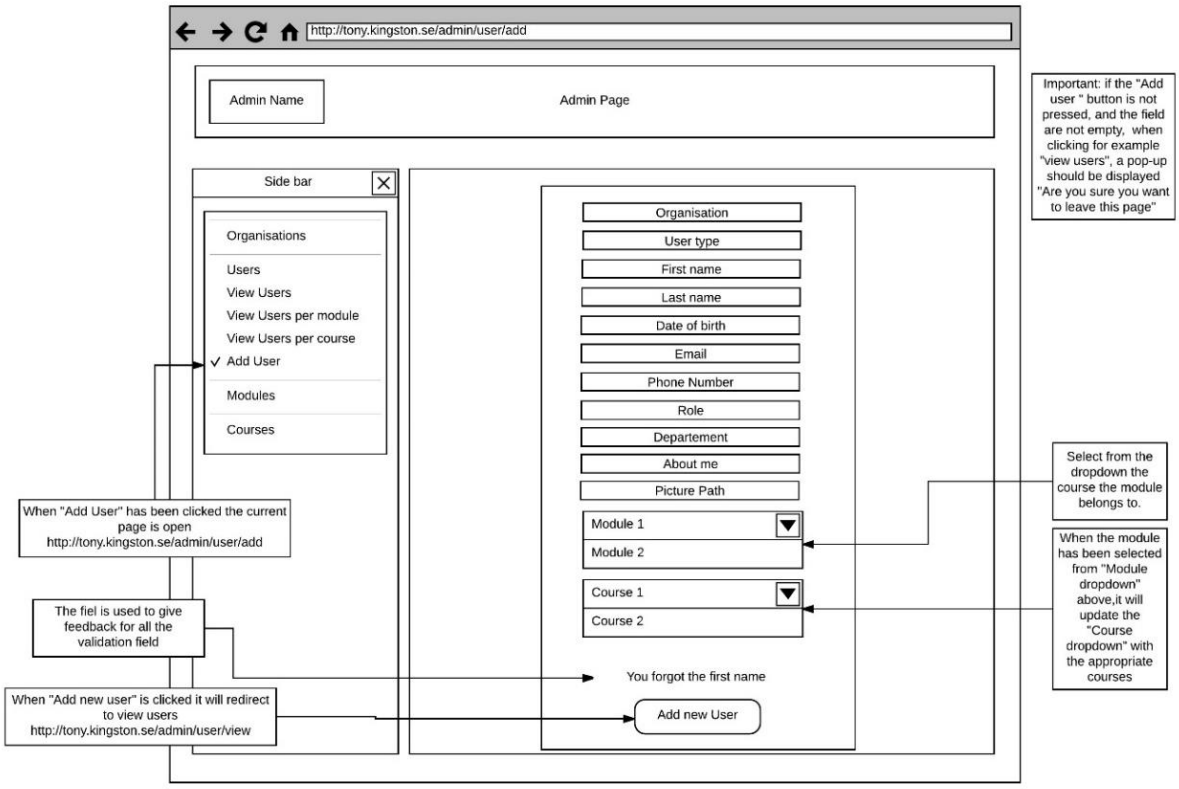
Main pages for users:

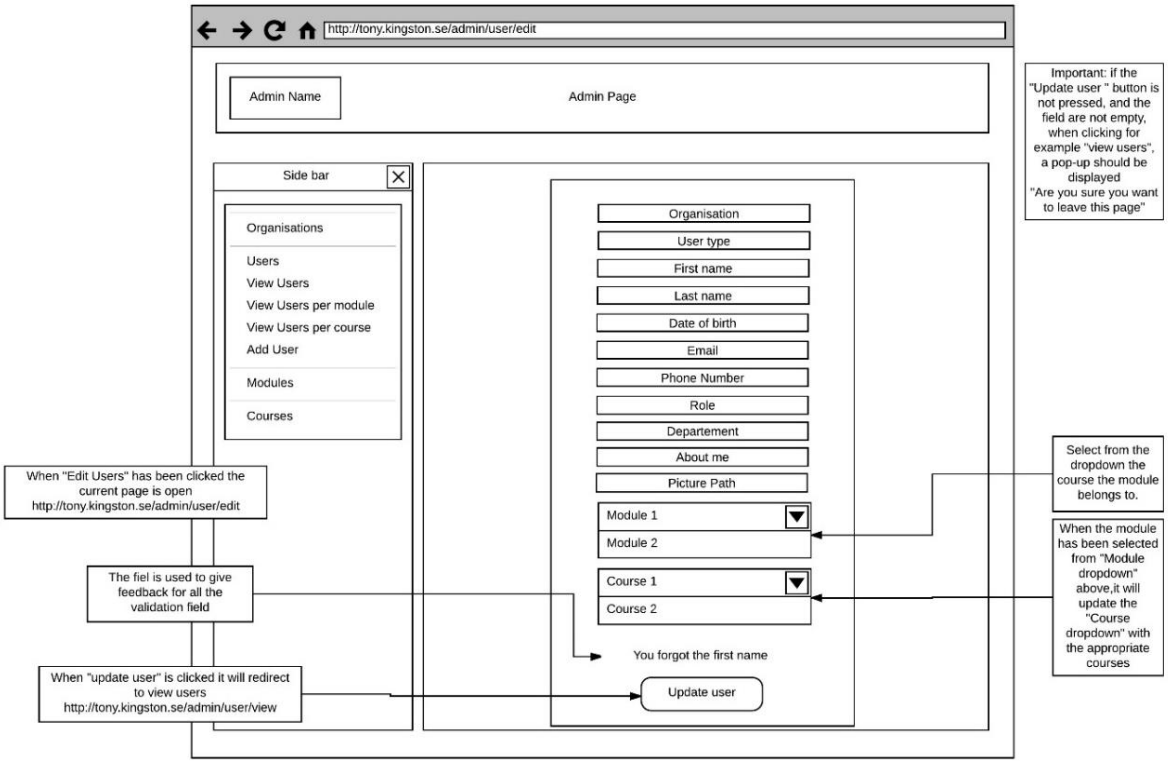


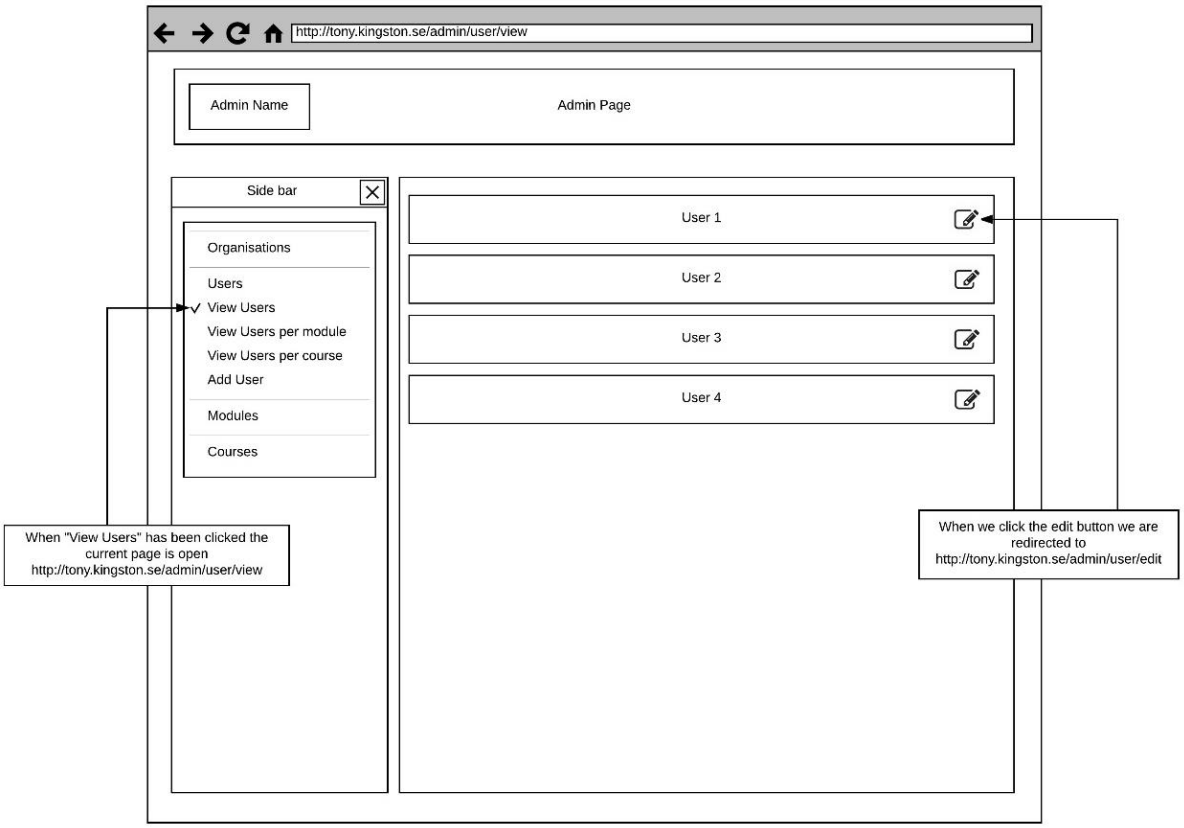
Admin page layout:

