

SOFTWARE DEVELOPMENT LIFE CYCLE MANAGEMENT TOOL FOR HUMAN RE-
SOURCES

by

ANASTASIOS KOUMARELIS
(MTP246)

Bachelor's degree, Hellenic Mediterranean University, 2020

A THESIS

submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

SCHOOL OF ENGINEERING

HELLENIC MEDITERRANEAN UNIVERSITY

2022

Approved by:

Major Professor
Dr. Nikolaos Vidakis

ΑΝΑΠΤΥΞΗ ΕΡΓΑΛΕΙΟΥ ΔΙΑΧΕΙΡΙΣΗΣ ΚΥΚΛΟΥ ΖΩΗΣ ΛΟΓΙΣΜΙΚΟΥ ΒΑΣΗ
ΑΝΘΡΩΠΙΝΟΥ ΔΥΝΑΜΙΚΟΥ

από

ΑΝΑΣΤΑΣΙΟΣ ΚΟΥΜΑΡΕΛΗΣ
(ΜΤΡ246)

Πτυχίο, Ελληνικό Μεσογειακό Πανεπιστήμιο, 2020

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

υποβάλλεται σε μερική εκπλήρωση των απαιτήσεων για την απόκτηση του πτυχίου

ΜΕΤΑΠΤΥΧΙΑΚΟ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΕΛΛΗΝΙΚΟ ΜΕΣΟΓΕΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ

2022

Εγκρίθηκε από:

Επιβλέπων καθηγητής
Δρ. Νικόλαος Βιδάκης

Acknowledgments

I would like to express my gratitude to my primary supervisor, Dr. Nikolao Vidaki, who guided me and supported me throughout this master's thesis. Also, I would like to thank my colleagues for the useful advices and my family who have always been by my side to support me throughout the years of my studies.

Abstract

Technology nowadays is a very important tool in many fields of our daily life. It is a major factor in the development of many objects that significantly improves the way and quality of life of many people. The rise of technology has been largely achieved using computers and more specifically using the right software. Many fields of technology evolved significantly with the development of new and more complex software capable of meeting the demands that arose. The need to develop complex software within a short time made it imperative to organize and design how the software was implemented to reduce the risk of failure.

Proper software development helps developers make huge savings in time and cost. With proper design and analysis of the requirements and functions that the software should have, the risk of failure and malfunctioning is greatly reduced. The Software Development Life Cycle (SDLC) methodologies define specific steps and actions for correct and efficient software development. In this way, the team of developers follows a specific way of implementation to reduce the risk of failure. Depending on the size and complexity of the software, a different SDLC model is required. The risk of software implementation failure can be further reduced by proper staff management. It is very important that the team leader can have an overview of the project and the programmers assigned to implement it.

The purpose of this master's thesis is to present an application for a team leader to manage and assign projects to his staff based on SDLC models. Initially, the team manager will be able to register all team members-programmers and then can assign projects based on any SDLC model wants (Waterfall, Iterative, Spiral, V-Shaped). He will be able to manage the phases of each model and have an overall estimate of the cost and delivery time of the project. The team members will also be able to access the application and be able to informed about their financial information to the company, as well as the projects assigned to them in detail with the deadline of each phase.

Σύνοψη

Η τεχνολογία στις μέρες μας αποτελεί ένα πολύ σημαντικό εργαλείο σε πάρα πολλούς τομείς στην καθημερινότητα μας. Αποτελεί κύριο παράγοντα εξέλιξης σε πολλά αντικείμενα που βελτιώνουν σημαντικά τον τρόπο και την ποιότητα ζωής πολλών ανθρώπων. Η άνοδος της τεχνολογίας επιτεύχθηκε σε μεγάλο βαθμό με την χρήση υπολογιστών και πιο συγκεκριμένα με την χρήση του σωστού λογισμικού. Πολλοί τομείς της τεχνολογίας εξελίχθηκαν σημαντικά με την ανάπτυξη καινούργιου και πιο πολύπλοκου λογισμικού, ικανού να ανταποκριθεί στις απαιτήσεις που εμφανιζόταν. Η ανάγκη ανάπτυξης πολύπλοκου λογισμικού σε καθορισμένα χρονικά όρια έκανε επιτακτική την ανάγκη οργάνωσης και σχεδίασης του τρόπου υλοποίηση του λογισμικού προκειμένου να μειωθεί ο κίνδυνος αποτυχίας.

Η σωστή ανάπτυξη λογισμικού βοηθάει τους προγραμματιστές να κάνουν τεράστια εξοικονόμηση χρόνου και κόστους. Με την σωστή σχεδίαση και ανάλυση των απαιτήσεων και λειτουργιών που θα πρέπει να έχει το λογισμικό, μειώνεται κατά πολύ ο κίνδυνος αποτυχίας και δυσλειτουργίας του. Βασικό εργαλείο για την σωστή και αποτελεσματική ανάπτυξη λογισμικού αποτελούν οι Software Development life cycle (SDLC) μεθοδολογίες που καθορίζουν συγκεκριμένα βήματα και ενέργειες που πρέπει να γίνουν σε σαφή χρονικά πλαίσια. Με αυτόν τον τρόπο η ομάδα των προγραμματιστών ακολουθεί έναν συγκεκριμένο τρόπο υλοποίησης για να μειωθεί ο κίνδυνος αποτυχίας. Ανάλογα το μέγεθος και την πολυπλοκότητα του λογισμικού απαιτείται και διαφορετικό SDLC μοντέλο. Ο κίνδυνος αποτυχίας υλοποίησης ενός λογισμικού μπορεί να μειωθεί ακόμα περισσότερο με την σωστή διαχείριση προσωπικού. Είναι πολύ σημαντικό ο υπεύθυνος της ομάδας να μπορεί να έχει μία συνολική εικόνα του έργου και των προγραμματιστών που τους έχει αναθέσει την υλοποίηση.

Σκοπός της συγκεκριμένης εργασίας είναι να παρουσιάσει μία εφαρμογή, ώστε να μπορεί ένας υπεύθυνος ομάδας να διαχειριστεί και να αναθέσει έργα στο προσωπικό του, βάση SDLC μοντέλων. Αρχικά θα μπορεί ο διαχειριστής-υπεύθυνος της ομάδας να κάνει εγγραφή για όλα τα μέλη-προγραμματιστές και στην συνέχεια θα μπορεί να τους αναθέτει έργα βάση όποιου SDLC μοντέλου επιθυμεί (Waterfall, Iterative, Spiral, V-Shaped). Θα μπορεί να διαχειρίζεται τις φάσεις κάθε μοντέλου και να έχει μία συνολική εκτίμηση του κόστους και χρόνου παράδοσης του έργου. Τα μέλη της ομάδας θα μπορούν να έχουν και αυτά πρόσβαση στην εφαρμογή και να ενημερώνονται για τα οικονομικά τους στοιχεία στην εταιρία, αλλά και για τα έργα που τους έχουν ανατεθεί αναλυτικά με τους χρόνους παράδοσης κάθε φάσης.

Table of Content

Acknowledgments	2
Abstract	3
1 Introduction	9
2 Background	11
2.1 Software Development Life Cycle (SDLC)	12
2.2 SDLC Models	14
2.2.1 Waterfall	15
2.2.2 Iterative	18
2.2.3 Spiral	20
2.2.4 V-Shaped	22
2.3 Problem Statement & Existing Systems.....	25
3 System Analysis & Design	26
3.1 System Textual Description Analysis	26
3.2 System Analysis	27
3.2.1 Requirements List and Main Requirement Diagram.....	28
3.2.2 Frontend requirement diagram	30
3.2.3 Backend requirement diagram	32
3.2.4 Database requirement diagram	33
3.3 System Design	33
3.3.1 Component Diagram	34
3.3.2 Use case	35
3.3.3 Activity Diagram	38
3.3.4 Sequence Diagrams	40
3.3.5 Entity Relationship Diagram (ERD)	42
3.4 System Technologies	43
3.4.1 React.js	44
3.4.2 Spring boot	46
3.4.3 MySQL	47

4	System Representative Use Case Scenarios	48
4.1	Representative Use Case Scenarios.....	48
4.1.1	Admin Role.....	49
4.1.2	Simple User Role	65
5	Conclusion & Future Work	69
5.1	Conclusion	69
5.2	Future work	70
6.	References.....	72

List of Figures

Figure 1 Software Development Life Cycle (SDLC)	13
Figure 2 Waterfall SDLC.....	16
Figure 3 Iterative SDLC.....	19
Figure 4 Spiral SDLC	21
Figure 5 V-Shaped SDLC.....	23
Figure 6 Requirement List	30
Figure 7 Main Requirement Diagram.....	31
Figure 8 Front-end Requirement Diagram.....	32
Figure 9 Backend Requirement Diagram	33
Figure 10 Repository Requirement Diagram.....	34
Figure 11 Main Component Diagram.....	36
Figure 12 Main Use Case Diagram.....	37
Figure 13 Use Case Scenario	38
Figure 14 Simple User Activity Diagram	39
Figure 15 Admin Activity Diagram.....	40
Figure 16 Admin First Sequence Diagram	41
Figure 17 Admin Second Sequence Diagram.....	42
Figure 18 Simple User Sequence Diagram.....	42
Figure 19 ERD	43
Figure 20 React Js Example.....	45
Figure 21 Material UI Example	46
Figure 22 Spring Boot Class Example.....	47
Figure 23 Spring Boot Controller Example	48
Figure 24 System Structure.....	49
Figure 25 Register Page	50
Figure 26 Register Wrong Password Page.....	51
Figure 27 Register Wrong Email Page	51
Figure 28 Login Page.....	52
Figure 29 Admin Dashboard Page.....	53
Figure 30 Admin Update Info Page	53

Figure 31 Admin Menu Page	54
Figure 32 Admin Employees Page	55
Figure 33 Admin Add Employees Page.....	56
Figure 34 Admin Delete Employees Page	57
Figure 35 Admin Financial Page	57
Figure 36 Admin Edit Employee Financials Page	58
Figure 37 Admin Add Simple Task Page	59
Figure 38 Admin Add Waterfall Task Page	60
Figure 39 Admin Iterative Task Page	61
Figure 40 Admin Add Spiral Page.....	62
Figure 41 Admin Add V-Shaped Task Page.....	63
Figure 42 Admin Calendar Page.....	64
Figure 43 Admin Calendar with Tasks Page	64
Figure 44 Admin Edit Calendar Task Page	65
Figure 45 Admin Settings Page	66
Figure 46 Simple User Dashboard Page	66
Figure 47 Simple User Financial Page.....	67
Figure 48 Simple User Calendar Page	68
Figure 49 Simple User Settings Page.....	69

1 Introduction

In recent years there has been a huge growth in technological development in many fields that directly affect everyday life in the modern world. The impact of this growth increases daily the demand for creating large-scale reliable software systems that serve the needs that emerge [1]. The development of such systems is in many cases a difficult process for the teams that undertake it, as it requires proper management of human resources, budget, and time for the development of each subsystem. Improper management of the above can lead to delayed delivery time or even the inability to complete the system [2][3].

As a result, the Software Development life cycle (SDLC) was created, which is a methodology for the creation and modification of a reliable software system. There are several SDLC models used depending on the software system to be developed [4][5][6]. The choice of each SDLC model depends heavily on the delivery time, the available human resources and by budget for the project assigned. The most popular and used SDLC models are the Agile model, Waterfall model, Iterative model, Spiral model, and V-shaped model [7][8].

There are project management platforms that allow companies to manage their human resources according to the models mentioned above [9][10][11]. Most systems, however, need several configurations to support management based on these SDLC models. The problem occurred when a project manager needs open-source software to be able to manage staff based on SDLC models that would be all in one application, without the need for configuration or changes to support the SDLC models. Still, be able to access it from all devices and give him the ability to have a complete view of his staff regarding financials and assignment of tasks.

In this thesis will be created an open-source system that will allow the user to manage the human resources and schedule to develop software systems with whichever SDLC model the user chooses (Waterfall, Iterative, Spiral, V-Shaped). Initially, it will be able to give data to the system such as names and salaries of employees through fields. The user will then be able to assign projects depending on the available human resources and the chosen SDLC model (each SDLC model has a standard management framework, based on the definition of each model separately). Eventually, the user has a complete perspective of the total delivery time of the project and the cost spent for its development.

In conclusion, the purpose of this thesis is to offer the optimum ability to manage and oversee the software development process to the user. In addition, the user can modify the system as an extension to an HRMS (Human Resource Management System) allowing additional SDLC models to be added to existing ones. Finally, with the API that the system will have internally, other systems will be able to connect to pull any data the user needs.

2 Background

The rise of technology has made it possible for computers to be used in critical fields of society and for everyone to have easy access to meet their daily needs. With the use of the right software, the computer is a very important tool for direct management and processing of data as well as for problem-solving in various fields such as:

- Industry
- Banks and financial
- Business
- Communication
- Education
- Medical
- Transportation

Depending on the field where the computer is used, the needs that arise and the way of solving problems differ. Some sectors need software to process a small amount of data, but other fields need quite complex software that can process a huge amount of data [12][13].

According to the above, the need to develop systems capable of meeting the needs of the users and to be able to develop them in a pre-defined schedule was created. The creation of these systems is in many cases a complex and time-consuming process for the teams of people who undertake it. Proper development of software systems requires proper management of manpower, resources, and time for the development of each subsystem until the completion of the whole system and the changes that may occur during its creation. In case of mismanagement of the above then the teams of people who have undertaken it are faced with delayed delivery time, wrong implementation, or even failure to deliver the system [2][3]. To avoid such phenomena and to develop software systems properly, Software Development life cycle (SDLC) models were developed where they are predefined methodologies by which a reliable high-quality software system is created and modified with a low risk of failure.

2.1 Software Development Life Cycle (SDLC)

Software Development Life Cycle (SDLC) is a process that is used for software development and includes all the required steps that need to be taken to develop and deliver a software system with the lowest possible risk of failure and the highest possible maintenance of quality [16]. It is a detailed illustration of the set of activities to be followed by all parties involved [14]. The parties involved are usually a programming company and the customer who is in frequent communication throughout the development of a software system due to changes or difficulties that arise. Some of the activities included in a software lifecycle are necessary and others are optional to enable direct interaction between developers and customers and the flexibility to deal with difficulties or errors that may arise throughout the lifecycle [17]. A simple life cycle that software has consists of five steps and includes software design to software maintenance. Each step includes sub-steps and procedures to be followed [14]. Figure 1 depicts all the steps in detail, and they are discussed in detail below.