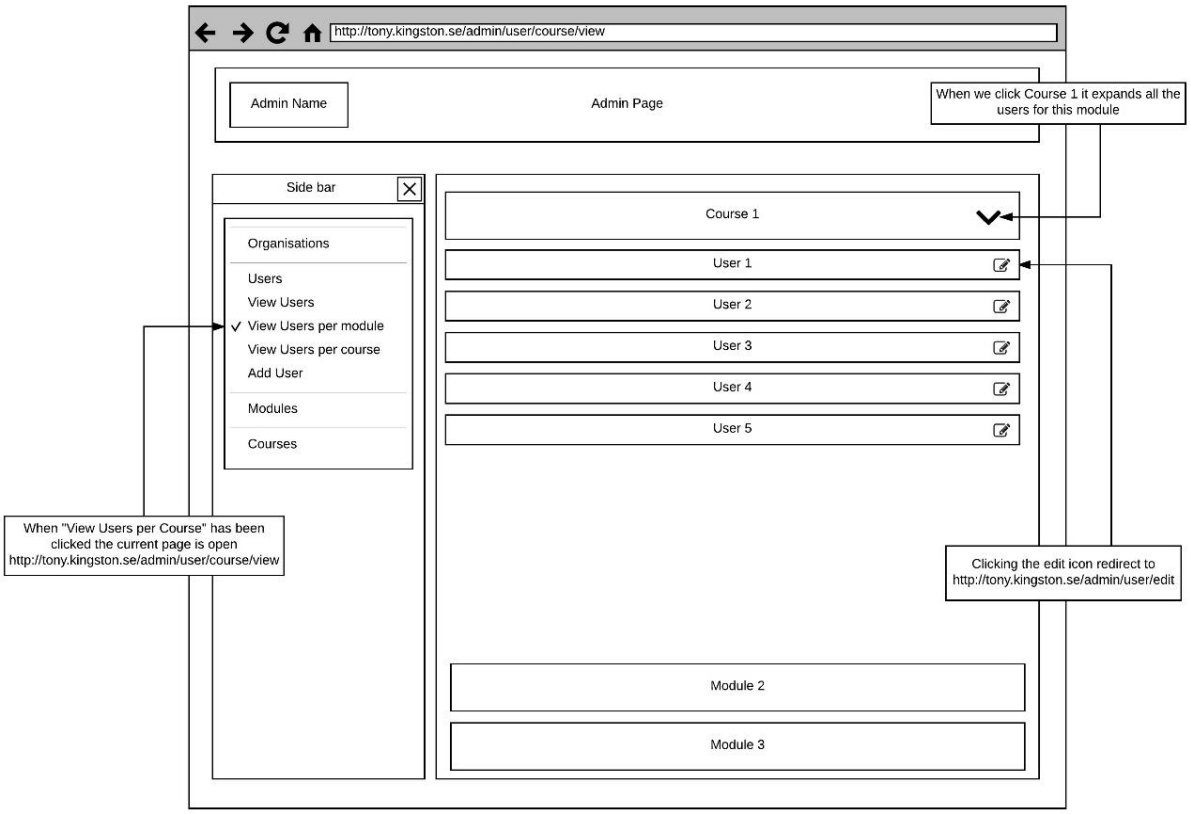


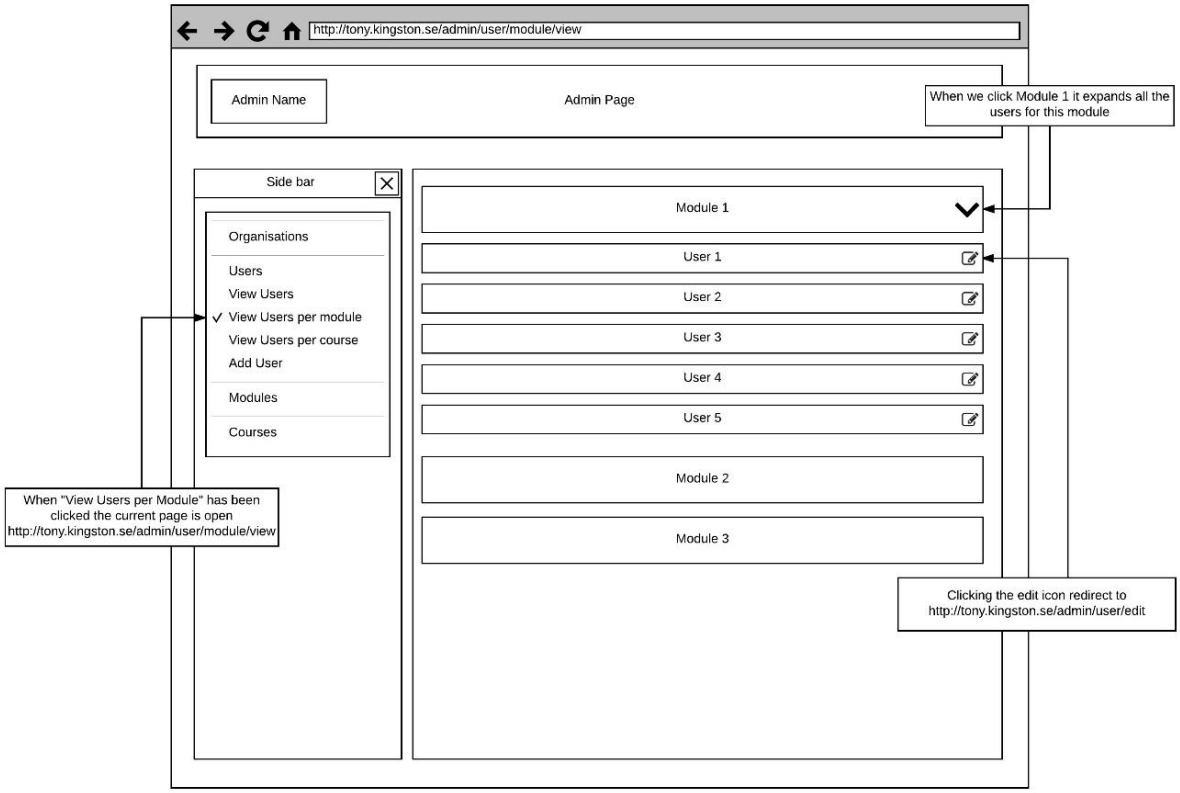
20 APPENDIX K - Wireframes

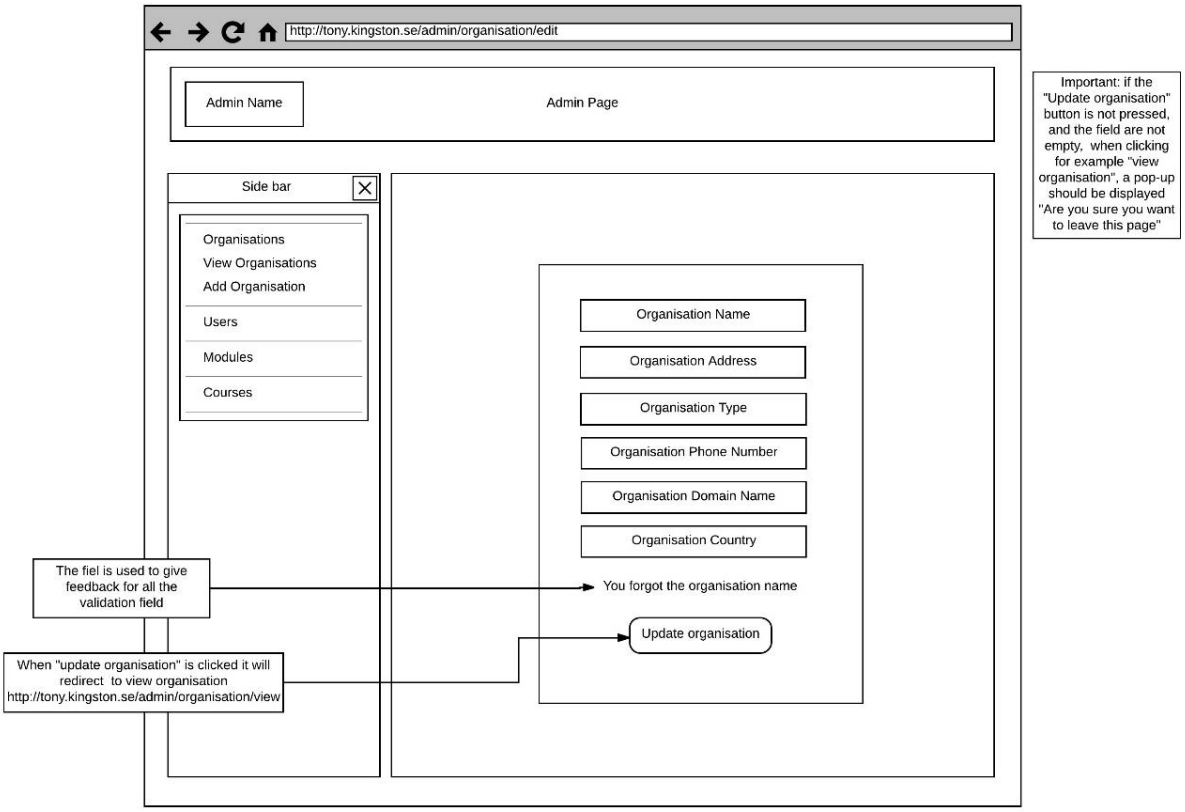


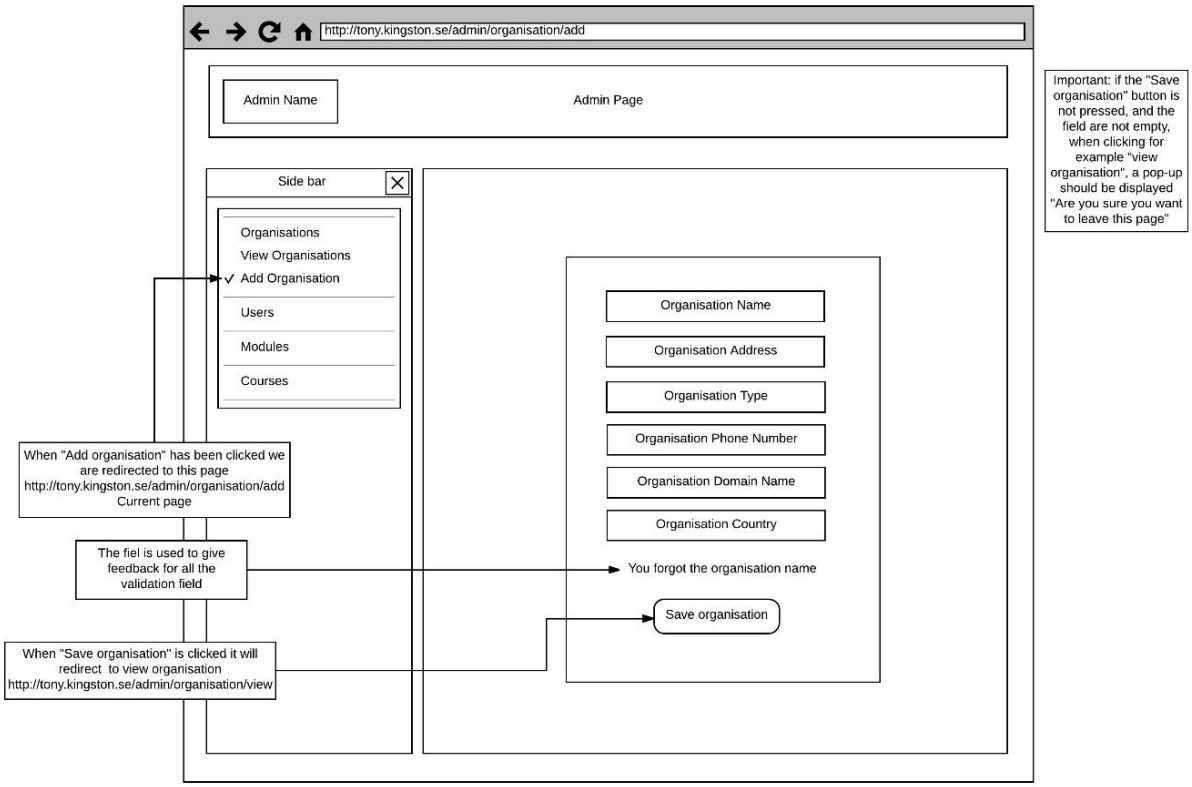


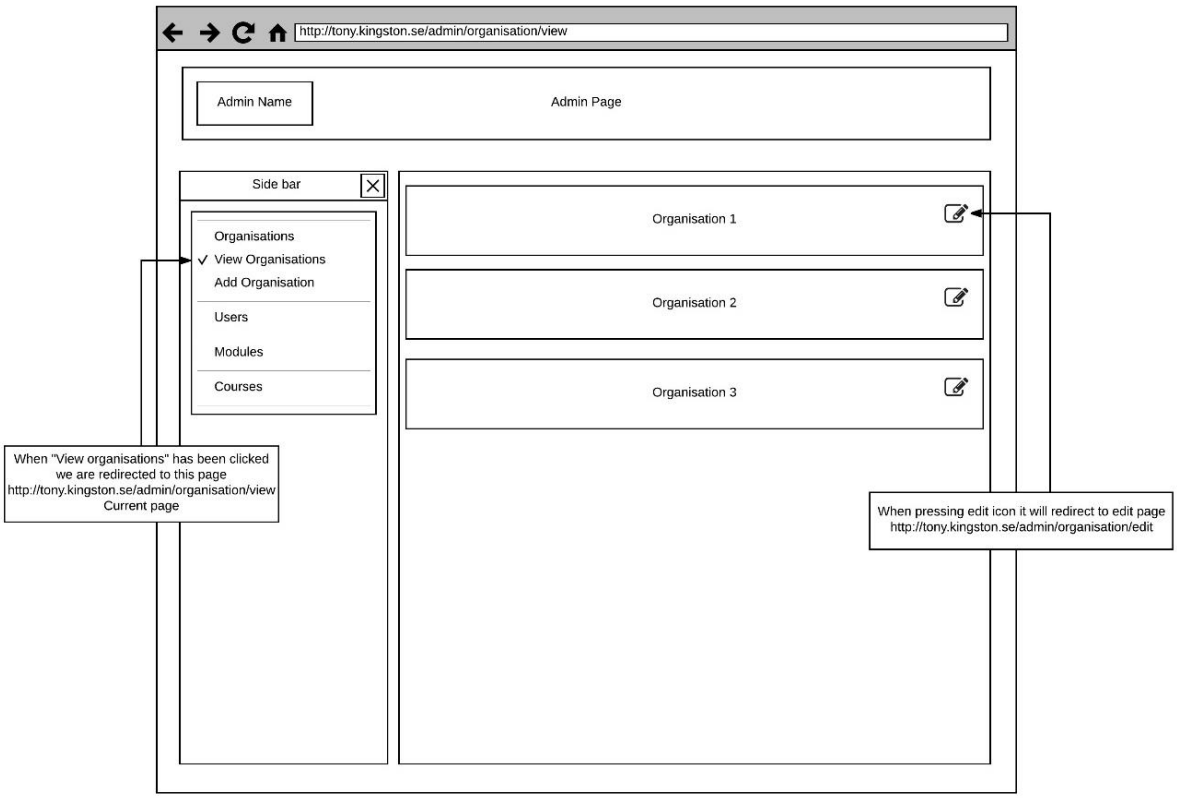

http://tony.kingston.se/admin/user/view
Admin Name
Admin Page
Side bar
Organisations
Users
View Users
View Users per module
View Users per course
Add User
Modules
Courses
User 1
User 2
User 3
User 4
When "View Users" has been clicked the current page is open http://tony.kingston.se/admin/user/view
When we click the edit button we are redirected to http://tony.kingston.se/admin/user/edit

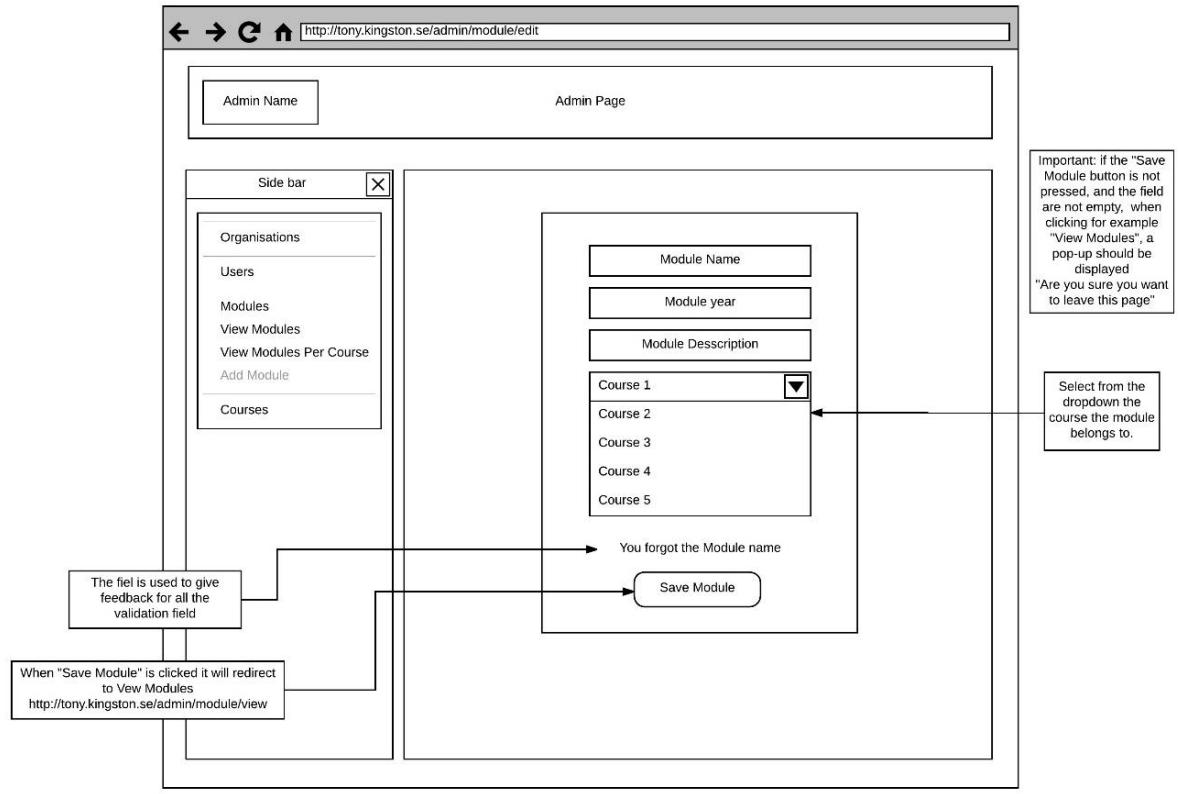

http://tony.kingston.se/admin/user/course/view
Admin Name
Admin Page
When we click Course 1 it expands all the users for this module
Side bar
Organisations
Users
View Users
View Users per module
View Users per course
Add User
Modules
Courses
Course 1
User 1
User 2
User 3
User 4
User 5
When "View Users per Course" has been clicked the current page is open http://tony.kingston.se/admin/user/course/view
Clicking the edit icon redirect to http://tony.kingston.se/admin/user/edit
Module 2
Module 3

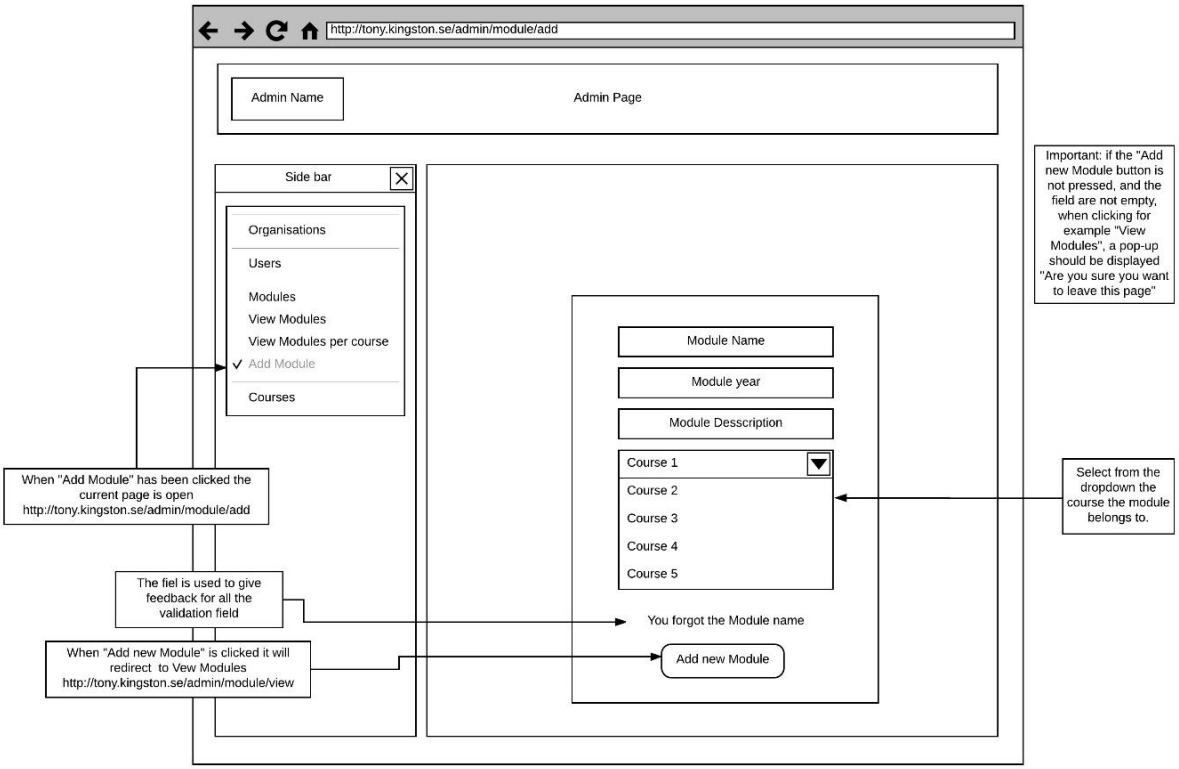

http://tony.kingston.se/admin/user/module/view
Admin Name
Admin Page
When we click Module 1 it expands all the users for this module
Side bar
Organisations
Users
View Users
View Users per module
View Users per course
Add User
Modules
Courses
Module 1
User 1
User 2
User 3
User 4
User 5
Module 2
Module 3
When "View Users per Module" has been clicked the current page is open http://tony.kingston.se/admin/user/module/view
Clicking the edit icon redirect to http://tony.kingston.se/admin/user/edit

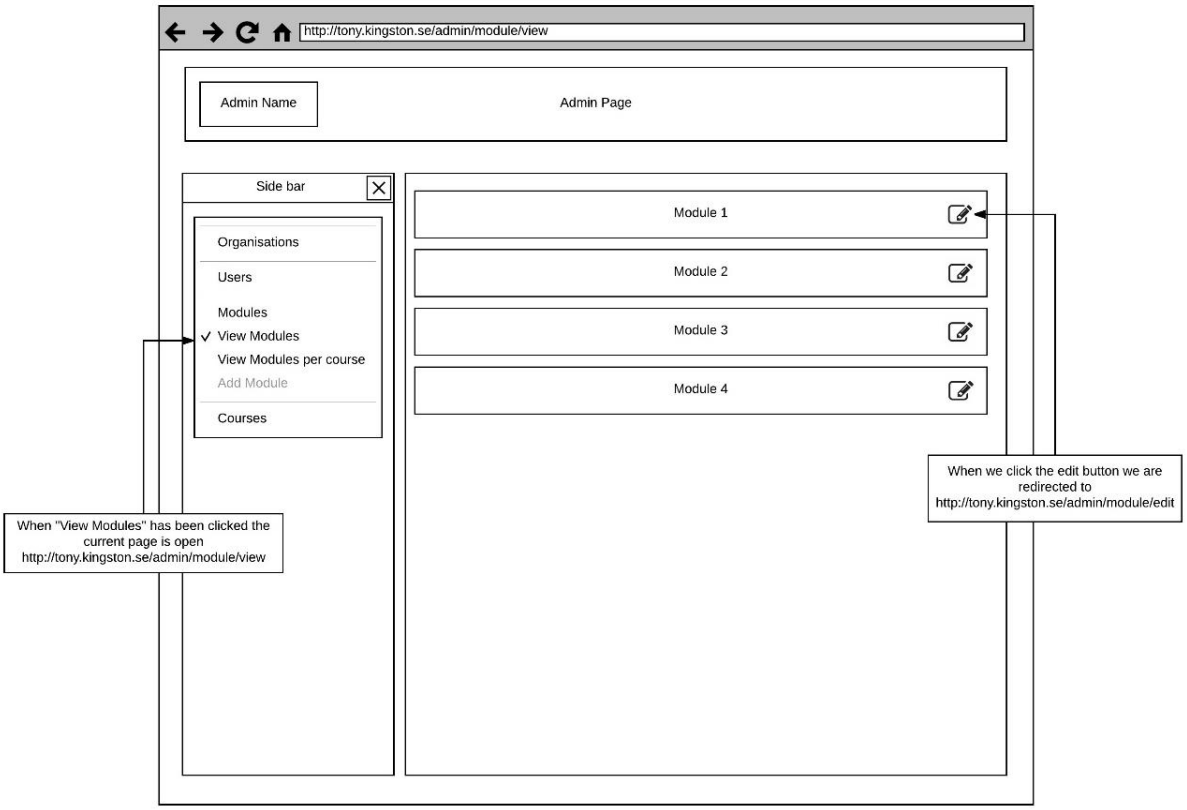

http://tony.kingston.se/admin/organisation/edit
Admin Name
Admin Page
Important: if the "Update organisation" button is not pressed, and the field are not empty, when clicking for example "view organisation", a pop-up should be displayed "Are you sure you want to leave this page"
Side bar
Organisations
View Organisations
Add Organisation
Users
Modules
Courses
Organisation Name
Organisation Address
Organisation Type
Organisation Phone Number
Organisation Domain Name
Organisation Country
You forgot the organisation name
Update organisation
The fiel is used to give feedback for all the validation field
When "update organisation" is clicked it will redirect to view organisation http://tony.kingston.se/admin/organisation/view

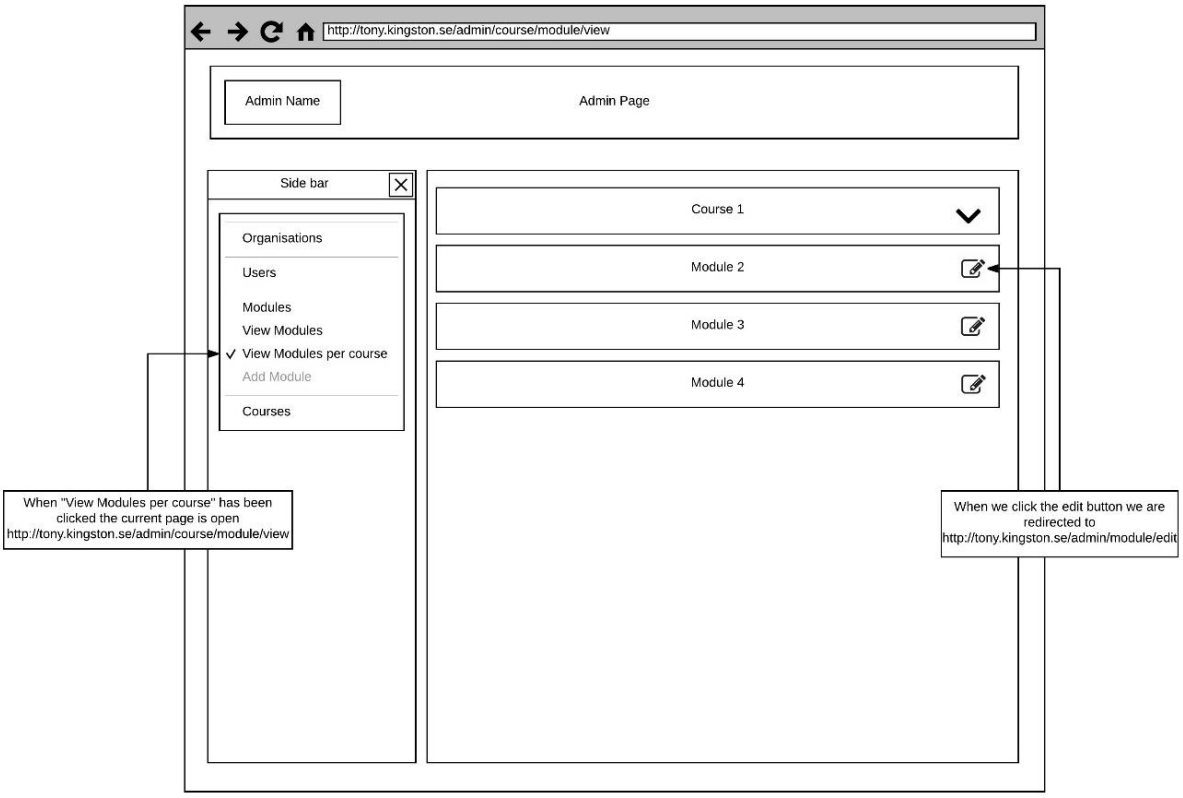

http://tony.kingston.se/admin/organisation/add
Admin Name
Admin Page
Important: if the "Save organisation" button is not pressed, and the field are not empty, when clicking for example "view organisation", a pop-up should be displayed "Are you sure you want to leave this page"
Side bar
Organisations
View Organisations
Add Organisation
Users
Modules
Courses
Organisation Name
Organisation Address
Organisation Type
Organisation Phone Number
Organisation Domain Name
Organisation Country
You forgot the organisation name
Save organisation
When "Add organisation" has been clicked we are redirected to this page http://tony.kingston.se/admin/organisation/add Current page
The fiel is used to give feedback for all the validation field
When "Save organisation" is clicked it will redirect to view organisation http://tony.kingston.se/admin/organisation/view