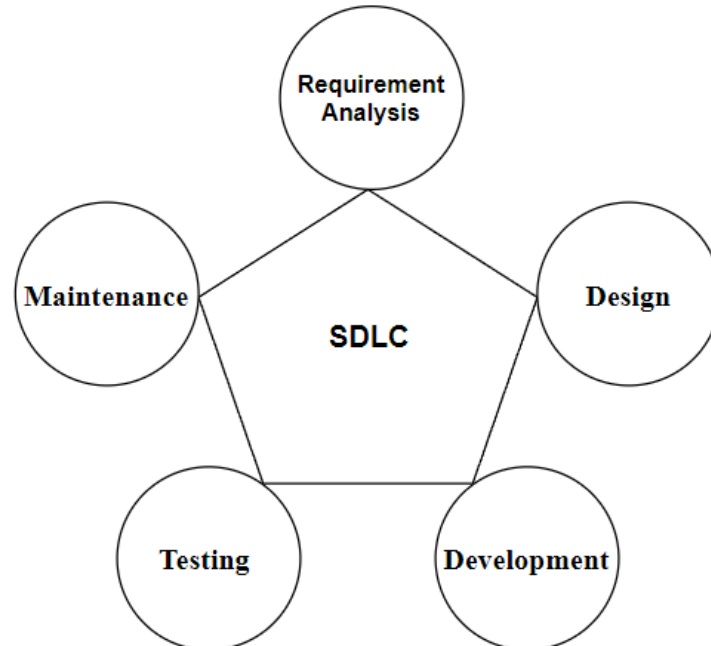


Figure 1 Software Development Life Cycle (SDLC)

1) Requirement Analysis

It is the first and most important step for proper software development. In this process, all the requirements of the product are analyzed and a forecast of the resources that will be needed and the management that must be done. During this process, the client is in constant communication with all the stakeholders who are developers, sales, external and internal experts, and others to make a complete analysis of the requirements they have about the product and the full schedule that the software development should have [16][17]. This way the cost evaluation is done and enables the developers to have a complete picture of the product they must develop. After this process, a Software Requirements Specification (SRS) document is produced[18].

The SRS document describes all the features that the product will have, the business processes that will be supported, and contains the functional requirements that have been set. It is a very important document since they rely on it to implement the next steps of the software lifecycle. It still prevents design errors such as conflicting requirements that may occur and need reassessment by all parties involved in the project [18].

2) Design

In the design phase, product architects or engineers based on the SRS document create design approaches for the architecture of the product or system. They convert the requirements into specifications and usually create different versions of Design Document Specifications (DDS). The DDS document defines all the architectural modules of the product and contains several design diagrams and models including class diagrams, data flows, Unified Modeling Language (UML) diagrams, and use case diagrams. The document is then checked against criteria such as correct design and time calculation to select the most correct DDS document in terms of its design approach [15][16][17].

3) Development

After the requirement analysis and design, the development of the software product is started by the developers who have undertaken the project. The software is developed following the DDS document discussed above and the guidelines set out individually by each organization. The developers have at their disposal many programming and data management tools that help them to develop software correctly and quickly without much difficulty if these tools are used properly.

Various programming languages are used depending on the product to be created and usually, a documentation of the code is written for the understanding of other developers who may need to maintain or extend this code [15][16] [17].

4) Testing

It is a very important process for the software as it is just before the product becomes available to users. The developers test the code to make sure that all functions and generally all software created works properly without bugs. This process usually identifies many points that need improvement or correction until the product reaches the specifications set in the SRS document discussed above. This process must be done correctly because once the software is made available to users it can create a lot of discomfort in the event of a defective operation [15][16] [17].

5) Maintenance

In the last stage of the software life cycle, the creation and testing are complete and have been given to the users. They, in turn, identify different bugs in the system over time, and the developers assigned to the project are required to solve the bugs that occurred [15][16][17].

2.2 SDLC Models

Over time with the use of classical software development life cycles, several models defining specific steps for software development have been created and are referred to as SDLC Models or Software Development Process Models (SDPM). Each model is different from the others and contains different steps or ways to develop software. Every SDLC model has different advantages and disadvantages and is used in different circumstances depending usually on the size and cost of each product [16]. In this report, the following four models will be analyzed and implemented in the project that will be analyzed below. The four models are:

- 1) Waterfall
- 2) Iterative
- 3) Spiral
- 4) Vmodel

2.2.1 Waterfall

The Waterfall model was introduced by Winston W. Royce in 1970 and is the first of the models created. It is a sequential model and is the basis for the creation of later models [28]. It is very simple to understand and apply and was the most popular model in earlier years. It is divided into six phases and each phase must end before the next phase can begin [20][27]. The original Waterfall model allows us to go back to previous phases at the end. Figure 2 shows the six phases of the Waterfall model.

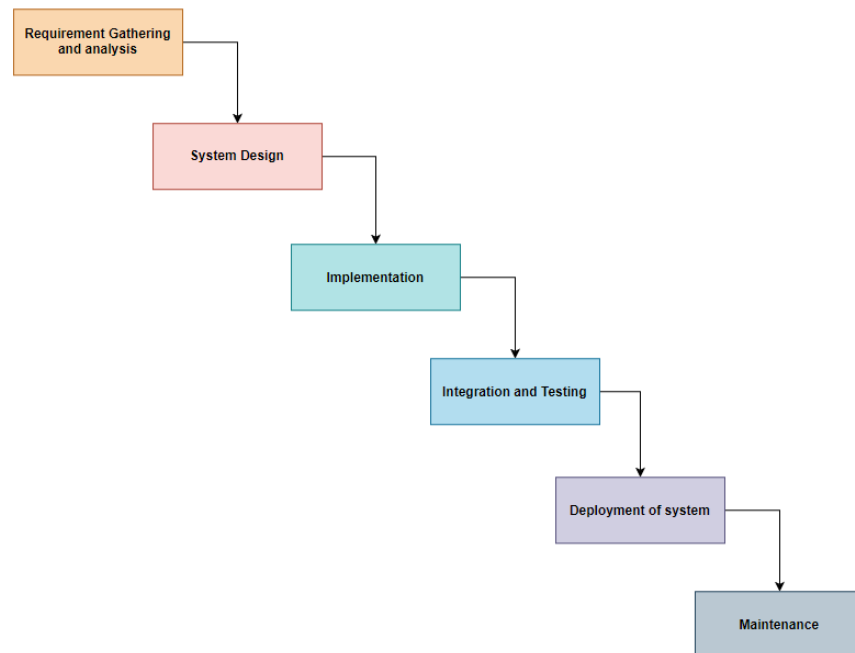


Figure 2 Waterfall SDLC

1) Requirement Gathering and analysis

In the first phase of the Waterfall model, a detailed record of the customer's requirements must be made and all the functions and capabilities that the customer wants the system to have must be analyzed. The requirements should be fully understood and recorded without any point that is not properly specified. The developers should know how the final product must be, by analyzing all the requirements of the customer. Then once the analysis has been done all the requirements and functions are recorded in an SRS document as discussed in the explanation above [12][31][33].

2) System Design

Based on the SRS created in the previous phase, the requirements are extracted and converted into a form that can be used to start code implementation. To do this a Software Design Document (SDD) is produced which contains a detailed analysis of the architecture of the system. It contains schedules, functional descriptions, user interface, solutions to problems that may arise, and objectives that in their entirety implement the customer's requirements. Once the creation of the SDD document is complete then the developers have the tools they need to start the implementation [12][31][45].

3) Implementation

The developers start the implementation of the system according to the customer's requirements and the documents mentioned above. The implementation can be done in several different programming languages or a combination of several of them. The type of language used will vary depending on the type of system and the experience of the team undertaking the development. The system is usually divided into smaller pieces called units and implemented separately in the next phase where they merged. The reason this is done is so that the developers can be much more aware of various bugs that may not be visible [12][31][33].

4) Integration and Testing

The smaller pieces of code created in the previous phase are checked for any errors or bugs they may have. Then they are integrated into a system that is subjected to many tests for proper operation. It is a very important phase for the model because errors are identified that can greatly affect the use of the system and even the experience that end users will have. If the system does not work correctly then there is an immediate risk that the software will cease to be used [12][31][33].

5) Deployment of system

After the completion of the tests on the system and the correction of any errors and bugs that occurred, the system is deployed and becomes accessible by the client and other users that may exist [31][33].

6) Maintenance

In the final phase of the Waterfall model, the system is maintained, and errors detected by users are fixed. [12][31][33].

Advantages

- 1) It is one of the simplest models and is very easy to understand.
- 2) The timelines are clearly defined from the start.
- 3) It has very good results in small systems where the requirements are well-known from the beginning.
- 4) Each phase of the model is implemented on its own each time rather than in parallel with another.
- 5) The customer can know the cost that the system development as the schedules are certain from the beginning [19][20][22].

Disadvantages

- 1) It is a difficult model when used for large systems.
- 2) The customer should fully know all the requirements they want from the system from the beginning.
- 3) It has a high risk of failure.
- 4) Not recommended for object-oriented projects and complex systems [22] [25] [27].

2.2.2 Iterative

This model was created to solve the problems or shortcomings created by the waterfall model. With the waterfall model there was the big problem that the customer's requirements had to be known from the beginning with absolute clarity. This was solved by creating the Iterative Waterfall Model. This is an iterative model that aims to complete the project by creating small pieces until the whole project is complete. In each iteration, the project goes through all the phases that a Waterfall model project goes through too, with the big difference that in the Iterative model as many iterations can be done as needed. Usually, with each new iteration, there are design updates and new functionality to make the system work better or to meet customer requirements. Figure 3 shows the Iterative Model with the lines pointing backward to represent the iteration of the model [19][21][23].

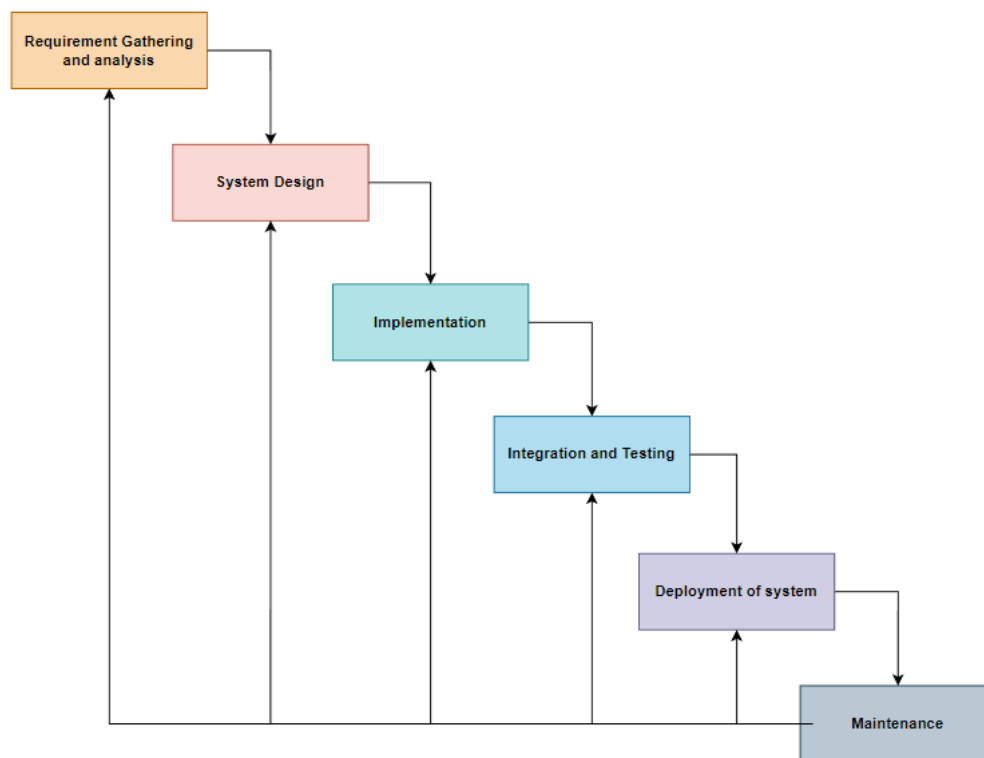


Figure 3 Iterative SDLC

Advantages

1. Not all requirements must be known from the beginning.
2. Changes can be made during the development of the system.
3. The development of the system is done with a lot of customer feedback.
4. Lower risk of failure compared to the simple Waterfall Model.
5. Ideal for large systems where requirements and functions can change.
6. It takes less time to start the implementation as the requirements may change in future iterations [21][22][25].

Disadvantages

1. Not ideal for small systems.
2. The cost can increase quite a bit compared to the simple Waterfall Model because when there is a change in requirements several points in the code must be changed or even deleted. This process requires a lot of time and therefore costs a lot of money for the development teams.
3. System integration is not known from the beginning with the risk that it takes much more time or more iterations than what was predicted at the beginning.
4. It is difficult to manage.
5. It requires a very proper design between iterations [22][31][25].

2.2.3 Spiral

The spiral model is an evolutionary software process model that combines the linear sequential model's controlled (Waterfall Model) and systematic elements with the iterative feature of prototyping. A very strong emphasis is placed on risk management which is directly related to the performance of the system. The spiral model consists of four phases that are iterated until the system is completed. Each iteration is based on risk management from the beginning, and it is not known how many iterations will be needed to complete the system. Figure 4 shows the spiral model along with its four phases [21][25][24].

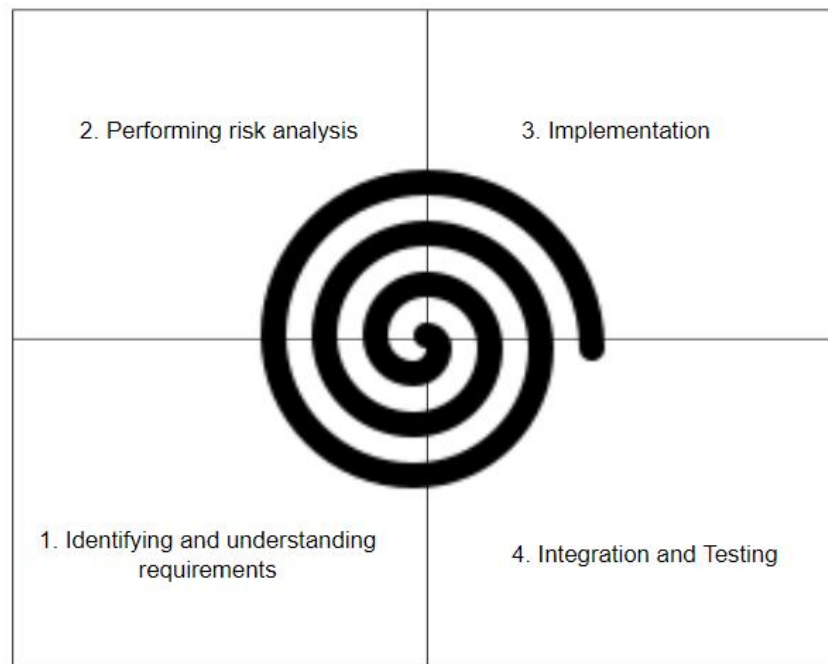


Figure 4 Spiral SDLC

Identifying and understanding requirements

In the first phase, the customer's requirements are analyzed to fully understand the development team. The team that will create the system should at this phase have a complete understanding of the functionality and requirements of the system [32][33][23].