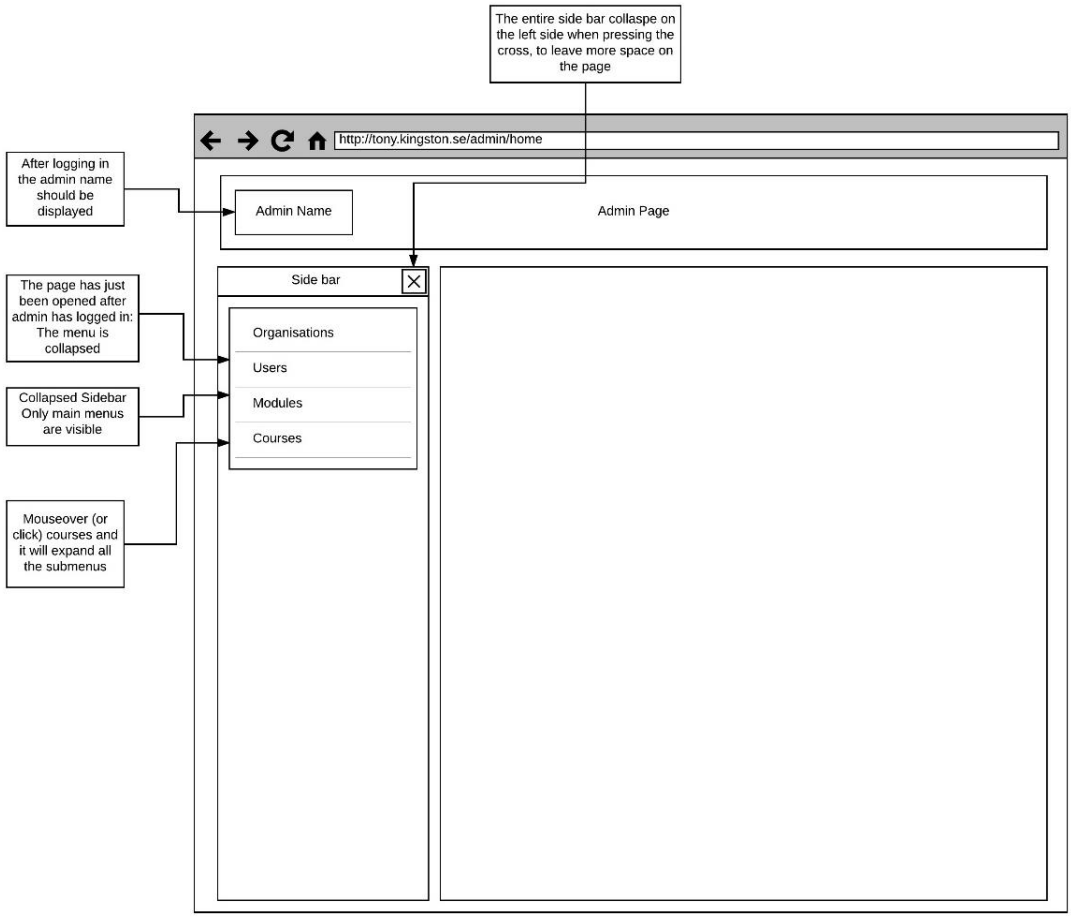

http://tony.kingston.se/admin/organisation/view
Admin Name
Admin Page
Side bar
Organisations
View Organisations
Add Organisation
Users
Modules
Courses
Organisation 1
Organisation 2
Organisation 3
When "View organisations" has been clicked we are redirected to this page http://tony.kingston.se/admin/organisation/view Current page
When pressing edit icon it will redirect to edit page http://tony.kingston.se/admin/organisation/edit

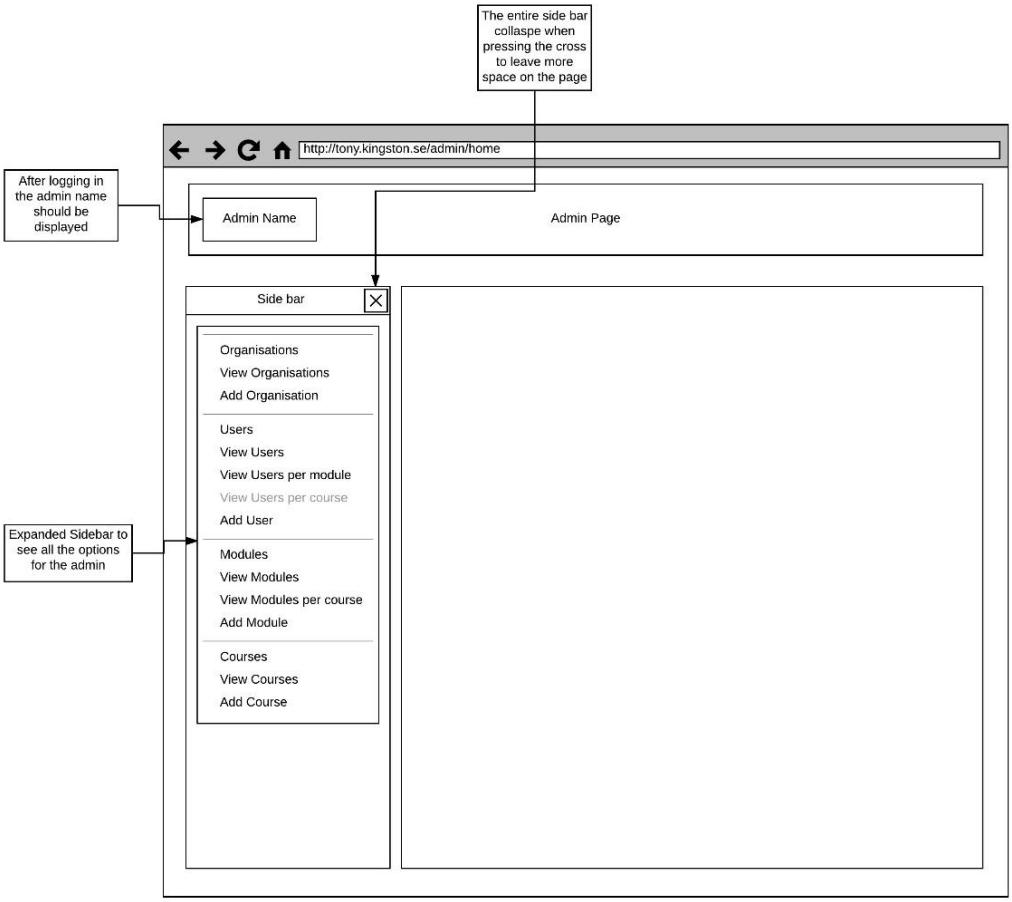

http://tony.kingston.se/admin/module/edit
Admin Name
Admin Page
Side bar
Organisations
Users
Modules
View Modules
View Modules Per Course
Add Module
Courses
Module Name
Module year
Module Desscription
Course 1
Course 2
Course 3
Course 4
Course 5
You forgot the Module name
Save Module
Important: if the "Save Module button is not pressed, and the field are not empty, when clicking for example "View Modules", a pop-up should be displayed "Are you sure you want to leave this page"
Select from the dropdown the course the module belongs to.
The fiel is used to give feedback for all the validation field
When "Save Module" is clicked it will redirect to Vew Modules http://tony.kingston.se/admin/module/view

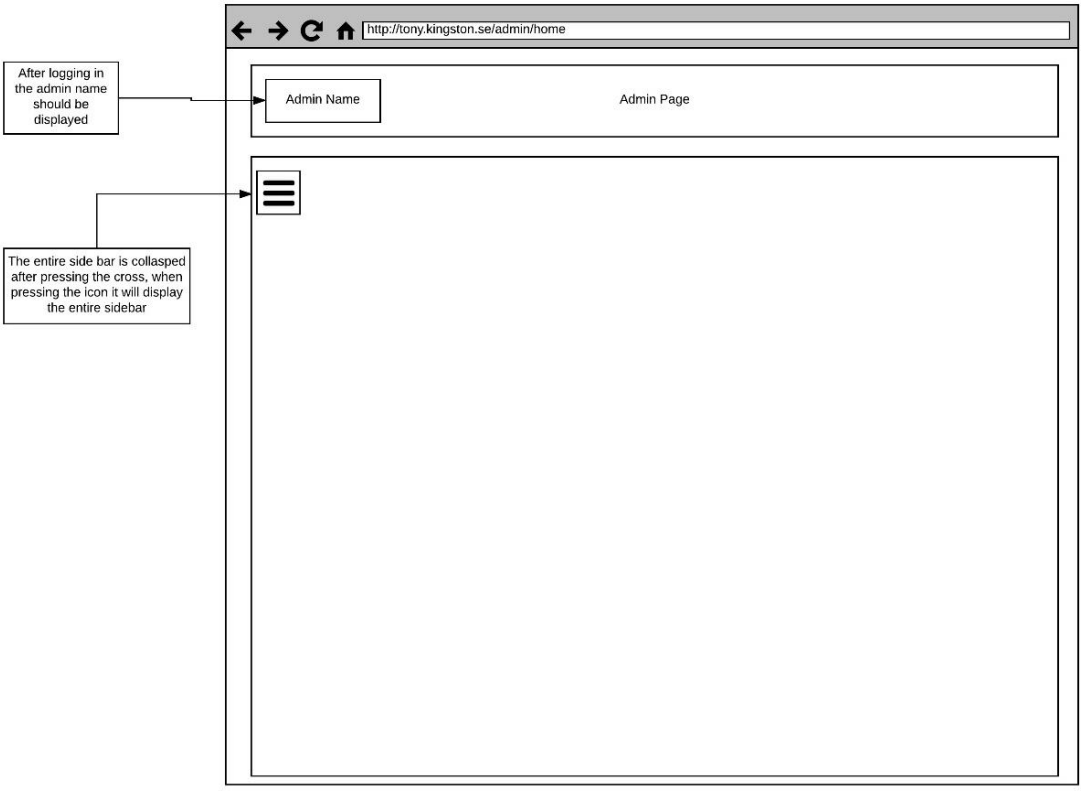

http://tony.kingston.se/admin/module/add
Admin Name
Admin Page
Side bar
Organisations
Users
Modules
View Modules
View Modules per course
Add Module
Courses
Module Name
Module year
Module Desscription
Course 1
Course 2
Course 3
Course 4
Course 5
You forgot the Module name
Add new Module
Important: if the "Add new Module button is not pressed, and the field are not empty, when clicking for example "View Modules", a pop-up should be displayed "Are you sure you want to leave this page"
Select from the dropdown the course the module belongs to.
When "Add Module" has been clicked the current page is open http://tony.kingston.se/admin/module/add
The fiel is used to give feedback for all the validation field
When "Add new Module" is clicked it will redirect to Vew Modules http://tony.kingston.se/admin/module/view

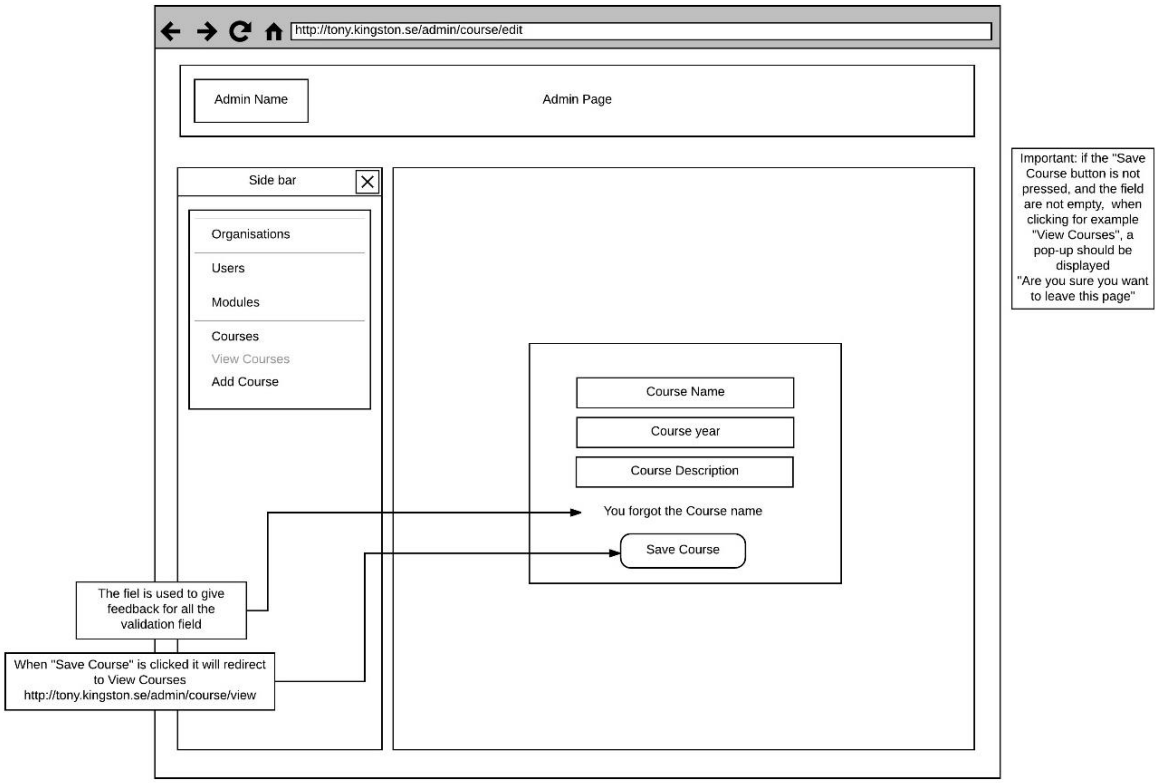

http://tony.kingston.se/admin/module/view
Admin Name
Admin Page
Side bar
Organisations
Users
Modules
View Modules
View Modules per course
Add Module
Courses
Module 1
Module 2
Module 3
Module 4
When "View Modules" has been clicked the current page is open http://tony.kingston.se/admin/module/view
When we click the edit button we are redirected to http://tony.kingston.se/admin/module/edit