Performing risk analysis

Once the first phase is completed a risk analysis is done and various risks for the system are predicted. An assessment is made of the weaknesses and risks identified such as cost and then the best implementation strategy with the lowest risks is chosen if the customer is satisfied [23][32][33].

Implementation

After the best version of the system has been selected then the implementation of the system is started by the team of developers. The developers should create the system based on the customer's requirements and be in constant communication with the customer [32][33].

Integration and Testing

In this phase, once the initial system or a version of the system (if it is the second iteration and beyond) is created, measurements and performance evaluation are done. Many tests are done to ensure the correct and smooth operation of the system. In the end, the customer uses the system to evaluate it and provide further requirements for future iterations [23][32][33].

Advantages

1. There is a strong emphasis on risk analysis to reduce the risk of failure.
2. Ideal for large and complex systems.
3. The customer can receive an initial version of the system very early on.
4. Allows requirements to be added to the system in future iterations [22][31].

Disadvantages

1. It is a high-cost model.
2. Risk analysis requires a lot of subject matter expertise.
3. Not recommended for small systems.
4. The number of iterations needed is unknown.
5. Complex compared to other models [22][31][33].

2.2.4 V-Shaped

The V-shaped model is one of the most used models and is an evolution of the Waterfall model. It has taken its name from its shape as shown in Figure 5 because it is divided into two groups of phases with the first group connected to the end and the second group connected to the beginning by coding. The first group of phases is about verification which focuses on the requirements that the system must have and its design. The second group of phases concerns validation which is mainly focused on testing the system and validating that all the requirements have been carried out correctly. Each phase from verification is directly linked to a phase from validation because in each implementation there will be a test that has to be done. Figure 5 shows the V-shaped model [19][23][25].

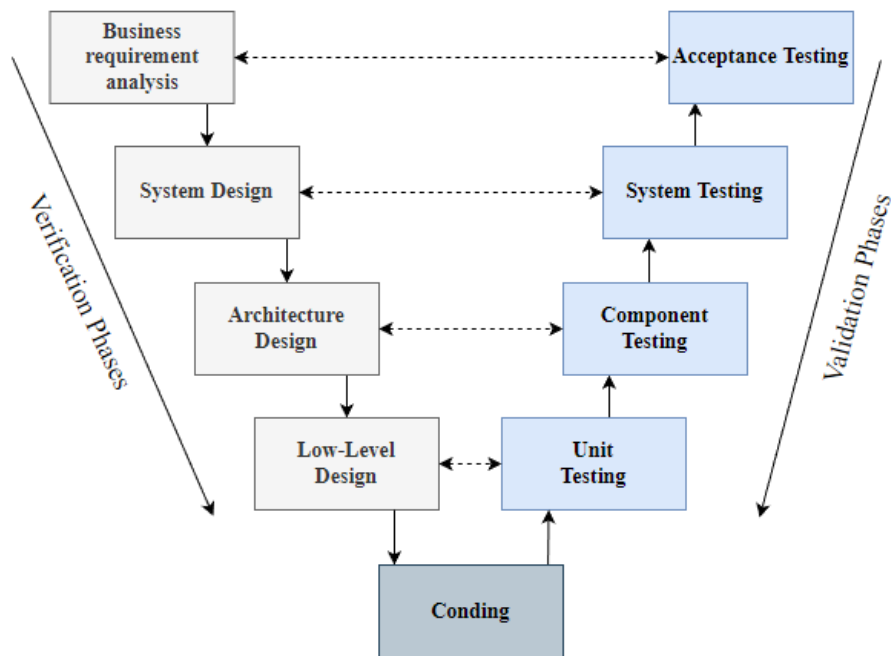


Figure 5 V-Shaped SDLC

Business requirement analysis

It is the first and most basic phase of all as the customer is given the requirements of the system and the functions it should have.

System Design

The basic design of the system is made according to the customer's requirements regarding the system.

Architecture Design

Analyze further the basic design of the system and all the technical specifications it will have such as database tables, architecture diagrams, etc.

Low-Level Design

The system is divided into smaller modules to make it easier to implement and check that everything is working properly.

Coding

Once all the previous phases are completed, the implementation of the system starts according to the customer's requirements and the analyses that have been made.

Unit Testing

Testing is started and performed on all the modules that the system has been split to identify and fix various bugs.

Component Testing

They start integrating the modules into the system and testing their communication.

System Testing

When the whole system is completed then testing is done by the client team to make sure that all the requirements of the system have been created [19][25].

Acceptance Testing

It tests the user environment in real-life conditions and identifies bugs or errors that may occur.

Advantages

1. Simple to understand and implement.
2. Ideal for small systems.
3. Has a lower failure rate than the Waterfall model.

Disadvantages

1. It is not ideal for complex and large systems.
2. Requirements must be fully known from the beginning. [19][24][26].

2.3 Problem Statement & Existing Systems

Before implementation of the system, a literature search was conducted on SDLC models aiming to fully understand them and applications that have been done in combination with human resources systems (HRMS). The main direction of the research was to find applications or studies that analyze implemented HRMS based on SDLC models, i.e., to enable a programming company or a manager to manage his staff in software development based on the SDLC models mentioned above. The research conducted revealed that many studies focused on how to develop software using SDLC models [36][37] rather than developing systems that manage staff based on these models. Most studies made comparisons between SDLC models [12][34][35] or focused on HRMS analysis without including their combination.

The problem occurred when someone needs open-source software to be able to manage their staff based on some SDLC models that would be all in one application, without needing configuration or changes to support the SDLC models. Still, be able to access it from all devices and give him the ability to have a complete view of his staff regarding financials and assignment of tasks. Many systems at commercial and research levels such as [11][38][39][40] offer the user the ability to manage their staff such as their salaries and other information. Some researchers analyze commercial tools that offer the possibility to directly manage staff based on some SDLC models or indirectly by adjusting various parameters [1][9][10].

This master thesis aims to create a system that solves the above problems and offers additional features to someone who needs it. It will have the main goal to enable the staff manager to assign tasks to his employees based on SDLC models (Waterfall, Spiral, Iterative, and V-shaped) and to be able to estimate costs based on salaries and personnel expenses. The main objective will not be to manage staff in general, such as shifts and documents, but to manage staff based on SDLC models which is the state of the art of the system. All SDLC models will be able to be used directly without any additional configuration and will be open source for future additions that one wishes to make or to adapt to own existing system.

3 System Analysis & Design

The goal of this master thesis is a system that helps someone manage the staff in his team according to the SDLC models (Waterfall, Spiral, Iterative, and V-shaped). On this innovation, the requirements that will be presented below and the diagrams that were created before the start of the implementation were based. It is very basic for a system before starting the implementation to be fully aware of the requirements and the functions it should have. First, I started the literature search, and then the requirements and functions were recorded. In the following chapters requirement list and diagram will be presented and then in System Design will analyze the use-case diagram, component diagram, sequence diagram, activity diagrams, and ERD diagram.

3.1 System Textual Description Analysis

The structure of the system consists of two basic subsystems that must operate in parallel for the smooth operation of the overall system. It is divided into the backend and the frontend. The frontend is the user interface i.e. it is what the user sees on the screen and uses. The backend is responsible for storing the data in the database, processing it, and sending it to the frontend to display it. The union of these two subsystems is done by sending data from one subsystem to the other.

Before the implementation of the system was started, in the first phase, research was done on the type of system that had to be implemented and its functions. More specifically I studied similar systems that exist in the market and in literature to define the features and functions that the system had to have and be state of the art at the same time. It turned out to be quite a time-consuming process as this system is innovative in the multiple options it gives to the user, thus requiring many different technologies for its implementation and proper recording of the requirements that emerged.

At the beginning of the implementation of the system, the database schema was designed, which defined the tables that the system needed to store the data and the relationships that had to be developed between the tables. The relationships between the tables were a very important point as in this way the data we need can be extracted and also we can directly find the data we are looking for as it is done in all relational databases.

Then the development of the backend started based on the database schema that had been designed. For the development, I used Spring Boot which is an open-source Java-based framework, and started creating the classes with the fields defined in the database schema. The classes were a visual representation of the tables in the database and the relationships between them. This was a way of verifying that our system was created according to the database schema that had been defined and that our data would be stored in the correct tables. Then the functions and controllers were created that is responsible for processing and storing the data in the database.

The data entered the system through endpoints created by HTTP requests via the Postman application and the entire system was controlled from the moment the data entered until it was processed, stored, or sent back again. The Postman application is an Application Programming Interface (API) and was used as a frontend simulator to send the data that would have been sent normally if the frontend had been developed to check that everything was working correctly on the backend.

For the development of the frontend was used JavaScript, CSS (Cascading Style Sheets), and React.js (JavaScript library for user interfaces). The system screens started to be created with the help of React and the use of Material-UI library which offers a wide variety of components that can be used with ease. The reason why these technologies were used is that they offer huge customization possibilities depending on the look the user wants to give to their system and are easy to learn and use. Once a screen was completed, tests were done on the display of the data and sending it to the backend. The frontend and backend in many cases were developed in parallel because changes and additions of new functions were needed. The two subsystems were running separately on a server each within the same computer just on different ports only during the development of the system.

3.2 System Analysis

The central idea of the system is to offer users the ability to manage their staff based on the SDLC models to have an overview of the software development progress and the cost of the project concerning the salaries of the teams that will take on the project. The team manager will be able to register and then enroll their staff in the system so that they can inform about the projects

assigned and other information such as their salary. The main feature of the system will be that everyone will be able to access it and be directed based on the instructions of the leader.

1.2.1 Requirements List and Main Requirement Diagram

Proper software development always requires that the requirements to be met by the system to be developed are fully articulated. To properly write and represent the requirements, the Systems Modeling Language (SysML) was used which was developed as a graphical dialect of the Unified Modeling Language (UML) to analyze and represent complex systems so that they can be easily understood [41]. Figure 6 shows a Requirement List that analyzes the requirements and functions that the system should have.

Requirement	Title	Description
REQ1	Frontend	The application must have a user interface user that will be easy to use.
REQ2	Backend	The system should process and store the data using the backend.
REQ3	Database	The system should have a database to store the data in relational tables so that they can be retrieved by the Backend
REQ4	Login/Register	The user will be able to log in to the system or register himself/herself or employees depending on his/her role
REQ5	Roles	The system should support two roles (Admin, User) so that each Admin can manage the Users he has registered.

REQ6	Change Password	The user will be able to change his/her password at any time.
REQ7	New Task	The admin will be able to assign tasks to any users wishes based on the SDLC models (Waterfall, Iterative, Spiral, V-shaped)
REQ8	Calendar	Users will be informed about the tasks assigned to them and their deadlines.
REQ9	Add/Delete User	Admin will be able to add and remove users whenever he needs
REQ10	Responsive	The system should be responsive to be able to work on mobile devices or screens of different sizes from the computer.
REQ11	JWT	The system will be secured by using JSON Web Tokens (JWT) between the user and the system to avoid external threats.
REQ12	UI Components	The UI should be created using components that are easy to use for the user.

Figure 6 Requirement List

The requirement diagram of the system that describes its central structure is shown in figure 7. It consists of the first three requirements that are needed for the system to work. It is a basic requirement that every system is related to. It should have a Frontend that the user will see and manipulate, a backend that will process the data, and the Database where the data will be stored.

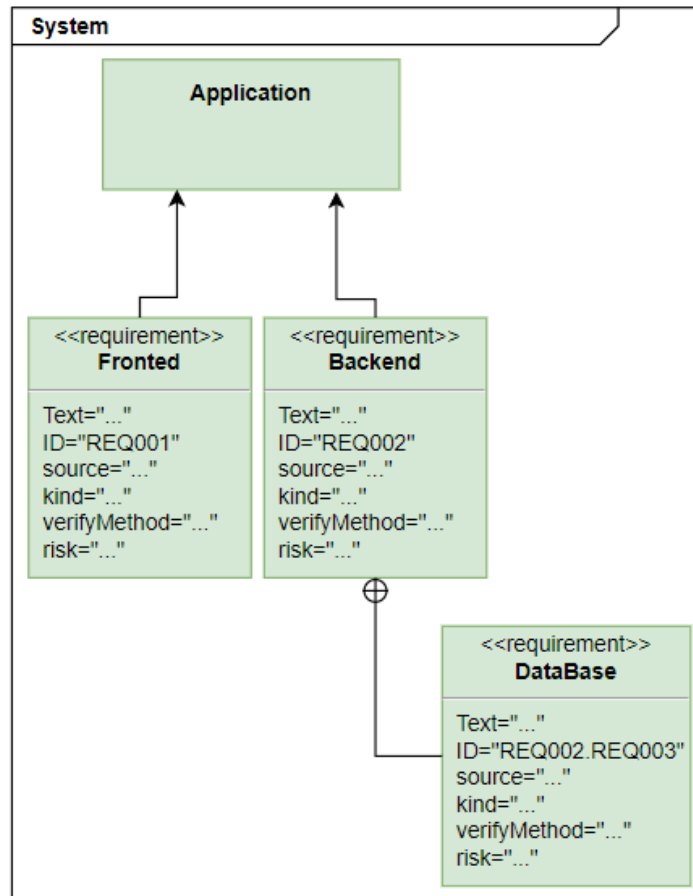


Figure 7 Main Requirement Diagram

3.2.2 Frontend requirement diagram

The Frontend is the user interface of the system, i.e. the image that the user sees on his screen and interacts with. It is a very important part of the system because based on the user's experience of the user interface and ease of use, the user's opinion of the system is determined. The

requirement diagram of the Frontend is shown in figure 8. The basic requirement (REQ017) is that the Frontend should be able to change according to the role the user will have. The screens will be different for the Admin who can assign tasks and for the User who can just be informed about the tasks assigned. All users should be able to use the system from all screen sizes whether from mobile or PCs, which is the requirement (REQ010), and the screens should be easy to use without the need for an explanation from an expert which is the requirement (REQ012). The screens are made easy to use by using UI Components in the implementation to allow the system to have high-quality graphics.

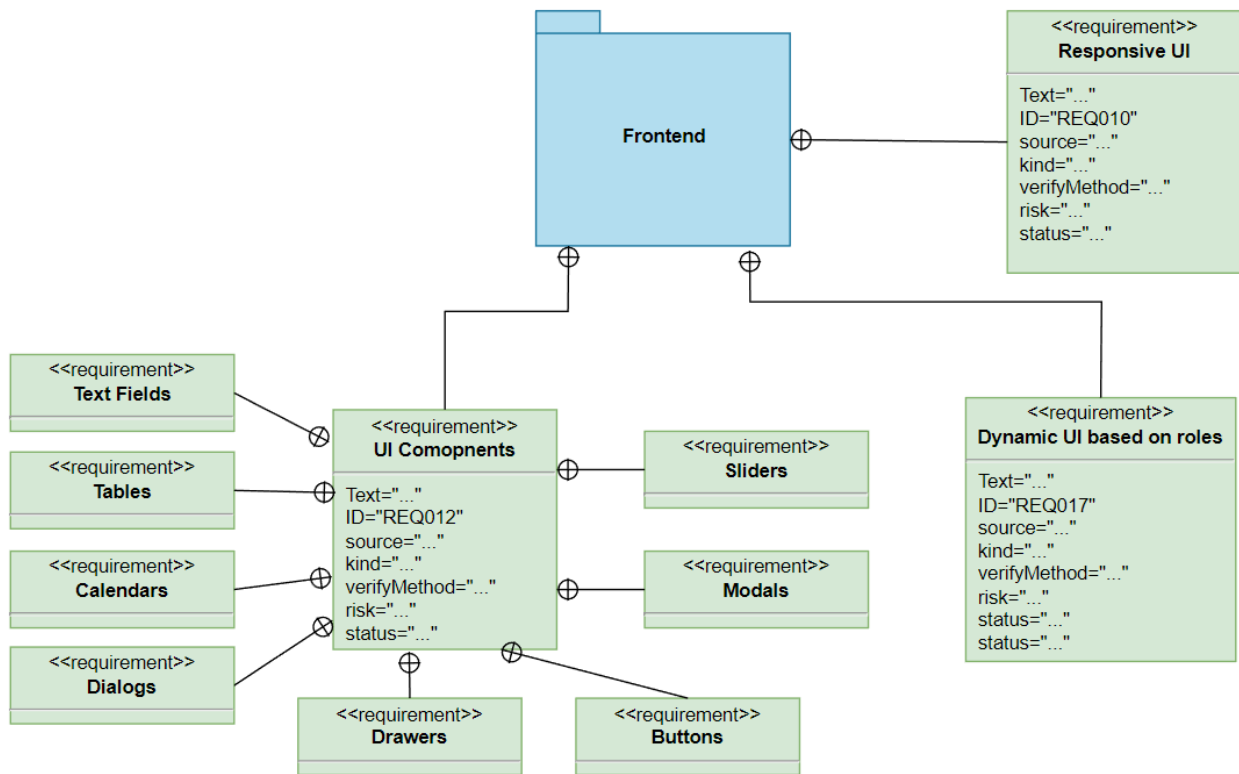


Figure 8 Front-end Requirement Diagram

3.2.3 Backend requirement diagram

The Backend requirement diagram is shown in figure 9. It is the part of the system that is responsible for processing the data and storing it in the database or sending it back to the Frontend. The data to be stored in the database should have the appropriate tables which is the requirement (REQ019) and the backend should be connected to the database which is the requirement (REQ025). A key feature that makes our system secure against malicious activities is the requirement (REQ011) that JSON Web Tokens are used by the system. With JWTs the system can authenticate the user sending data by verifying the identity of the user at each iteration through the tokens.

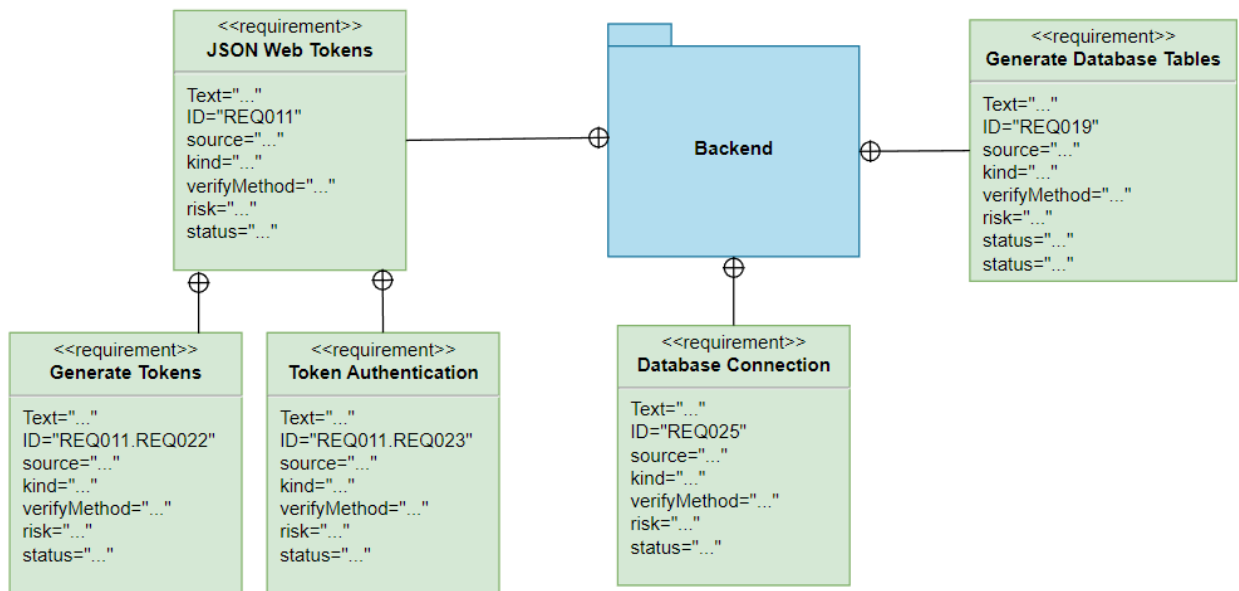


Figure 9 Backend Requirement Diagram

3.2.4 Database requirement diagram

The database stores all system data and is connected to the backend. In the beginning, we need to create the database which is required (REQ029) to be able to store the data within the tables. The tables should have relationships between them, depending on the role each table has. This way the backend will be able to pull the data according to the fields it needs.

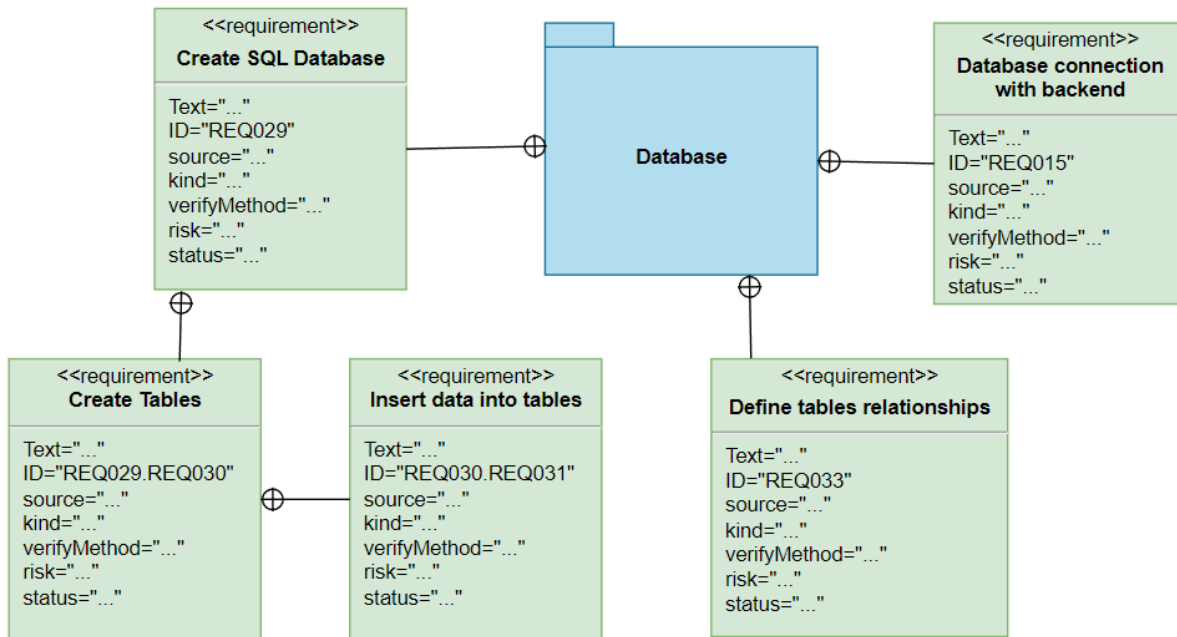


Figure 10 Repository Requirement Diagram

3.3 System Design

The system design will be analyzed using Unified Medical Language System (UMLS) diagrams. It is a key tool because it is a graphical language for visualizing and identifying the elements that make up a software system like the one being analyzed. With UML diagrams an overview of the system based on its functions can be made and can be much easier to understand for developers without the need for unnecessary explanation time. It still offers the possibility of easy understanding by people who are not part of the technical team and a complete analysis of the system before starting its implementation. UML diagrams are divided into several categories depending on whether they depict classes, packages, components, and the relationships between

them [46]. Five different UML diagrams developed for the system implementation will be presented below:

1. Component Diagram
2. Use case
3. Activity Diagram
4. Sequence Diagrams
5. Entity Relationship Diagram (ERD)

3.3.1 Component Diagram

The Component Diagram is used in large object-oriented systems to analyze the static models of the system and the relationships between them. It divides the system into smaller components and is a physical representation of the behavior that each component has with the others. Figure 11 shows the component diagram of the system with the main components. The Frontend, Backend, and Database are the main components of the system. Initially, the Web Browser runs the Frontend code to display the system on the screen. The Frontend uses libraries (Material UI, React-Bootstrap) as depicted in the figure so that it can be easy to use. It then communicates with the Backend using the Axios library that allows HTTP requests. The Backend communicates with the Frontend and the Database that stores the data. Finally, the Database uses MySQL.

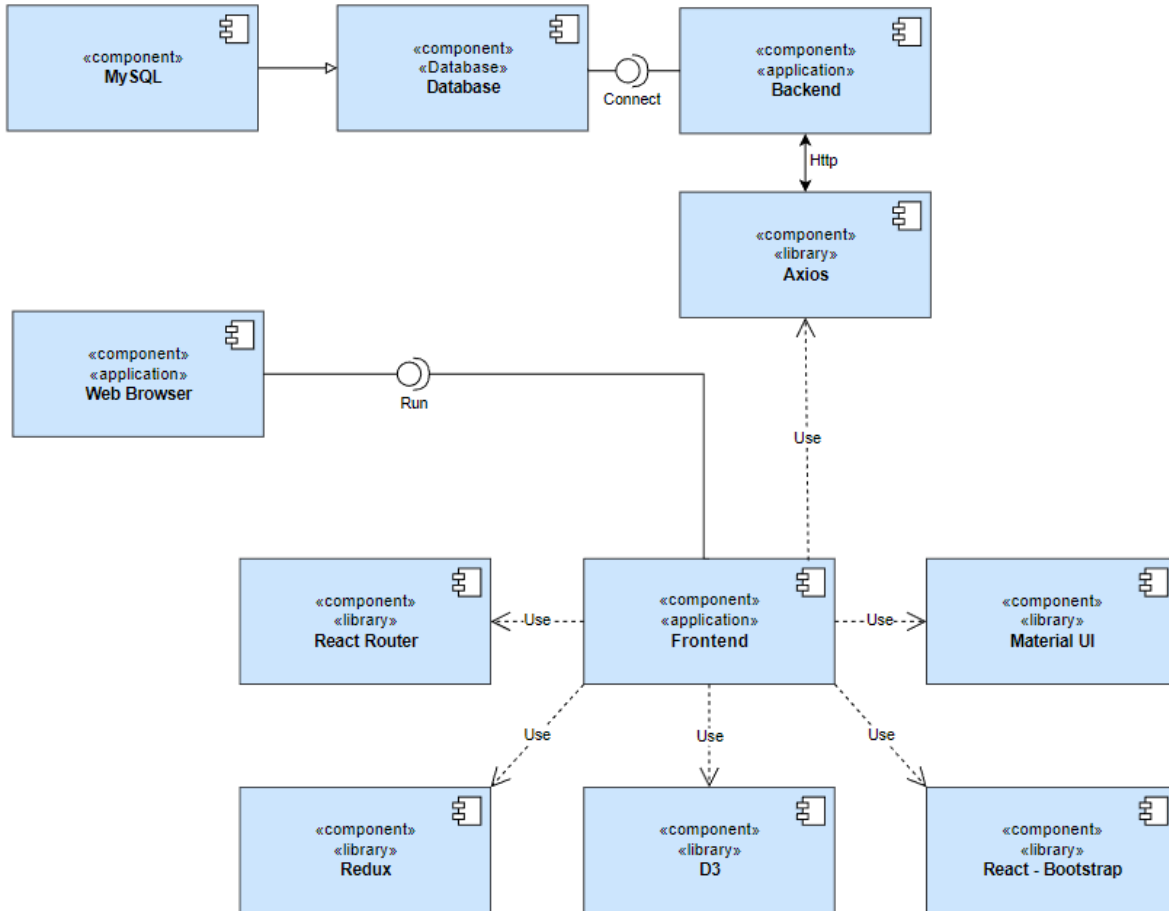


Figure 11 Main Component Diagram

3.3.2 Use case

The Use case diagram shows the interactions that the user (actor in UML) can have with the system. It contains the scenarios that the user can follow in the system with all the options provided. In the system being analyzed, we have two different roles, the simple user and the admin who can create as many users as he wants. Figure 12 shows the basic use case of the system with all the possible options that both roles can have.

Initially, the Admin can register or log in depending on whether they already have an account or not. Then he can view his account information in the dashboard and edit it. He can see the list of all the employees he has added, and he can add more or remove someone. Also, he can still view and edit the financial information of his employees and assign them tasks based on whatever model he wants. The tasks are displayed in the calendar which he can edit and finally, he can change his account password or delete his entire account with his employees together if he wants.

The simple user can log in to the system if the admin has added him and can view his account information in the dashboard and edit it. He can then see his financial details like his salary and in the calendar, he can see the tasks assigned to him. Finally, in the settings, he can only change his password but not delete his account as only the admin has this choice available.