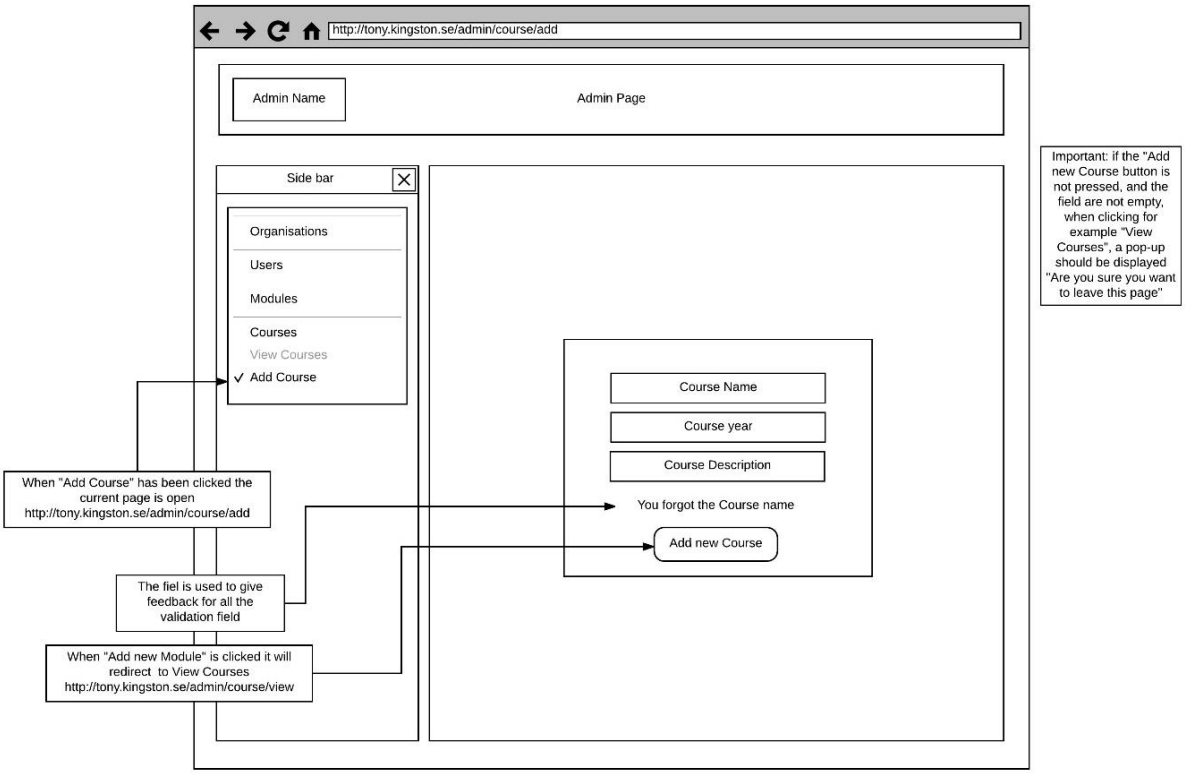

http://tony.kingston.se/admin/course/module/view
Admin Name
Admin Page
Side bar
Organisations
Users
Modules
View Modules
View Modules per course
Add Module
Courses
Course 1
Module 2
Module 3
Module 4
When "View Modules per course" has been clicked the current page is open http://tony.kingston.se/admin/course/module/view
When we click the edit button we are redirected to http://tony.kingston.se/admin/module/edit

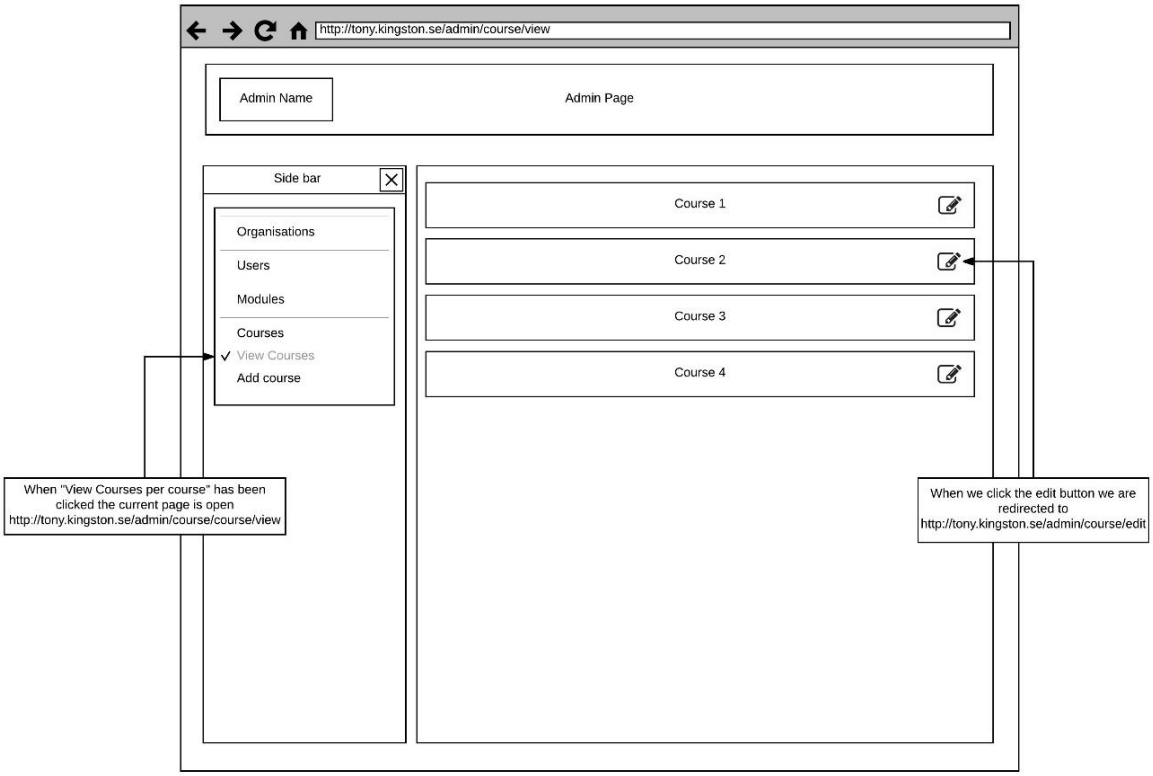

The entire side bar collaspe on the left side when pressing the cross, to leave more space on the page
http://tony.kingston.se/admin/home
After logging in the admin name should be displayed
Admin Name
Admin Page
Side bar
The page has just been opened after admin has logged in: The menu is collapsed
Organisations
Users
Modules
Courses
Collapsed Sidebar Only main menus are visible
Mouseover (or click) courses and it will expand all the submenus

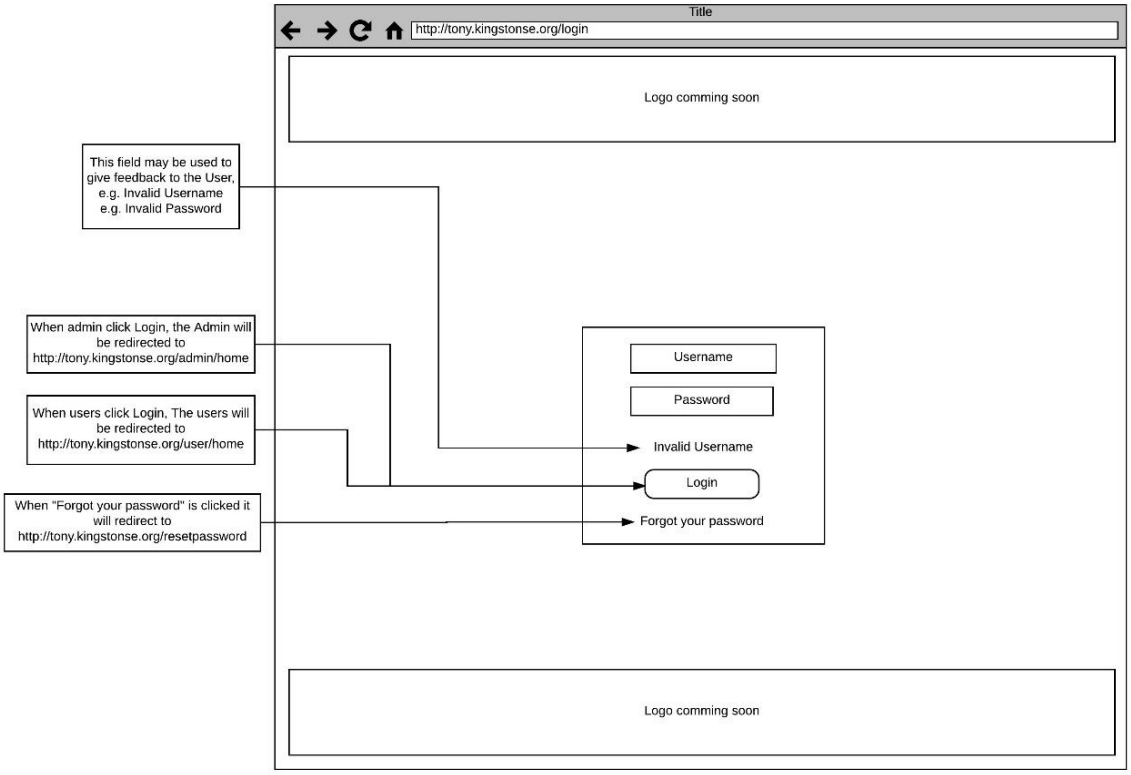

The entire side bar collaspe when pressing the cross to leave more space on the page
http://tony.kingston.se/admin/home
After logging in the admin name should be displayed
Admin Name
Admin Page
Side bar
Organisations
View Organisations
Add Organisation
Users
View Users
View Users per module
View Users per course
Add User
Modules
View Modules
View Modules per course
Add Module
Courses
View Courses
Add Course
Expanded Sidebar to see all the options for the admin

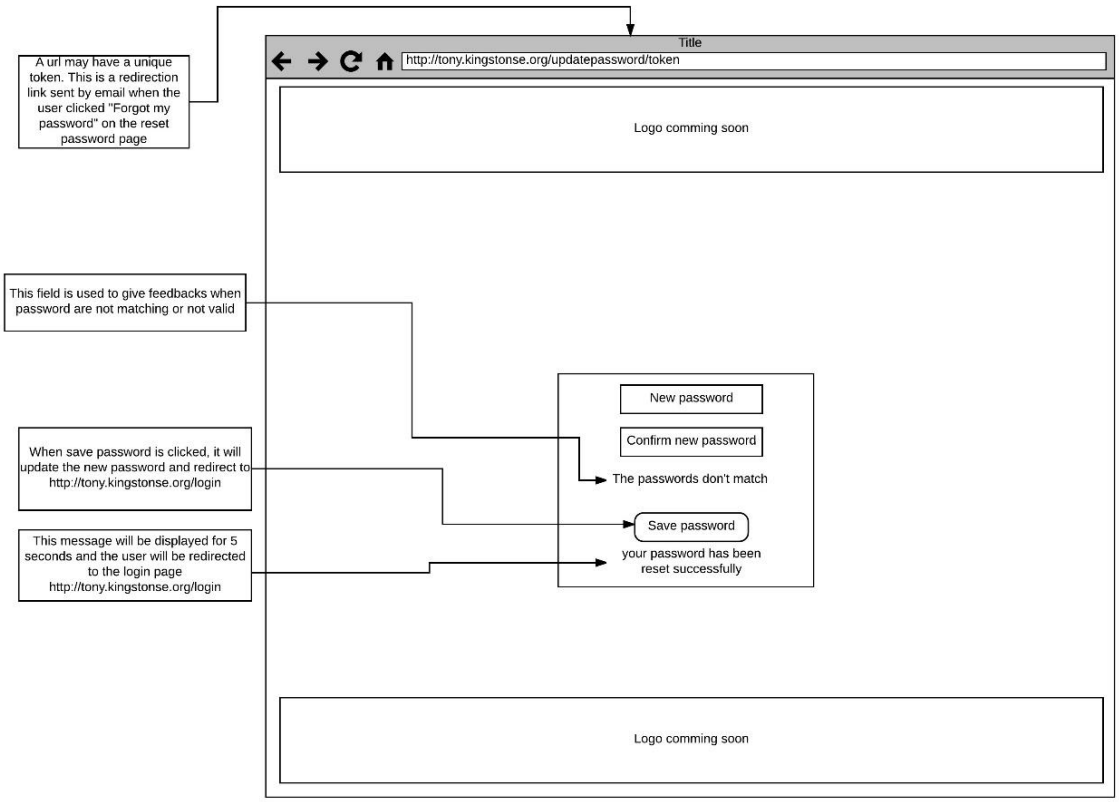

http://tony.kingston.se/admin/home
After logging in the admin name should be displayed
Admin Name
Admin Page
The entire side bar is collasped after pressing the cross, when pressing the icon it will display the entire sidebar