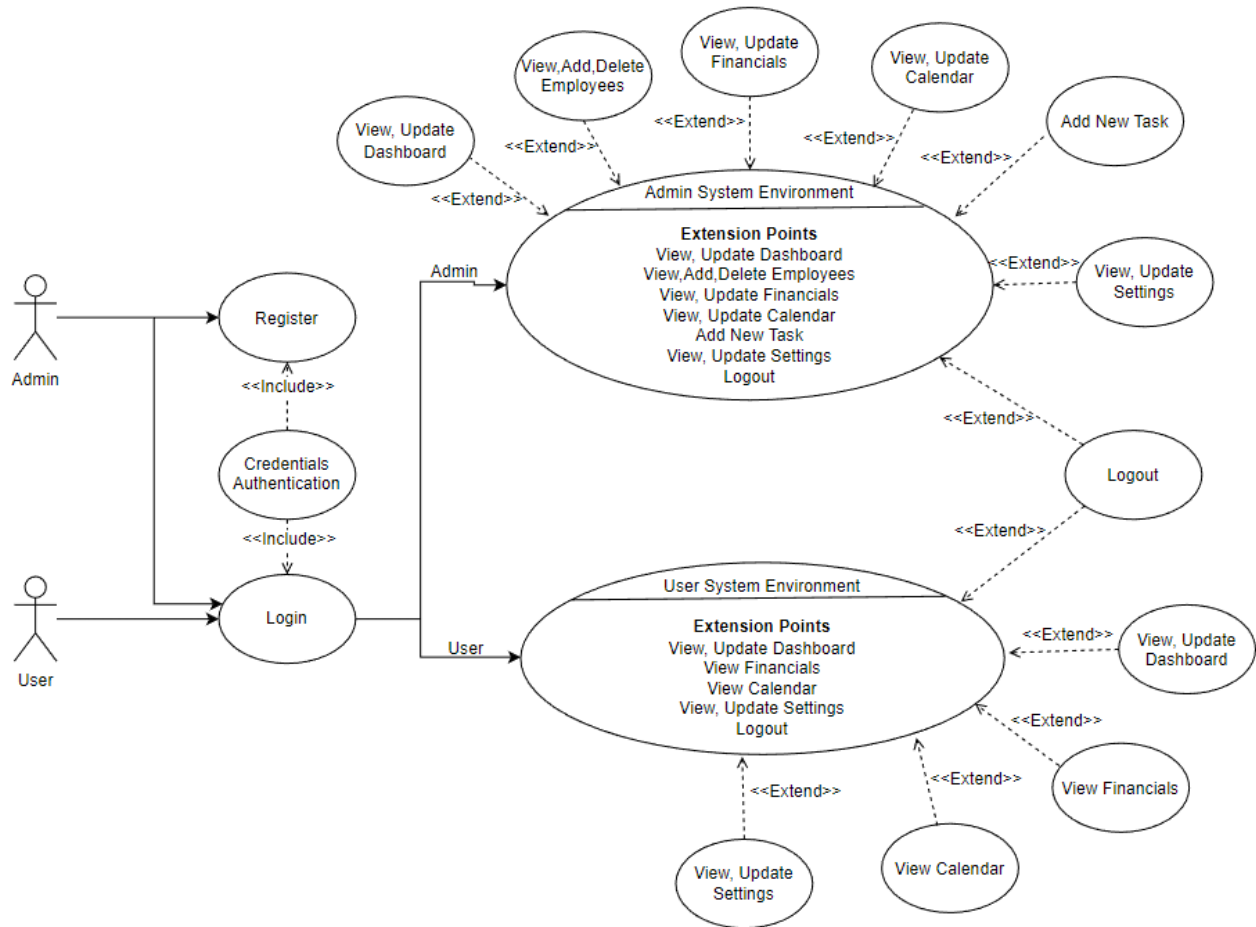


Figure 12 Main Use Case Diagram

The second use case is shown in Figure 13 and is a more specific scenario between the admin and a simple user. The admin can see all his employees and has three options if he wants. First, he can search for a specific employee, second, he can add an employee but form validation will have to be done with the employee details he declares and third, he can delete an employee with form validation forced. Then both roles can view the financials section, but the admin has the overall view of the expenses and number of employees while he can edit this data. The simple user can see his monthly or yearly earnings from the company.

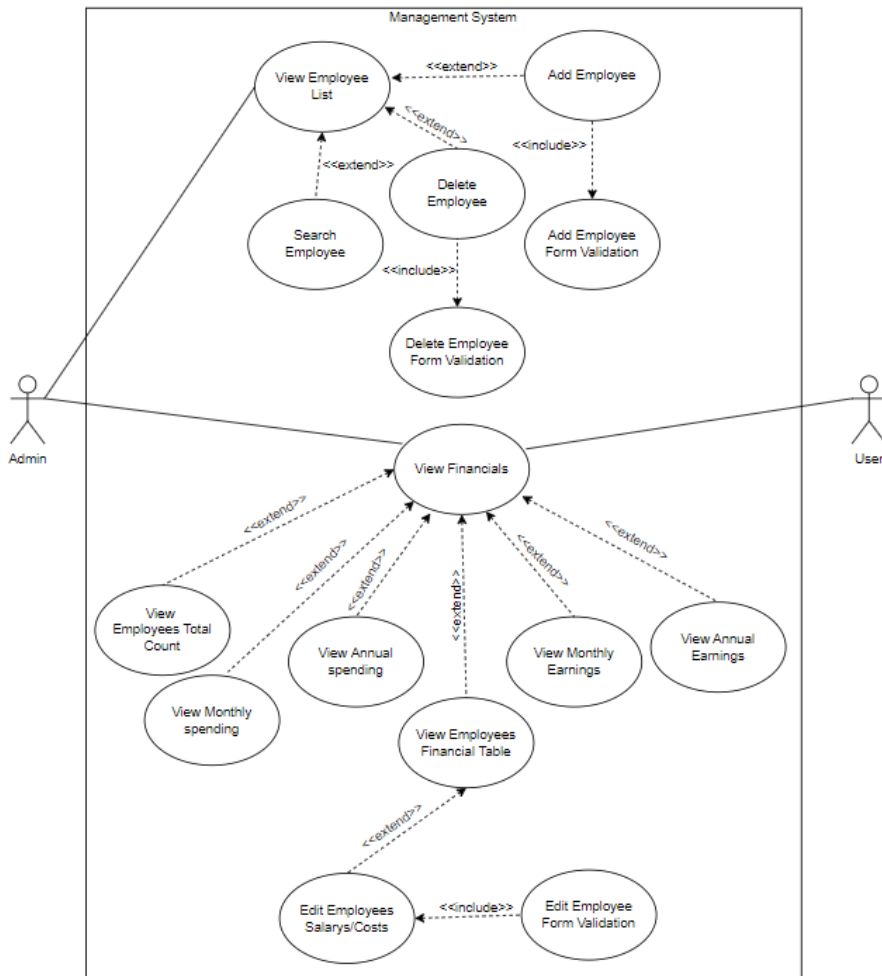


Figure 13 Use Case Scenario

3.3.3 Activity Diagram

The Activity Diagram shows the steps that take place in a system when the use case is executed. The user can see the activities he can have in the system each time he selects a path of activities. The main purpose of the diagram is to describe the flow from one activity to another rather than between objects in the system. Figure 14 shows the activity diagram of the simple user when using the system. It starts with login and only when authentication is done the user can enter the system. Then he has many options as mentioned above such as viewing the dashboard and editing his information. All activities also contain the option to log out of the system at any time the user chooses.

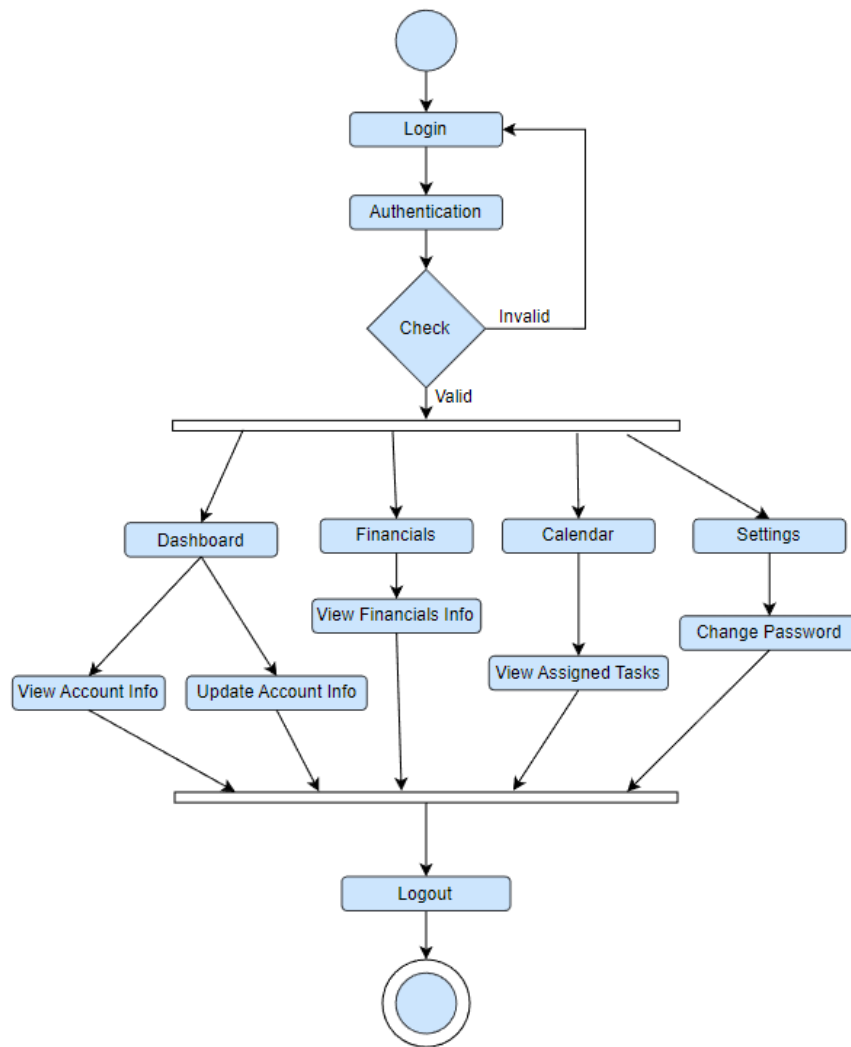


Figure 14 Simple User Activity Diagram

The Activity Diagram of the admin is shown in figure 15 and it is much larger than the simple user because he has many more permissions in the application. He can view and edit his employees that the simple user cannot and assign tasks in any model he wants. He can still manage the financials of his staff and renew their tasks. Finally, he can delete only his account which implies that all his employees' accounts will be automatically deleted.

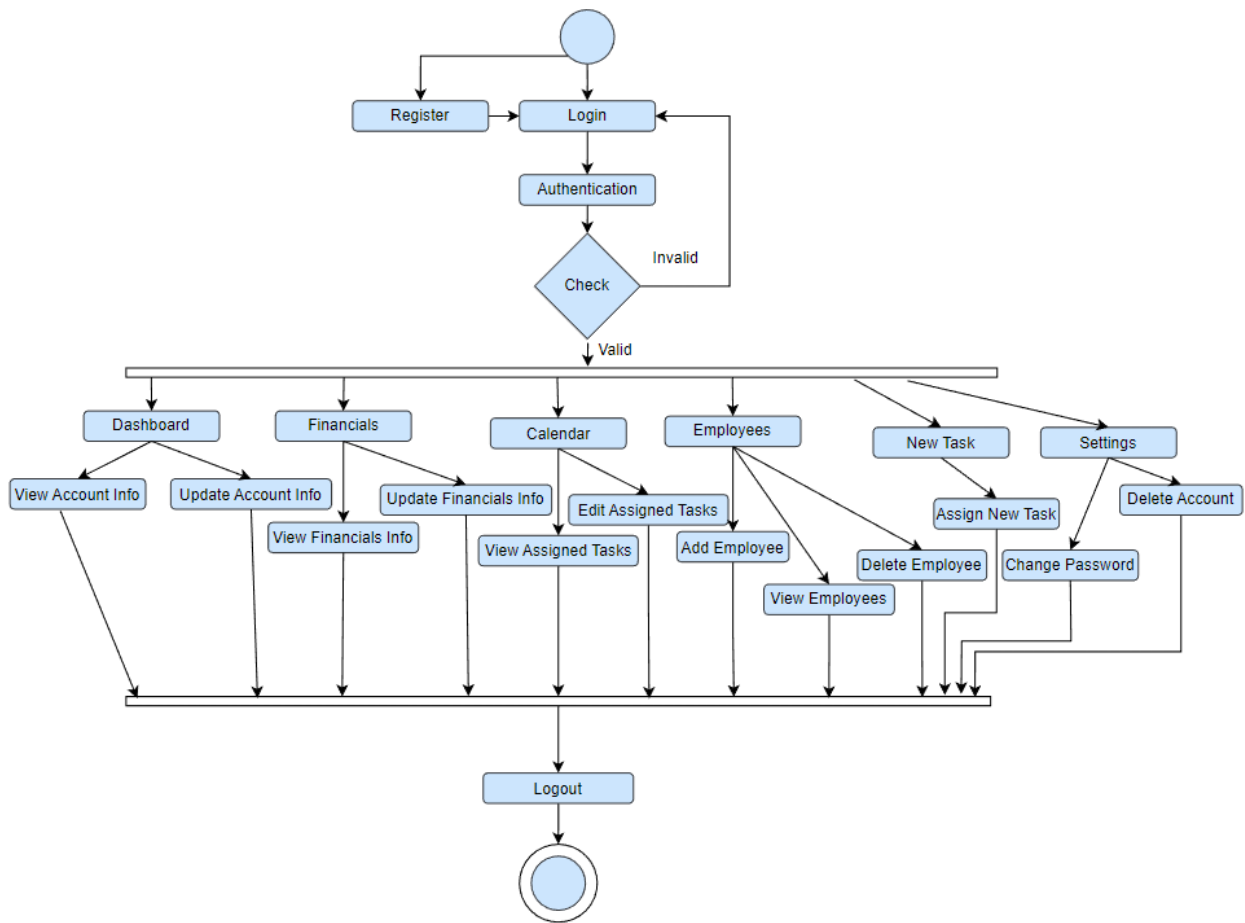


Figure 15 Admin Activity Diagram

3.3.4 Sequence Diagrams

Sequence Diagrams is an interaction diagram and illustrates how and in what order the objects in the system work together. The visualization is based on the time when the user starts using the system and shows step-by-step how the operations are performed. In many cases, the sequence diagrams become very large in scale because they have to show many functions. That is why in the system analyzed two diagrams were created for the admin and the simple user. Figure 16 shows the actor which is the admin and contains the dashboard, employees and financials sections.

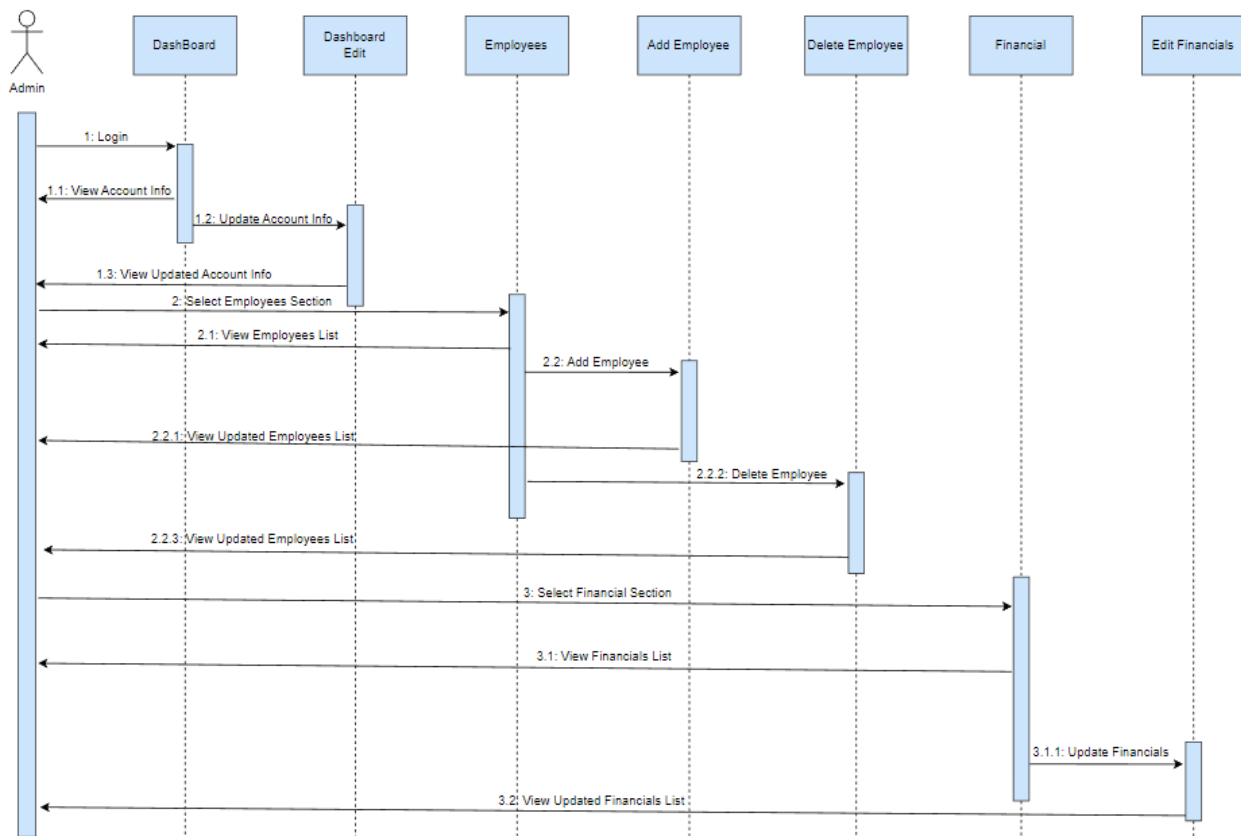


Figure 16 Admin First Sequence Diagram

Figure 17 shows the other sections for admin which are calendar, new task, settings and logout.

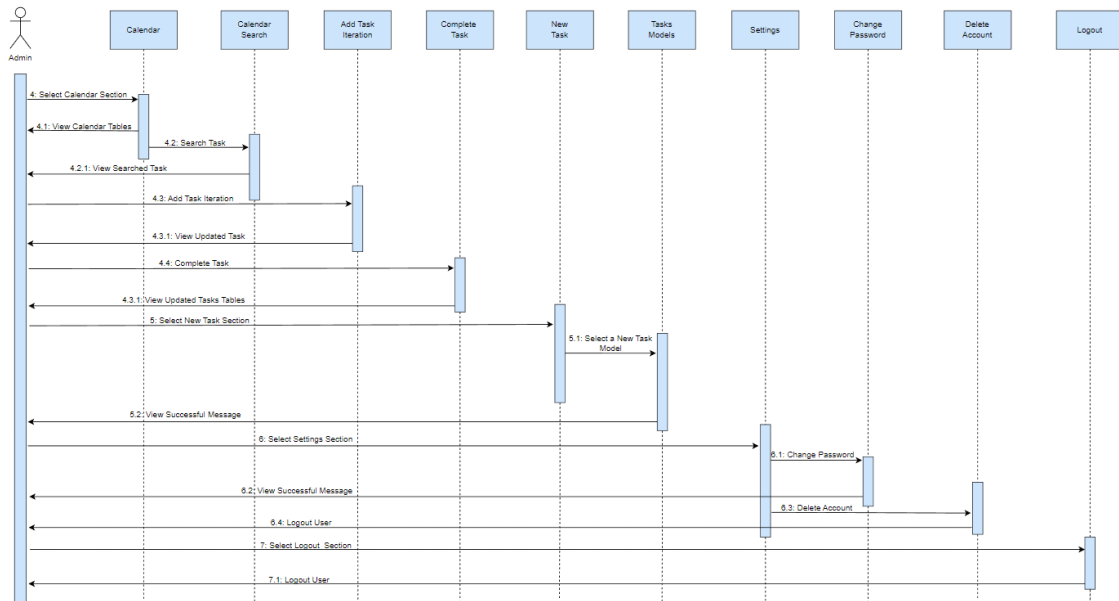


Figure 17 Admin Second Sequence Diagram

Finally, figure 18 shows the sequence diagram that has a simple user as an actor. The simple user has sections dashboard, financial, calendar, settings, and logout.

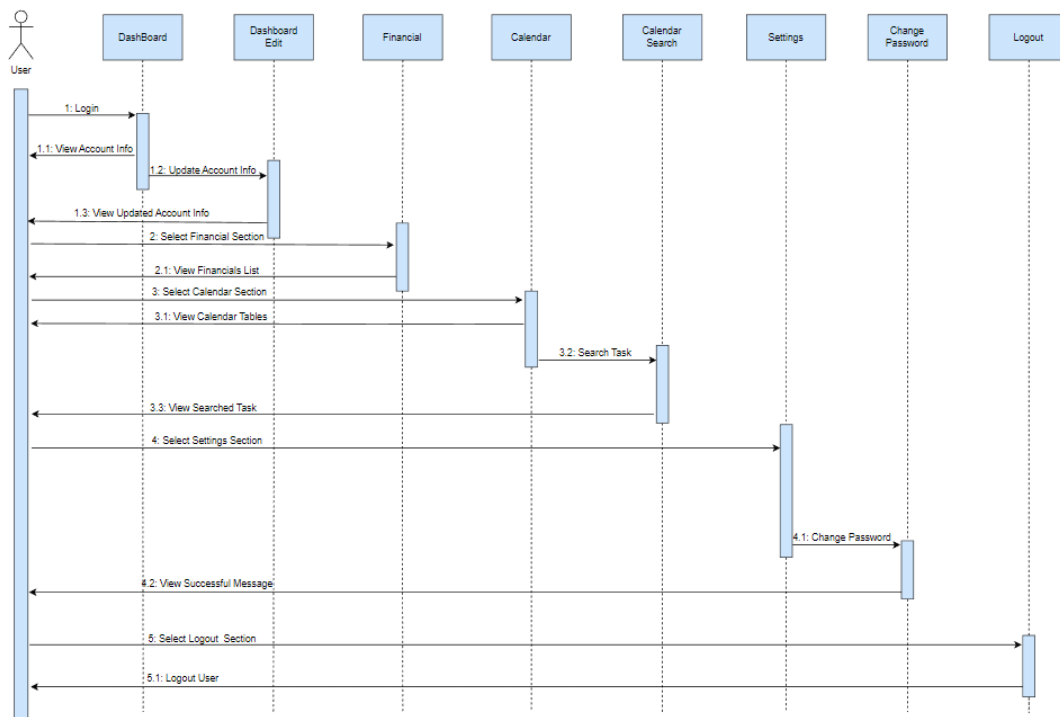


Figure 18 Simple User Sequence Diagram

3.3.5 Entity Relationship Diagram (ERD)

The Entity Relationship Diagram (ERD) is a flowchart that shows how entities in the database are connected. Figure 19 shows the ERD for the system being analyzed and shows in detail the tables that exist in the relational database. The key entity for the whole system is the users table which contains the basic information for all users and all other entities are directly linked to this entity. The roles for users are contained in the user_roles entity that defines when a user is an administrator or a simple user. Finally, each different model that the administrator assigns to simple users has its entity with information such as dates or costs.

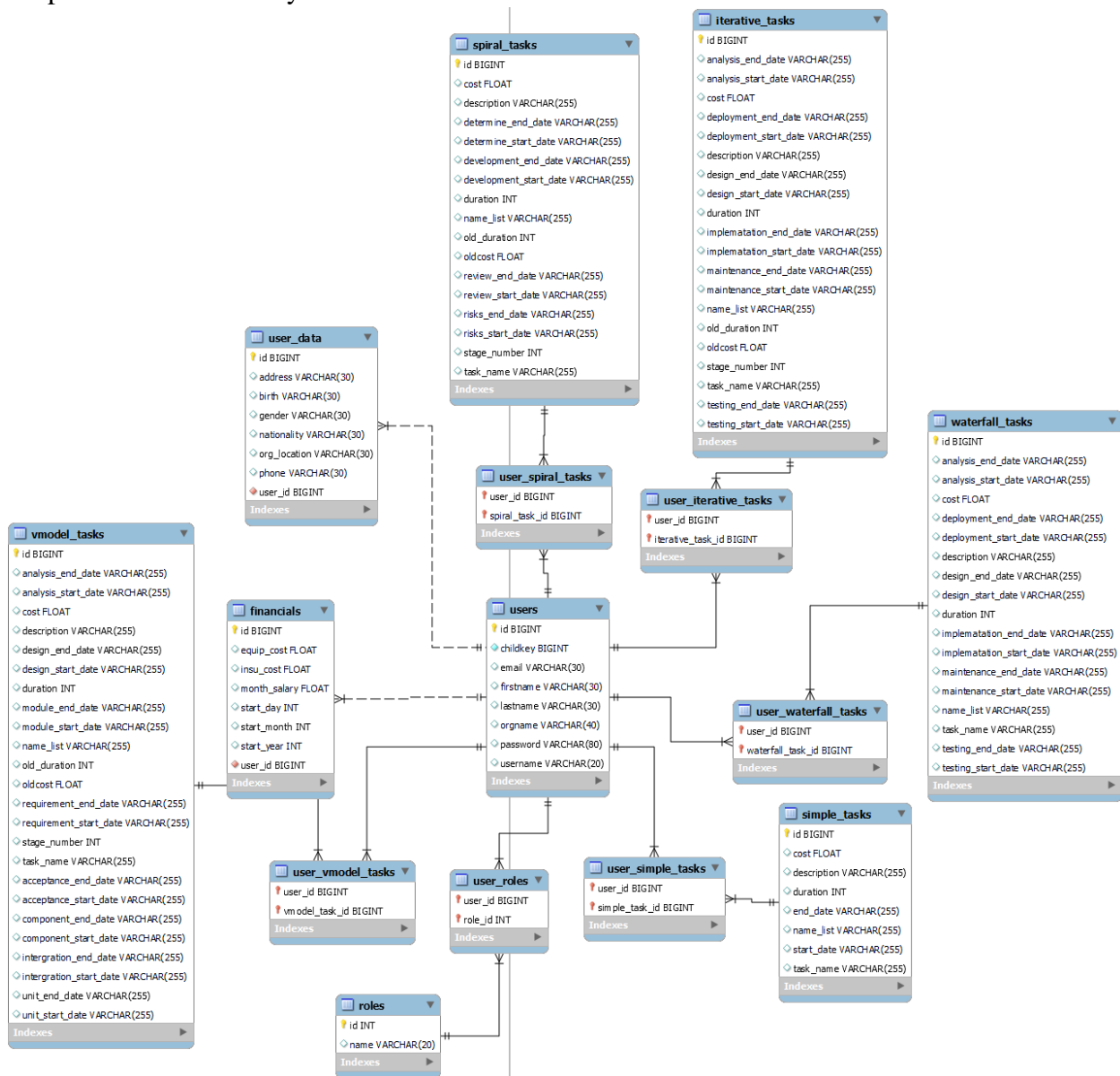


Figure 19 ERD

3.4 System Technologies

The technologies used to develop the system aim to provide the user with a system that is easy to use and safe to operate. Before the implementation of the system, research was conducted for innovative technologies that can fully interoperate with each other and have the ability to support complex systems such as the one being analyzed. They should be maintainable but also leave room for future additions to the system. The technologies chosen had in common that they were all quite popular and there is a huge community offering open-source and ready-made implementations for a variety of issues that need implementation. For the thesis, they had to be technologies that other developers could easily use in the future since the code would be open source. The technologies chosen for the implementation of the system are:

1. React.js : It is a JavaScript framework and is responsible for the User Interface (frontend). It is supported by a large community of developers and is one of the most popular choices.
2. Spring boot: It is responsible for data processing and storage (backend). It is one of the most famous Java frameworks and is very much used for developing web applications and microservices.
3. MySQL: MySQL is an open-source relational database management system (RDBMS). This system uses Structured Query Language (SQL) and this is where all the data of the system being analyzed is stored.

3.4.1 React.js

React.js is one of the most popular JavaScript frameworks in the world. It was developed in 2011 by developers and is used by very large companies like Netflix and Apple. It is the first choice for many programming companies as it has many advantages over other JavaScript frameworks [42]. The main advantages of React.js are:

- Its performance is quite high even on huge amounts of data
- It is easy enough to understand for programmers who want to develop systems quickly
- It offers long-term stability without the risk of interrupting upgrades.
- It is supported by a large and active community where someone can search for different implementations or even ask for help if needed
- Has a huge variety of component libraries like Material UI used in this system

For the master thesis, it was chosen because it needed a framework with these advantages discussed above and because it had to be on technologies that would allow future modifications and additions to the system easily. Figure 20 shows the code containing the main paths of the system, written in React.js.

```
1  import * as React from "react";
2  import Login from "../Login-Register/Login"; 1
3  import Register from "../Login-Register/Register";
4  import { BrowserRouter as Router, Routes, Route } from "react-router-dom";
5  import AdminSidebar from "../Welcome-Admin/AdminSidebar";
6  import { createTheme, ThemeProvider } from "@mui/material/styles";
7
8  function App() {
9
10     const theme = createTheme({ 2
11       palette: {
12         background: {
13           default: "#278ae6",
14         },
15       },
16     });
17
18     return ( 3
19       <ThemeProvider theme={theme}>
20         <div>
21           <Router> 4
22             <Routes>
23               <Route path="/" element={<Login />}></Route>
24               <Route path="/signup" element={<Register />}></Route>
25               <Route path="/adminDashboard/*" element={<AdminSidebar />}></Route>
26             </Routes>
27           </Router>
28         </div>
29       </ThemeProvider>
30     );
31   }
32
33   export default App;
```

Figure 20 React Js Example

Point 1 shown in figure 20 all the imports that the code needs to run. They are data stored in other files or libraries ready for use. Point 2 is the JavaScript code that consists of a few lines as in the figure or can reach hundreds of lines when we have complex functions that need to run. In this example, we are simply giving a color code to a variable that will be used as a theme throughout the application. Point 3 is the HTML code and React components used. A component can be written with code locally in the application in another file or it can be ready-made from a library such as Material UI. In all cases, a component should always be opened and closed depending on whether it has internal components or is called by itself. In the example, in figure 20 the Router component opens on line 22 and closes on line 27 because it has other components called internally, while the Route component opens and closes on line 23. The purpose of these lines of code throughout the figure is to create paths that the user can have within the application to switch pages.

The Material UI library was used because we needed the application to be easy to use and have many screens dynamic in the user interface. This library provides dozens of ready-made components that can be very easily integrated into the code and are immediately functional. Figure 21 shows a table of employees that the admin view when managing his staff. This table is a Material UI component called TableContainer and with many changes and code additions, it was configured as shown in the figure.