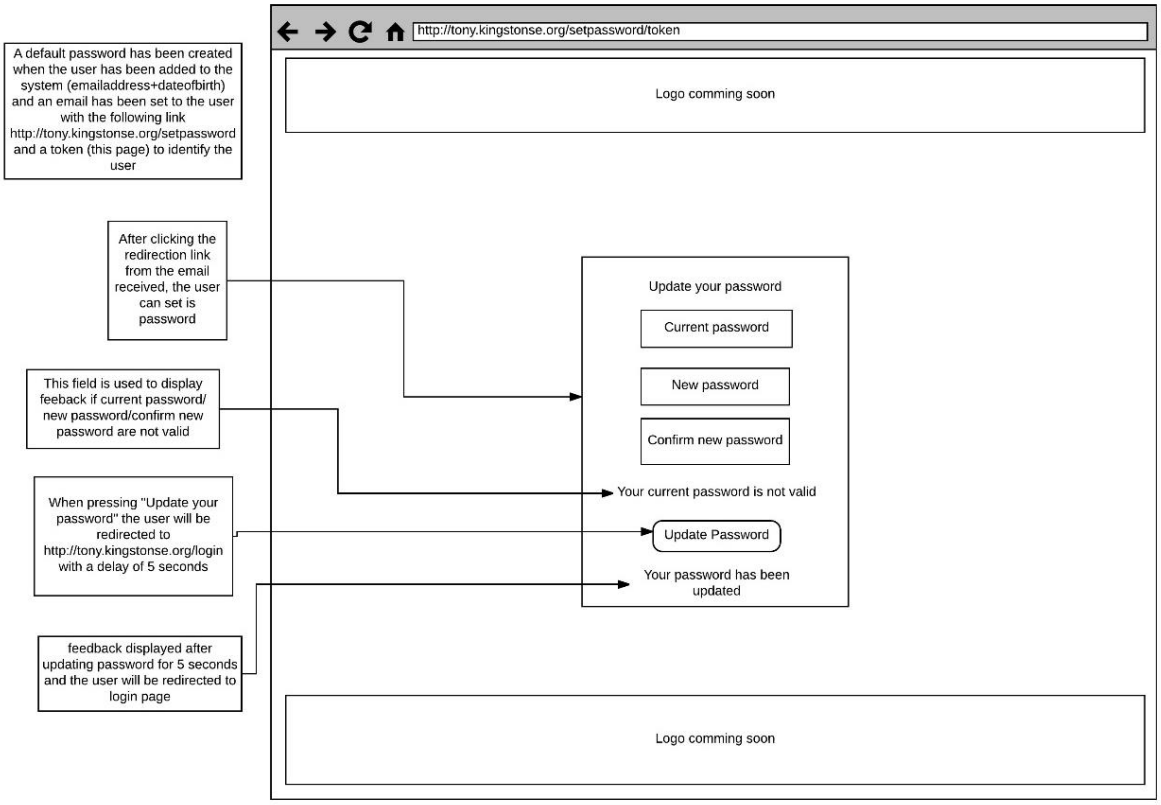

http://tony.kingston.se/admin/course/edit
Admin Name
Admin Page
Side bar
Organisations
Users
Modules
Courses
View Courses
Add Course
Course Name
Course year
Course Description
You forgot the Course name
Save Course
Important: if the "Save Course button is not pressed, and the field are not empty, when clicking for example "View Courses", a pop-up should be displayed "Are you sure you want to leave this page"
The fiel is used to give feedback for all the validation field
When "Save Course" is clicked it will redirect to View Courses http://tony.kingston.se/admin/course/view

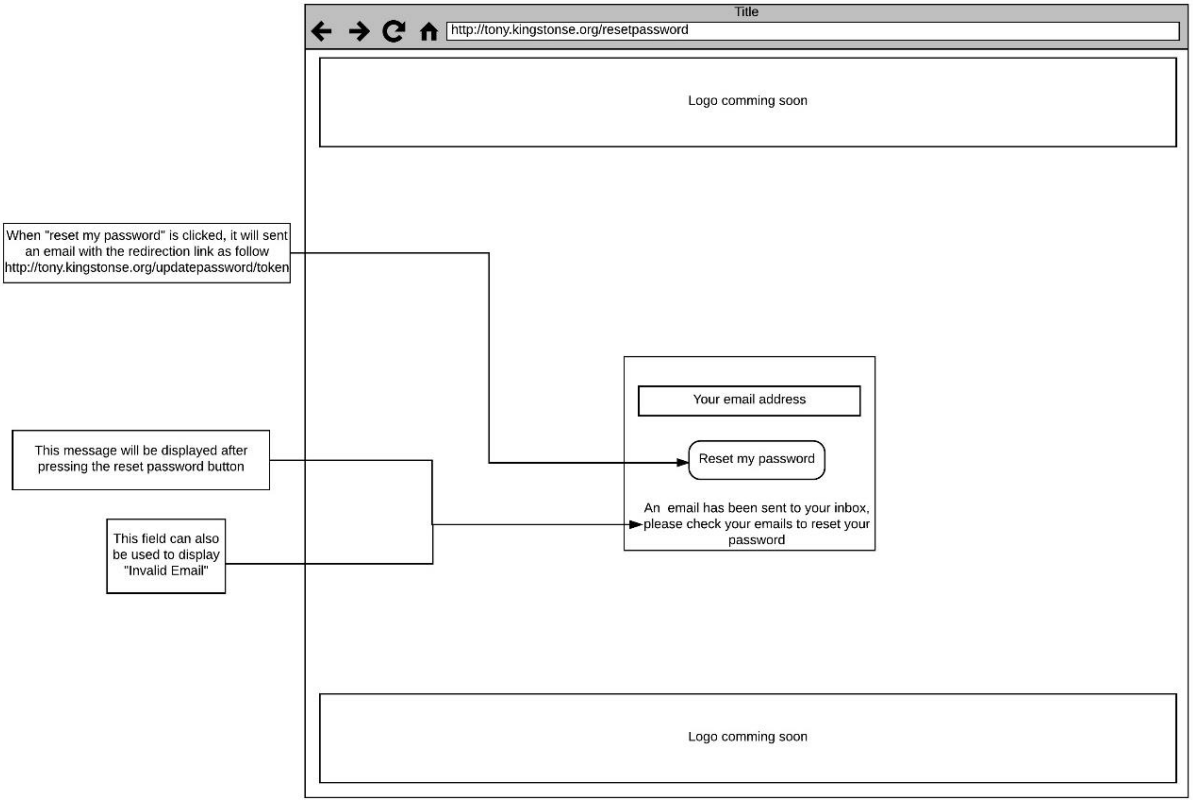

http://tony.kingston.se/admin/course/add
Admin Name
Admin Page
Side bar
Organisations
Users
Modules
Courses
View Courses
Add Course
Course Name
Course year
Course Description
You forgot the Course name
Add new Course
Important: if the "Add new Course button is not pressed, and the field are not empty, when clicking for example "View Courses", a pop-up should be displayed "Are you sure you want to leave this page"
When "Add Course" has been clicked the current page is open http://tony.kingston.se/admin/course/add
The fiel is used to give feedback for all the validation field
When "Add new Module" is clicked it will redirect to View Courses http://tony.kingston.se/admin/course/view

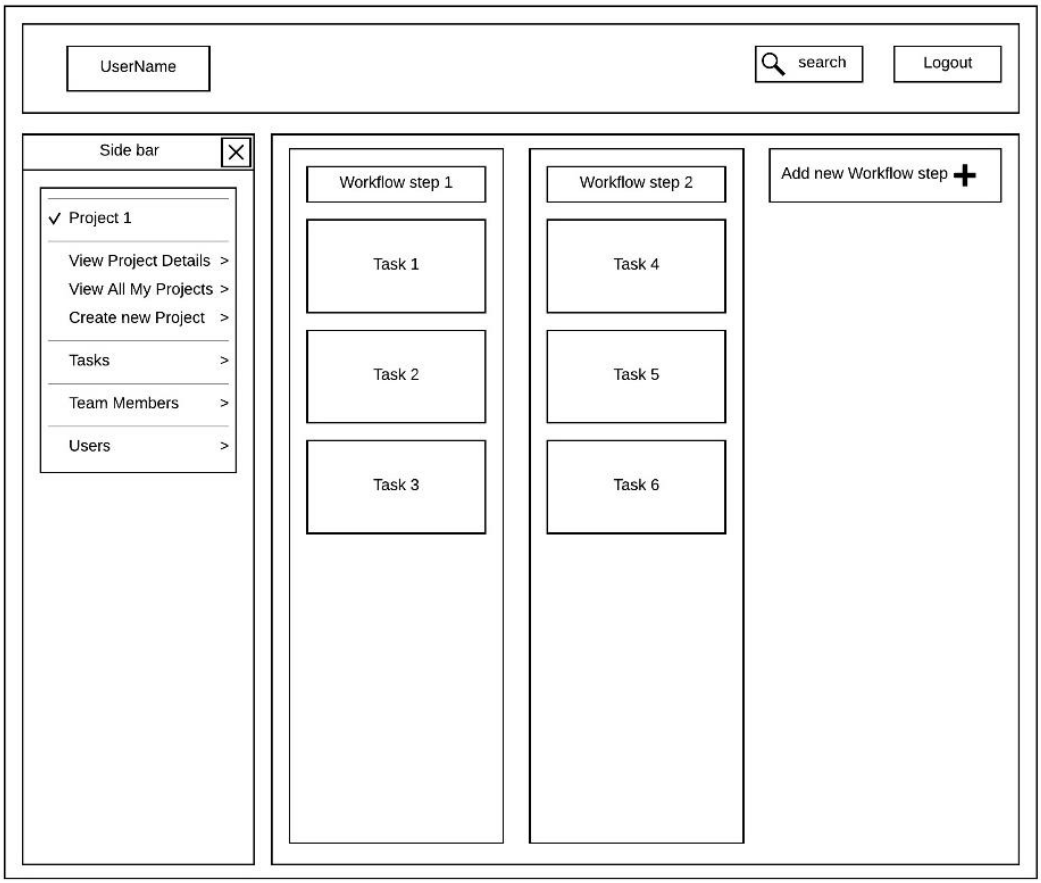

http://tony.kingston.se/admin/course/view
Admin Name
Admin Page
Side bar
Organisations
Users
Modules
Courses
View Courses
Add course
Course 1
Course 2
Course 3
Course 4
When "View Courses per course" has been clicked the current page is open http://tony.kingston.se/admin/course/course/view
When we click the edit button we are redirected to http://tony.kingston.se/admin/course/edit

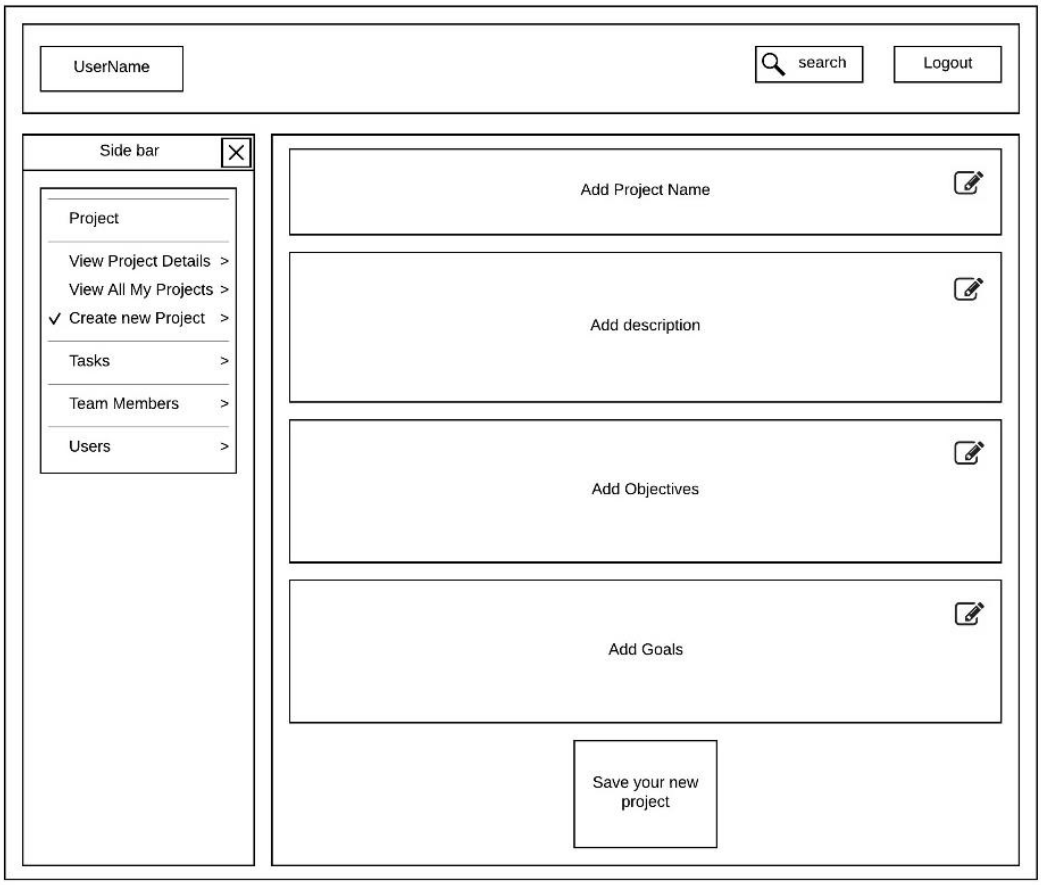

Title
http://tony.kingstonse.org/login
Logo comming soon
This field may be used to give feedback to the User, e.g. Invalid Username e.g. Invalid Password
When admin click Login, the Admin will be redirected to http://tony.kingstonse.org/admin/home
When users click Login, The users will be redirected to http://tony.kingstonse.org/user/home
When "Forgot your password" is clicked it will redirect to http://tony.kingstonse.org/resetpassword
Username
Password
Invalid Username
Login
Forgot your password
Logo comming soon