The image shows a screenshot of a web application interface. At the top, there is a search bar with a magnifying glass icon and the text "Search". Below the search bar is a table with a dark blue header and five data rows. The table has four columns: "Username", "First Name", "Last Name", and "Email". The data rows contain the following information:

Username	First Name	Last Name	Email
anita	Anita	Sayid	anita@gmail.com
stanley	Stanley	Knife	stanley@gmail.com
willie	Willie	Makit	willie@gmail.com
penny	Penny	Biding	penny@gmail.com
chris	Chris	Creem	chris@gmail.com

Figure 21 Material UI Example

3.4.2 Spring boot

It is one of the most popular open-source Java-based frameworks used for standalone and commercial applications. It is an ideal tool for developers who need to create microservices for their applications or create web applications. Spring boot is very popular in the developer's community because it offers the dependency injection feature that allows objects to have their dependencies that the Spring container then integrates into them [43]. In this way, developers can create applications that consist of unified components which is the central idea behind microservices.

In the analyzed system, Spring boot was chosen because it is very easy to create endpoints to send data to the frontend. Still, a very big advantage is offered by the Java Persistence API (JPA) and it is a Jakarta EE application programming interface specification to maintain relational databases like the one used in the system. Many dependencies were used for the implementation such as MySQL-connector-java which helps to connect the backend to the database or spring-boot-starter-data-JPA integrating the JPA discussed above. Figure 22 shows a small piece of code showing how a class (user) creates relationships within the database with other classes that also represent a table within the database. At point 1 a many-to-many relationship is created with the user_roles class containing their system roles. This means that many users can have many user_roles and the other way around. In point 2 of the figure, however, we can see that we have an OneToOne relationship which means that a user can have one Financial and the other way round.

```
49 | @ManyToMany(fetch = FetchType.LAZY)
50 | @JoinTable( name = "user_roles",
51 |             joinColumns = @JoinColumn(name = "user_id"),
52 |             inverseJoinColumns = @JoinColumn(name = "role_id"))
53 | private Set<Role> roles = new HashSet<>();
54 |
55 |
56 | @OneToOne(mappedBy = "user", fetch = FetchType.LAZY,
57 |           cascade = CascadeType.ALL)
58 | private Financials financials ;
59 |
60 | @OneToOne(mappedBy = "user", fetch = FetchType.LAZY,
61 |           cascade = CascadeType.ALL)
62 | private UserData userData ;
63 |
64 | @ManyToMany(fetch = FetchType.LAZY,
65 |            cascade = {
66 |                CascadeType.PERSIST,
67 |                CascadeType.MERGE
68 |            }
69 |            )
70 | @JoinTable(name = "user_simpleTasks",
71 |           joinColumns = { @JoinColumn(name = "user_id") },
72 |           inverseJoinColumns = { @JoinColumn(name = "simpleTask_id") })
73 | private Set<SimpleTask> simpleTasks = new HashSet<>();
```

Figure 22 Spring Boot Class Example

Figure 23 shows the controller of the user class and consists of two endpoints, each for a different purpose. At point 1 there is a Get with which when someone requests data with the appropriate URL ("/api/auth /getEmployees/{id}") and the appropriate id then it will return all employees corresponding to that id i.e. the id of the administrator as depicted in line 150. At point 2 there is an endpoint with Delete which means that when someone calls this endpoint with the appropriate URL ("api/auth/deleteEmployee/ {username}") then it will delete the employee from the database according to the username given by the user who can only be the administrator for this call.

```

144 // Get a array List with users and the id of their admin id example:
145 //http://localhost:8080/api/auth/getEmployees/1 1
146
147 @GetMapping("/getEmployees/{id}")
148 public ResponseEntity<List<User>> getEmployees(@PathVariable("id") long id) {
149     try {
150         List<User> users = userRepository.findByChildkey(id);
151         if (users.isEmpty()) {
152             return new ResponseEntity<>(HttpStatus.NO_CONTENT);
153         }
154         return new ResponseEntity<>(users, HttpStatus.OK);
155     } catch (Exception e) {
156         return new ResponseEntity<>(HttpStatus.INTERNAL_SERVER_ERROR);
157     }
158 }
159
160
161
162 //Delete employee based of the id 2
163 // http://localhost:8080/api/auth/deleteEmployee/5
164 @DeleteMapping("/deleteEmployee/{username}")
165 public ResponseEntity<?> deleteEmployee(@PathVariable("username") String username) {
166     User user = userRepository.findByUsernameIs(username);
167     if (user.getIterativeTasks().size() == 0 && user.getWaterfallTasks().size() == 0 && user.getSpiralTasks().size() ) {
168         userRepository.deleteById(user.getId());
169         return ResponseEntity.ok(new MessageResponse("Employee deleted successfully!"));
170     } else {
171         return ResponseEntity.ok(new MessageResponse("Employee is assigned to tasks can't deleted."));
172     }
173 }

```

Figure 23 Spring Boot Controller Example

3.4.3 MySQL

MySQL is an open-source relational database management system (RDBMS) that is based on the structured query language (SQL). It was developed and maintained by Oracle Corporation and runs on all platforms such as Windows and Linux which are among the most widely used. MySQL is one of the most famous systems in the world because it offers the ability to back up data and ensures that data is not going to be lost through recovery strategies. It manages relational databases and allows the developer to add, edit and view their data within the database [44].

4 System Representative Use Case Scenarios

During the implementation of the system, priority was given to all the UML diagrams analyzed above as the requirements that the system should implement and the basic structure that it will have in its entirety had to be clearly defined. It is very basic before starting any implementation of a system that these steps are done very properly and in detail because there is a risk that the result will be much too time-consuming to implement and less efficient. In this chapter, the result of the implementation will be analyzed for each possible scenario that may arise from all the roles. In this way, the structure of the system and the possibilities it provides for the user will be fully understood.

1.1 Representative Use Case Scenarios

The system is aimed at companies or teams of developers who have a team leader and assigns tasks to employees or simply other team members. Based on this central idea the system supports two roles, admin and simple user. Each admin can have as many workers as they want but the worker-simple users can only have one administrator. The structure is shown in figure 24 and the admin is in the middle handling his simple users on the side.

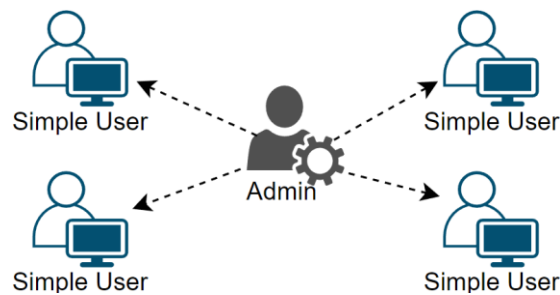
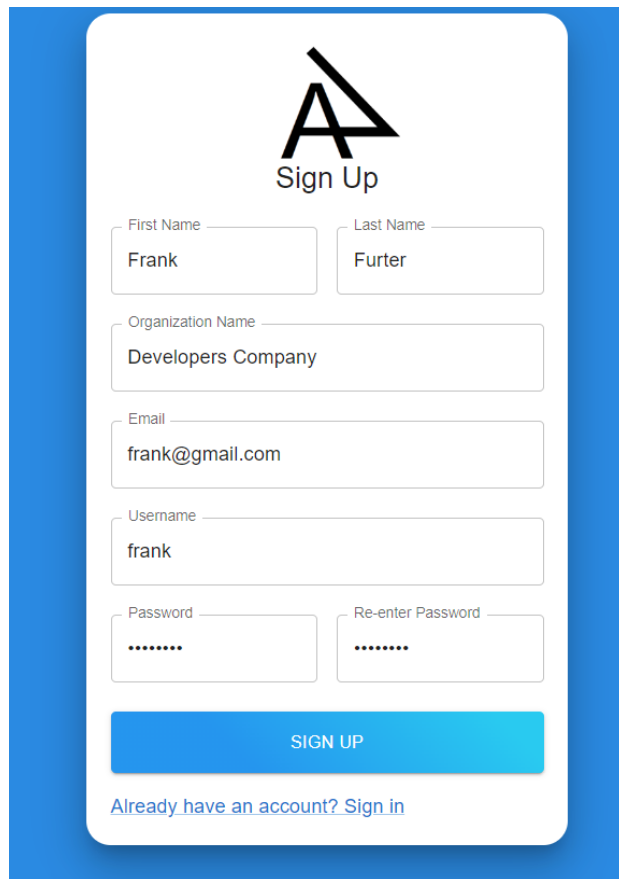


Figure 24 System Structure

4.1.1 Admin Role

When opening the application for the first time the admin will have to register and fill in his information. Only the admin is allowed to register, and the registration form is shown in figure 25.



The registration form is titled "Sign Up" and features a logo consisting of a stylized 'A' with a diagonal line. The form includes the following fields and values:

- First Name: Frank
- Last Name: Furter
- Organization Name: Developers Company
- Email: frank@gmail.com
- Username: frank
- Password: [masked]
- Re-enter Password: [masked]

A blue "SIGN UP" button is located at the bottom of the form. Below the button, there is a link: "Already have an account? [Sign in](#)".

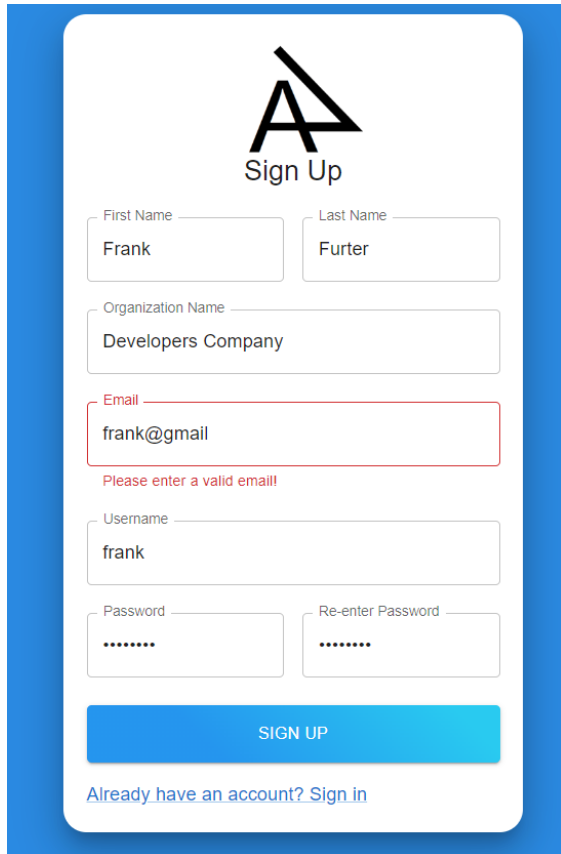
Figure 25 Register Page

The information that the application requests from Admin are:

1. First Name
2. Last Name
3. Organization Name
4. Email
5. Username
6. Password

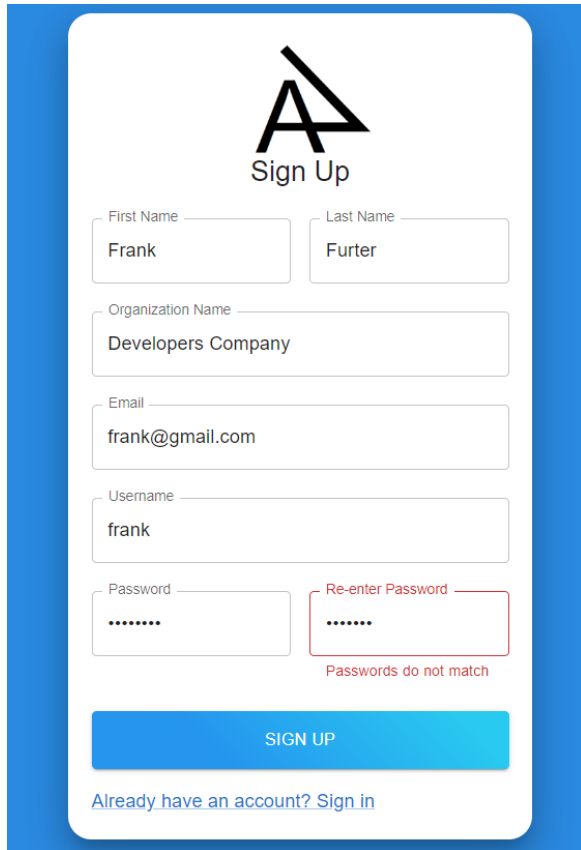
7. Re-Password

The admin after entering the data required then the system validates if all the fields are filled in and if they are in the correct format. The correct format changes depending on the field like the email is checked if the one given with @ in it or the passwords must be the same. Figure 26 and 27 shows how the application notifies the admin that he has given incorrect data.



The screenshot shows a 'Sign Up' form with the following fields and values: First Name: Frank, Last Name: Furter, Organization Name: Developers Company, Email: frank@gmail, Username: frank, Password: [masked], and Re-enter Password: [masked]. A red border highlights the email field, and a red error message below it reads 'Please enter a valid email!'. A blue 'SIGN UP' button is at the bottom, with a link 'Already have an account? Sign in' below it.

Figure 27 Register Wrong Email Page



The screenshot shows the same 'Sign Up' form as Figure 27, but with the email field filled as 'frank@gmail.com'. The password field is filled with '*****' and the re-enter password field is also filled with '*****'. A red border highlights the re-enter password field, and a red error message below it reads 'Re-enter Password' and 'Passwords do not match'. A blue 'SIGN UP' button is at the bottom, with a link 'Already have an account? Sign in' below it.

Figure 26 Register Wrong Password Page

Once the admin successfully registers then he can enter the application with the details he provided during registration. Figure 28 shows the login of the application which is the same for both roles. He needs the username and password to log in.

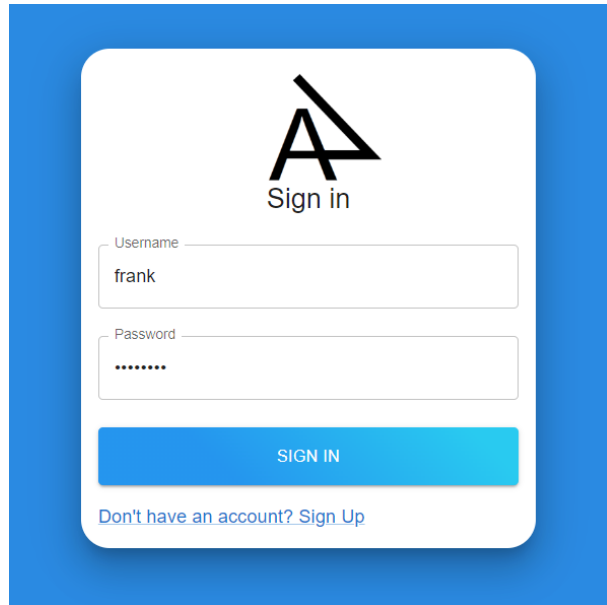


Figure 28 Login Page

The system validates that the information provided in the login form is correct and the admin enters his dashboard. Figure 29 shows the admin's dashboard and his account information. The left part shows the details:

1. First Name
2. Last Name
3. Username
4. Gender
5. Birthday
6. Nationality

In the right part of the screen the name and location of the company organization are displayed and below it the contact details such as email, phone, and address. Admin can edit all this information by pressing the button at point 2 shown in the figure.