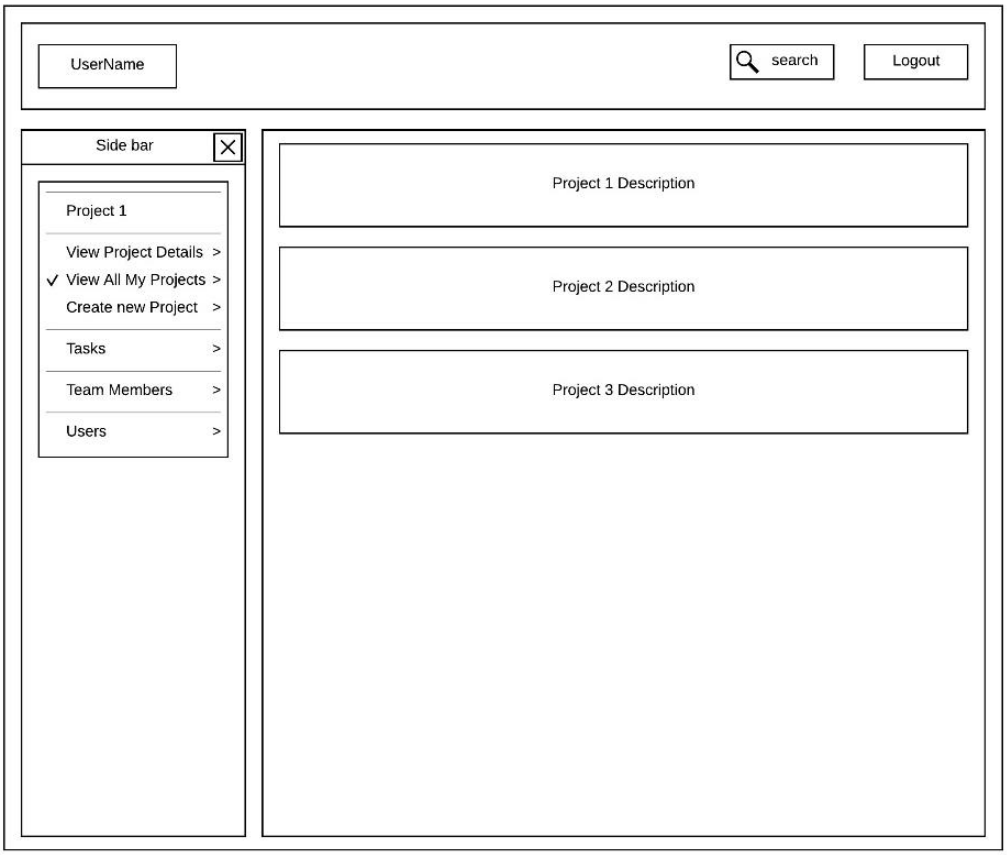

Title
http://tony.kingstonse.org/updatepassword/token
A url may have a unique token. This is a redirection link sent by email when the user clicked "Forgot my password" on the reset password page
Logo comming soon
This field is used to give feedbacks when password are not matching or not valid
When save password is clicked, it will update the new password and redirect to http://tony.kingstonse.org/login
This message will be displayed for 5 seconds and the user will be redirected to the login page http://tony.kingstonse.org/login
New password
Confirm new password
The passwords don't match
Save password
your password has been reset successfully
Logo comming soon


http://tony.kingstonse.org/setpassword/token
A default password has been created when the user has been added to the system (emailaddress+dateofbirth) and an email has been set to the user with the following link http://tony.kingstonse.org/setpassword and a token (this page) to identify the user
Logo comming soon
After clicking the redirection link from the email received, the user can set is password
Update your password
Current password
New password
Confirm new password
This field is used to display feeback if current password/ new password/confirm new password are not valid
Your current password is not valid
When pressing "Update your password" the user will be redirected to http://tony.kingstonse.org/login with a delay of 5 seconds
Update Password
Your password has been updated
feedback displayed after updating password for 5 seconds and the user will be redirected to login page
Logo comming soon


Title
http://tony.kingstonse.org/resetpassword
Logo comming soon
When "reset my password" is clicked, it will sent an email with the redirection link as follow http://tony.kingstonse.org/updatepassword/token
Your email address
Reset my password
This message will be displayed after pressing the reset password button
An email has been sent to your inbox, please check your emails to reset your password
This field can also be used to display "Invalid Email"
Logo comming soon


UserName
search
Logout
Side bar
Project 1
View Project Details >
View All My Projects >
Create new Project >
Tasks >
Team Members >
Users >
Workflow step 1
Task 1
Task 2
Task 3
Workflow step 2
Task 4
Task 5
Task 6
Add new Workflow step


UserName
search
Logout
Side bar
Project
View Project Details >
View All My Projects >
Create new Project >
Tasks >
Team Members >
Users >
Add Project Name
Add description
Add Objectives
Add Goals
Save your new project


UserName
search
Logout
Side bar
Project 1
View Project Details >
View All My Projects >
Create new Project >
Tasks >
Team Members >
Users >
Project 1 Description
Project 2 Description
Project 3 Description

UserName
search
Logout
Side bar
Project 1
View Project Details >
View All My Projects >
Create new Project >
Tasks >
Team Members >
Users >
Status
Project 1 Description
Project 1 Objectives
Objective 1
Objective 2
Objective 3
Project 1 Goals
Goal 1
Goal 2
Goal 3
Comments

21 APPENDIX L – Complex queries

Return all the tasks titles and subtasks titles for the project id = 1

SELECT task.task_title from users

INNER join users_team

on users.user_id = users_team.user_id

INNER join team

on users_team.team_id = team.team_id

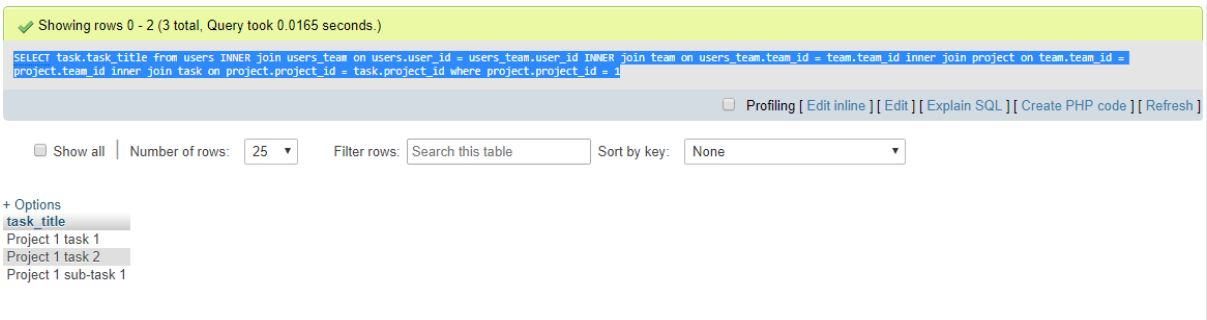
inner join project

on team.team_id = project.team_id

inner join task

on project.project_id =task.project_id

where project.project_id = 1;



Return the first name and last name of all the users currently in the module "Contextualising Contemporary Practice: Fine Art"

SELECT users.user_id,

users.user_first_name,

users.user_last_name

FROM users

INNER JOIN users_course

ON users.user_id = users_course.user_id

INNER JOIN course

ON users_course.course_id = course.course_id

INNER JOIN module_course

ON course.course_id = module_course.course_id

INNER JOIN module

ON module_course.module_id = module.module_id

WHERE module.module_name = "Contextualising Contemporary Practice: Fine Art"

Showing rows 0 - 0 (1 total, Query took 0.0059 seconds.)

```
SELECT users.user_id, users.user_first_name, users.user_last_name from users INNER JOIN users_course ON users.user_id = users_course.user_id INNER JOIN course ON users_course.course_id = course.course_id INNER JOIN module_course ON
course.course_id = module_course.course_id INNER JOIN module ON module_course.module_id = module.module_id WHERE module.module_name = "Contextualising Contemporary Practice: Fine Art"
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [

Show all | Number of rows: 25 Filter rows: Search this table

+ Options

user_id	user_first_name	user_last_name
3	Graeme	Jones

Show all | Number of rows: 25 Filter rows: Search this table

Query results operations

INNER JOIN

SELECT assignee.assignee_name assigneeName,

task.task_title taskTitle

FROM assignee

INNER JOIN task

ON assignee.assignee_id = task.assignee_id;

Showing rows 0 - 1 (2 total, Query took 0.0030 seconds.)

```
SELECT assignee.assignee_name assigneeName, task.task_title taskTitle FROM assignee INNER JOIN task ON assignee.assignee_id = task.assignee_id
```

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

assigneeName	taskTitle
tony	Project 1 task 1
roberto	Project 1 task 2

Only the records that are matching from both tables will be returned

RIGHT JOIN

SELECT assignee.assignee_name assigneeName,

task.task_title taskTitle

FROM assignee

RIGHT JOIN task

ON assignee.assignee_id = task.assignee_id;

Showing rows 0 - 5 (6 total, Query took 0.0028 seconds.)

```
SELECT assignee.assignee_name, task.task_title assName FROM assignee right join task on assignee.assignee_id = task.assignee_id
```

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

assignee_name	assName
tony	Project 1 task 1
NULL	Project 1 task 1
roberto	Project 1 task 2
NULL	Project 2 task 2
NULL	Project 1 sub-task 1
NULL	Project 2 sub-task 2

All the records from the Assignee table (RIGHT table) will be returned even the non-matching one

LEFT JOIN

SELECT assignee.assignee_name assigneeName,

task.task_title taskTitle FROM assignee

LEFT JOIN task

ON assignee.assignee_id = task.assignee_id

Showing rows 0 - 5 (6 total, Query took 0.0030 seconds.)

```
SELECT assignee.assignee_name assigneeName, task.task_title taskTitle FROM assignee LEFT JOIN task ON assignee.assignee_id = task.assignee_id
```

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

assigneeName	taskTitle
tony	Project 1 task 1
james	NULL
roberto	Project 1 task 2
rob	NULL
John	NULL
Adam	NULL

All the records from the task table will be returned even the non-matching one

LEFT EXCLUDING JOIN

SELECT assignee.assignee_name assigneeName,

task.task_title taskTitle

FROM assignee LEFT JOIN task

ON assignee.assignee_id = task.assignee_id where task.task_id IS NULL

Showing rows 0 - 3 (4 total, Query took 0.0034 seconds.)

```
SELECT assignee.assignee_name assigneeName, task.task_title taskTitle FROM assignee LEFT JOIN task ON assignee.assignee_id = task.assignee_id where task.task_id IS NULL
```

Profilin

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

assigneeName	taskTitle
james	NULL
rob	NULL
John	NULL
Adam	NULL

All the records in assignee table that do not match any records in task table are returned

RIGHT EXCLUDNG JOIN

SELECT assignee.assignee_name assigneeName,
task.task_title taskTitle
FROM assignee
RIGHT JOIN task
ON task.assignee_id = assignee.assignee_id
where assignee.assignee_id IS NULL

Showing rows 0 - 3 (4 total, Query took 0.0035 seconds.)

```
SELECT assignee.assignee_name assigneeName, task.task_title taskTitle FROM assignee RIGHT JOIN task ON task.assignee_id = assignee.assignee_id where assignee.assignee_id IS NULL
```

Profiling [Edit inli

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

assigneeName	taskTitle
NULL	Project 1 task 1
NULL	Project 2 task 2
NULL	Project 1 sub-task 1
NULL	Project 2 sub-task 2

All the records in Task table that do not match any records in assignee table are returned

FULL OUTER JOIN DOES NOT EXIST IN SQL

```
SELECT assignee.assignee_name assigneeName,
task.task_title taskTitle
FROM assignee
LEFT JOIN task
ON assignee.assignee_id = task.assignee_id
UNION
SELECT assignee.assignee_name assigneeName,
task.task_title taskTitle
FROM assignee RIGHT
JOIN task
ON assignee.assignee_id = task.assignee_id
```

```
SELECT assignee.assignee_name assigneeName, task.task_title taskTitle FROM assignee LEFT JOIN task ON assignee.assignee_id = task.assignee_id UNION
RIGHT JOIN task ON assignee.assignee_id = task.assignee_id
```

Show all | Number of rows: 25 Filter rows: Search this table

+ Options

assigneeName	taskTitle
tony	Project 1 task 1
james	NULL
roberto	Project 1 task 2
rob	NULL
John	NULL
Adam	NULL
NULL	Project 1 task 1
NULL	Project 2 task 2
NULL	Project 1 sub-task 1
NULL	Project 2 sub-task 2

We use union with LEFT JOIN AND RIGHT JOIN all the records that do not match in assignee and task tables.

Query all the USERS for a specific course by id or course name:

```
SELECT users.*
FROM users
INNER JOIN users_course
ON users.user_id = users_course.user_id
INNER JOIN course
ON users_course.course_id = course.course_id
 INNER JOIN module_course
ON module_course.course_id = module_course.module_id
INNER JOIN module
 ON module.module_id = module_course.module_id
WHERE course.course_id = 1
```

Query all the MODULES for a specific course

```
SELECT module.* from module
INNER JOIN module_course
ON module.module_id = module_course.module_id
INNER JOIN course
ON module_course.course_id = course.course_id
WHERE course.course_id = 1
```

Query to view all the modules

```
SELECT * FROM module;
```

Query to view all the courses

```
SELECT * FROM course;
```

Query to view all the users

```
SELECT * FROM users;
```

Query all the GOALS for a specific project

```
SELECT goal.* FROM goal
INNER JOIN project
on goal.project_id = project.project_id
WHERE project.project_id = 1
```

Query all the OBJECTIVES for a specific project by id or by name:

```
SELECT objective.* FROM objective
INNER JOIN project
ON objective.objective_id = project.project_id
WHERE project.project_id = 1
```

Query all the WORKLOW STEP for a specific project:

```
SELECT workflow_step.* FROM workflow_step
INNER JOIN project
ON workflow_step.project_id = project.project_id
WHERE project.project_id = 1
```

Query all the COMMENTS for a specific project by id or by name:

```
SELECT * FROM project_comment
INNER JOIN project
ON project_comment.project_id = project.project_id
WHERE project.project_id = 1
```

Query the TEAM for a specific project.

```
SELECT team.* FROM team
INNER JOIN project
ON project.team_id = team.team_id
WHERE project.project_id = 1;
```

Query the all the USERS for a specific project by id or by name

```
SELECT * FROM users
 inner join users_team
on users.user_id = users_team.user_id
inner join team
on team.team_id = users.user_id
inner join project
on project.project_id = team.team_id
where project.project_id = 1
```

Query all the TASKS and SUBTASKS for a specific project

```
SELECT task.* FROM task
INNER JOIN project
ON task.project_id = project.project_id
WHERE project.project_id = 1
```

Query to display all the users and their course and modules:

```
SELECT
users.user_first_name first_name,
users.user_last_name last_name,
users.user_password password,
users.user_email email,
users.user_phone_number phone_numer,
course.course_name course ,
users.user_departement departement,
users.user_about_me about_me
FROM users
inner join users_course
on users.user_id = users_course.user_id
inner join course
on users_course.course_id = course.course_id
inner join module_course
on course.course_id = module_course.course_id
inner join module
on module_course.module_id = module.module_id
group by users.user_id
```

22 APPENDIX M - Slim Setup

https://getcomposer.org/download/

Download composer first

Install COMPOSER in HTDOCS

```
php -r "copy('https://getcomposer.org/installer', 'composer-setup.php');"
```

```
php -r "if (hash_file('SHA384', 'composer-setup.php') === '544e09ee996cdf60ece3804abc52599c22b1f40f4323403c44d44fdfdd586475ca9813a858088ffbc1f233e9b180f061') { echo 'Installer verified'; } else { echo 'Installer corrupt'; unlink('composer-setup.php'); } echo PHP_EOL;"
```

```
php composer-setup.php
```

```
php -r "unlink('composer-setup.php');"
```

Composer.phar inside project folder

Install SLIM with composer

```
Php composer.phar require slim/slim "^3.8.1"
```

Apache configuration

Create a folder called public

Inside public folder create a .htaccess and copy the following

```
RewriteEngine On
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^ index.php [QSA,L]
```

23 APPENDIX N – Main reasons not to use MySQL

- It is not under active development.
- It is officially deprecated as of PHP 5.5 (released June 2013).
- It has been removed entirely as of PHP 7.0 (released December 2015).
- It would not be supported in any version of PHP starting of the 31 Dec 2018.
- Lacks an object-oriented interface.

MySQL query do not support:

- Prepared statements or parameterized queries to avoid SQL injection.
- Non-blocking, asynchronous queries.
- Stored procedures.
- Multiple Statements.
- Transactions.
- The "new" password authentication method (by default in MySQL 5.6; required in 5.7).
- The functionalities in MySQL 5.1.

24 APPENDIX O - Test cases functional testing

Requirements: Add organisation

ID Test	FR ID	SCENARIO	EXPECTED RESULT	RESULT
14	1	Users enter name with unexpected characters (e.g. numbers)	The field “name” should be validated	PASS
15	1	Users enter address with less than 10 characters	The field “address” should not be validated	PASS
16	1	Users enter type with unexpected characters (e.g. numbers)	The field “type” should not be validated	PASS
17	1	Users enter phone number with unexpected characters (e.g. letters)	The field “phone” number should not be validated	PASS
18	1	Users enter an email address without a ‘@’ character or a dot.	The field “email” should not be validated	PASS
19	1	Users enter country with unexpected characters (e.g. Numbers)	The field “country” should not be validated	PASS
20	1	Users did not complete the form	The button “submit” should not be available	PASS
21	1	Users submit the form	The users should get sensible feedback when completed	PASS
22	1	Users submit the form	The form should be cleared to add another organisation	PASS
23	1	Users click add organisation	The users should be able to view the form add organisation	PASS

Requirements: View organisation

ID Test	FR ID	SCENARIO	EXPECTED RESULT	RESULT
24	3	Users click “view organisation” link	The users should be able to view all the organisations of the system	PASS

Requirements: Edit user

ID Test	FR ID	SCENARIO	EXPECTED RESULT	RESULT
25	7	Users click “edit users” link	The users should be able to view all the users of the system	PASS
26	7	Users click edit users button	All the fields should become editable	PASS
27	7	Users click cancel	All the fields should not be editable and the changes should be reset	PASS
28	7	Users click save	The changes should be saved	PASS
29	7	Users edit first name with unexpected characters (e.g. numbers)	The field “first name” should not be validated	PASS
30	7	Users edit last name with unexpected characters (e.g. numbers)	The field “last name” should not be validated	PASS
31	7	Users enter an email address without a ‘@’ character or a dot.	The field “email” should not be validated	PASS
32	7	Users enter phone number with unexpected characters (e.g. letters)	The field “phone” number should not be validated	PASS
33	7	Users enter Department with unexpected characters (e.g. numbers)	The field “department” should not be validated	PASS
34	7	Users click save	The users should get a sensible feedback when it has been edited	PASS

Requirements: View user

ID Test	FR ID	SCENARIO	EXPECTED RESULT	RESULT
35	3	Users click “view all users” link	The users should be able to view all the users of the system	PASS
36	3	Users click “view users per course” link	The users should be able to select the course	PASS
37	3	Users select a course	The users should be able to view all the users associated to the course	PASS
38	3	Users select a different course	The users should be able to view all the users associated to the course	PASS
39	3	Users select a course with no users	The users should get a feedback “No users for this course”	PASS

Requirements: Add modules

ID Test	FR ID	SCENARIO	EXPECTED RESULT	RESULT
40	52	Users click “add module” link	The users should be able to view the form “add module”	PASS
41	52	The users select a course with already 4 modules assign to it	The button “add a new module” should not be available	PASS
42	52	The users select a course with already 4 modules assign to it.	The button “submit” should not be available	PASS
43	52	The users select a course with less than 4 modules assign to it.	The button “add module” should be available	PASS
44	52	The users click “add module” button	The form “add module” should be available	PASS
45	52	The users click “add module” button more than the number of modules allowed for this course	The button “add module” should not be available	PASS
46	52	The users click “delete module “button	The form “should be updated”	PASS
47	52	The users enter a “module name” with unexpected characters (e.g. numbers)	The field “module name” should not be validated	PASS
48	52	The users enter a “module description” with unexpected characters (e.g. numbers)	The field “module description” should not be validated	PASS
49	52	The users enter a “module year” in the past	The field “module year” should not be validated	PASS
50	52	The users do not complete the form	The “submit” button should not be available	PASS
51	52	The users click “submit” button	The users should get a sensible feedback	PASS
52	52	The users select another course while not completing the current form	The form should be reset	PASS

Requirements: Add course

ID Test	FR ID	SCENARIO	EXPECTED RESULT	RESULT
53	55	The users click “add course”	The users should be able to view the form “add course”	PASS
54	55	The users enter a “course name” with unexpected characters (e.g. numbers)	The field “course name” should not be validated	PASS
55	55	The users enter a “course description” with unexpected characters (e.g. numbers)	The field “course description” should not be validated	PASS
56	55	The users do not complete the form	The button “submit” should not be available	PASS
57	55	The users click “submit” button	The users should get sensible feedback	PASS

Requirements: Multiple requirements are related to these test cases

ID Test	FR ID	SCENARIO	EXPECTED RESULT	RESULT
58		The users click “hamburger menu” to expand the navigation sidebar	The navigation sidebar is expanded	PASS
59		The users click “Hamburger menu” to retract the navigation sidebar	The navigation sidebar is retracted	PASS
60		The users click “Users” in the navigation sidebar	The submenus for “users” are expanded	PASS
61		The users click “Courses” in the navigation sidebar	The submenus for “courses” are expanded	PASS
62		The users click “Modules” in the navigation sidebar	The submenus for “modules” are expanded	PASS
63		The users click “Organisations” in the navigation sidebar	The submenus for “organisations” are expanded	PASS

25 APPENDIX P - Test cases non-functional testing

ID	NFR AREA	SCENARIO	EXPECTED RESULT	RESULT
1	Compatibility	Login to the system from Chrome	Interface should automatically resize to the correct screen	PASS
2	Compatibility	Login to the system from Opera	Interface should automatically resize to the correct screen	PASS
3	Compatibility	Login to the system from Safari	Interface should automatically resize to the correct screen	PASS
4	Compatibility	Login to the system from Mozilla	Interface should automatically resize to the correct screen	PASS
5	Compatibility	Login to the system from Chrome	The users can access the system	PASS
6	Compatibility	Login to the system from Opera	The users can access the system	PASS
7	Compatibility	Login to the system from Safari	The users can access the system	PASS
8	Compatibility	Login to the system from Mozilla	The users can access the system	PASS
9	Design	View navigation bar across all the pages for students	The navigation bar should be identical across all the pages for lecturers	PASS
10	Design	View navigation bar across all the pages for students	The navigation bar should be identical across all the pages for students	PASS
11	Design	View navigation sidebar across all the pages for students	The navigation sidebar bar should be identical across all the pages for lecturers	PASS
12	Design	View navigation sidebar across all the pages for students	The navigation sidebar bar should be identical across all the pages for students	PASS
13	Design	The navigation sidebar is expanded	The navigation sidebar should be responsive	PASS
14	Design	The navigation sidebar is retracted	The navigation sidebar should be responsive	PASS
15	Performance	Inserting data into the database	The system should retrieve the data within a second	PASS
16	Performance	Querying the data from the database	The system should retrieve the data within a second	PASS
17	Performance	Inserting data into the database with multiple consecutive insert statement	The system should retrieve the data within a second	PASS
18	Performance	Querying data from the database with multiple consecutive complex queries	The system should retrieve the data within a second	PASS
19	Performance	Call REST API to retrieve all the users	The system should return a JSON object within a second	PASS
20	Performance	Call REST API to retrieve all the courses	The system should return a JSON object within a second	PASS
21	Performance	Call REST API to retrieve all the modules	The system should return a JSON object within a second	PASS
22	Performance	Call REST API to retrieve all the organisations	The system should return a JSON object within a second	PASS
23	security	Attempt to login with the wrong password	The system should not redirect the users to the home page	PASS
24	security	Attempt to login with the wrong username	The system should not redirect the users to the home page	PASS
25	security	Attempt to access a page without logging in	The system should redirect the users to the login page	PASS

26 APPENDIX Q - REST API testing

ID	REQUEST TYPE	REQUEST	EXPECTED RESULT	RESULT
1	GET	http://slim.kingstonse.org/view/course	Return JSON object of all the courses	PASS
2	GET	http://slim.kingstonse.org/home/login	Return JSON object of the right user	PASS
3	GET	http://slim.kingstonse.org/view/module	Return JSON object of all the modules	PASS
4	GET	http://slim.kingstonse.org/return/specific/{{id}}	Return JSON object of all the modules matching the id course	PASS
5	GET	http://slim.kingstonse.org/view/organisation	Return JSON object of all the organisations	PASS
6	POST	http://slim.kingstonse.org/add/organisation	Add an organisation into the system	PASS
7	GET	http://slim.kingstonse.org/view/user	Return JSON object of all the users	PASS
8	POST	http://slim.kingstonse.org/add/user	Add a user into the system	PASS
9	GET	http://slim.kingstonse.org/view/user/{id}	Return JSON object of all the users matching the is	PASS
10	PUT	http://slim.kingstonse.org/edit/user	Update user details	PASS
11	GET	http://slim.kingstonse.org/view/usertype	Return JSON object of all the user type	PASS