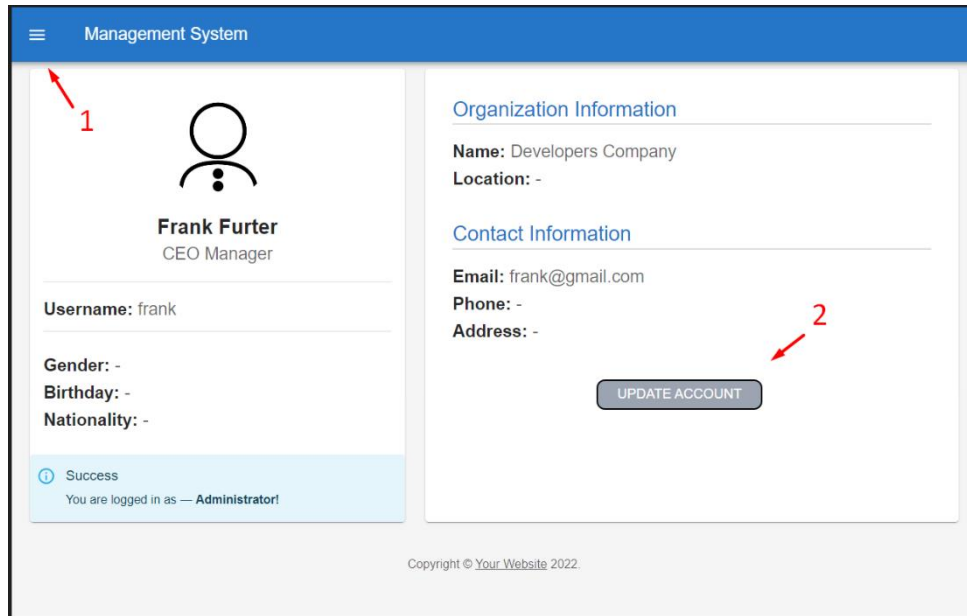


Figure 29 Admin Dashboard Page

When the button at point 2 is pressed a window opens which is shown in figure 30.

EDIT THE ACCOUNT DATA

REGISTRATION DATA

First Name: Frank
Last Name: Furter

Organization Name: Developers Company

EDIT

EXTEND USER DATA

Organization Location: -

Gender: -
Nationality: -

Address: -

Phone: -
Birthday: -

EDIT

CLOSE

Figure 30 Admin Update Info Page

The information provided in the registration is already filled in and the rest is blank. All fields can be changed and at the end, the corresponding EDIT button can be pressed to save or the CLOSE button to close the window. The other option the admin has is to press the button located at point 1 of his dashboard and open the navigation menu shown in figure 31.

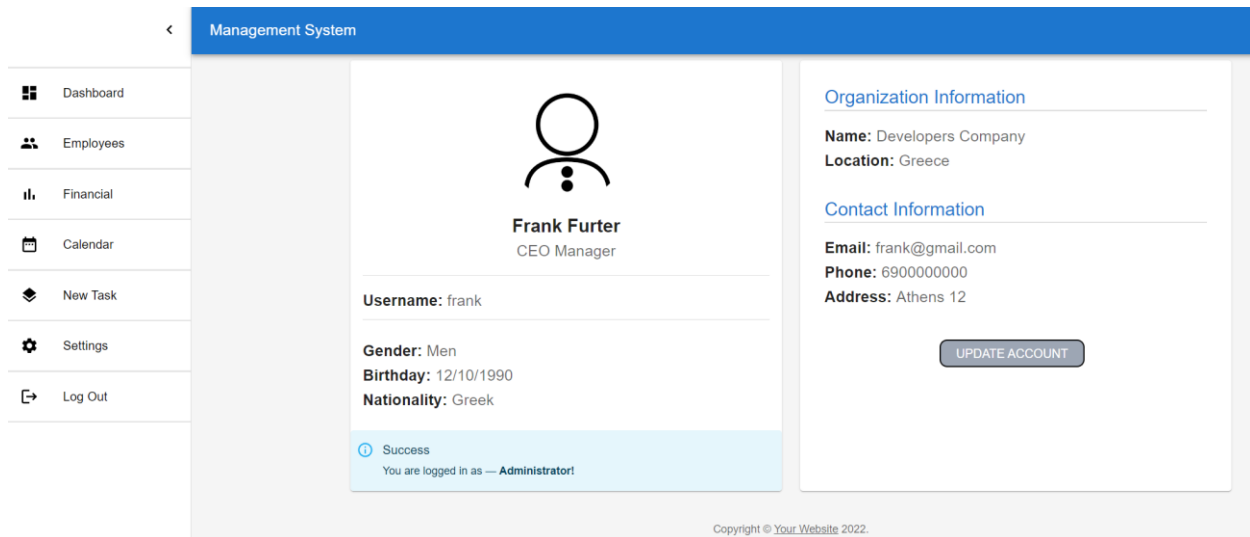


Figure 31 Admin Menu Page

The options that the Admin role has for navigating through the system are:

1. Dashboard
2. Employees
3. Financial
4. Calendar
5. New Task
6. Settings
7. Log Out

Each category has different functions which will be analyzed below. When the Employees option is clicked then the admin sees the overall view of his employees as shown in figure 32.

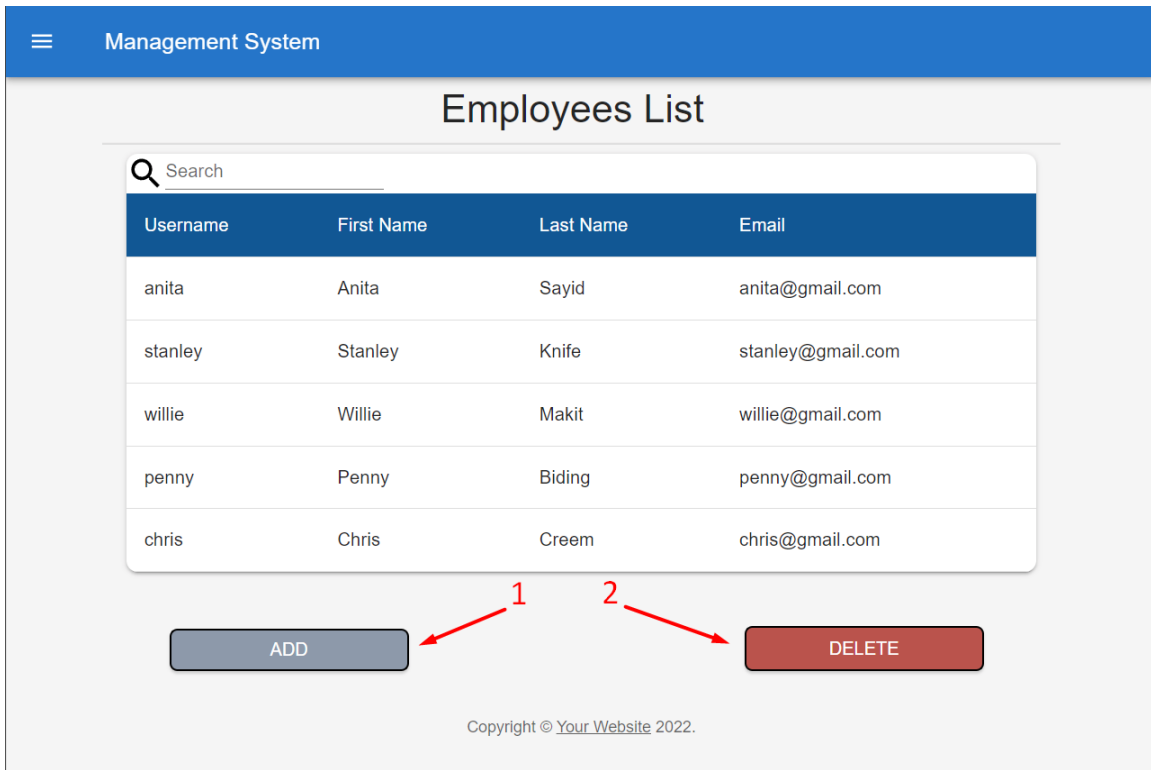


Figure 32 Admin Employees Page

The table at the beginning is empty but to make the description easier to understand I added some employees at the beginning. The table shows employee details such as Username, First Name, Last Name, and Email. The admin has two options and that is to add an employee or delete one. When he clicks to add an employee then the window is shown in figure 33.

ADD A EMPLOYEE

EMPLOYEE REGISTRATION

First Name Anita	Last Name Sayid
Email anita@gmail.com	
Username anita	
Set Monthly Salary 1200	Set Insurancer Cost 120
Set Equipment Cost 30	Select Start Date: 07/27/2022
Set Password *****	Re-enter Password *****

Figure 33 Admin Add Employees Page

The application demands the admin to enter the details for the employee they wish to add and the details are:

1. First Name
2. Last Name
3. Email
4. Username
5. Monthly Salary
6. Insurance Cost
7. Equipment Cost
8. Start Date
9. Password
10. Re-Password

When the simple user-employee is added then the simple user will be able to log in with the details given to him by the admin and change his password himself for security reasons. When the admin wishes to delete an employee then the window is shown in figure 34.

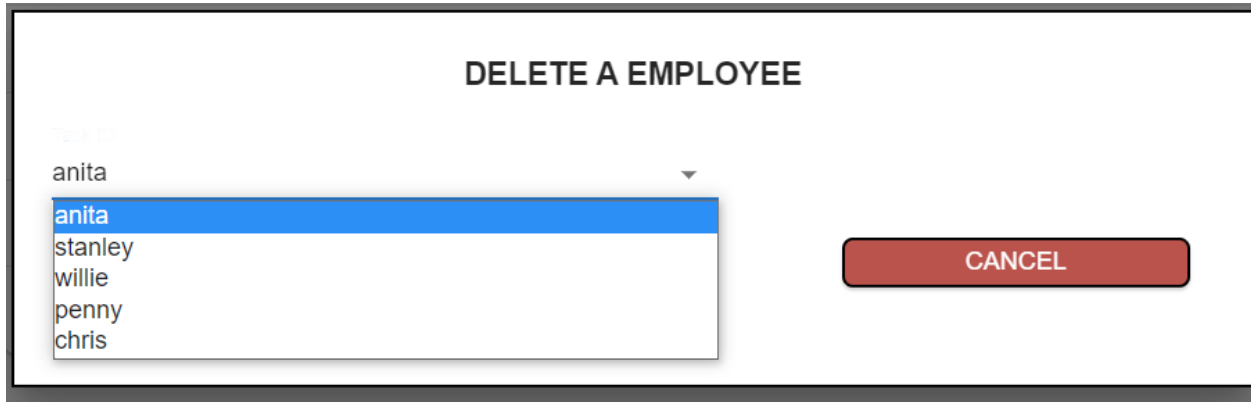


Figure 34 Admin Delete Employees Page

Admin selects the employee he chooses to delete and clicks the DELETE button located below the drop-down button with usernames in figure 34 (not shown in the example). The next section that the admin role can select from the navigation menu is the financial section which is shown in figure 35.

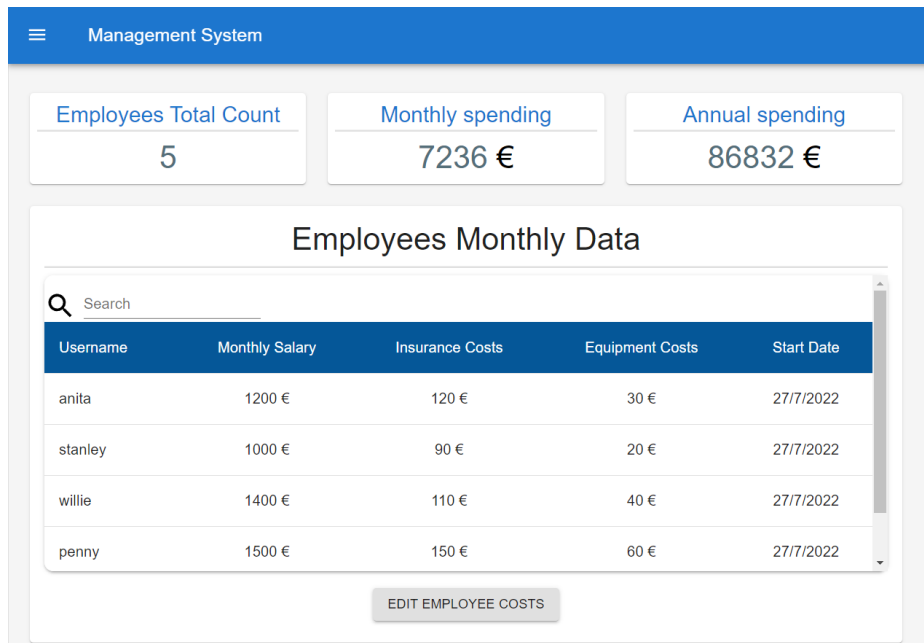


Figure 35 Admin Financial Page

In this section, he has an overview of the financial data of his company or group. At the top, he can see the total number of employees, the monthly expenses that the team has, and the annual expenses calculated based on salaries and the equipment and insurance expenses of all team members. At the bottom is a table that has detailed financial information for each employee and the date they started working for the company. There is still an option to edit the costs that employees have by clicking the button at the bottom of the screen. Figure 36 shows the window for editing costs.

The screenshot shows a web form titled "EDIT THE EMPLOYEE FINANCIAL DATA". Inside the form, there is a dashed box labeled "EMPLOYEE MONTHLY COSTS". Within this box, there are four input fields: a dropdown menu for "Username" with "anita" selected, a text input for "Set Equipment Cost" with the value "30", a text input for "Set Monthly Salary" with the value "1200", and a text input for "Set Isurancer Cost" with the value "120". Below the dashed box, there are two buttons: a blue "EDIT" button and a red "CANCEL" button.

Figure 36 Admin Edit Employee Financials Page

Depending on the username chosen by the admin, he can edit the expenses for the employee's equipment, salary, and insurance. The next section that the admin role can choose is a new task which is shown in figure 37.

The screenshot shows a web interface for assigning a simple task. At the top, there are five tabs: SIMPLE TASK, WATERFALL, ITERATIVE, SPIRAL, and VMODEL. The 'SIMPLE TASK' tab is selected. Below the tabs is a form titled 'Assign Simple Task'. The form contains several sections: 'Task Name' with a text input field; 'Description' with a larger text area; 'Assign To' with a dropdown menu currently showing 'Employees'; 'Task Dates' with two date pickers, both set to '27-07-2022'; and 'Estimate Cost' with two lines of text: 'Total cost for the task: 0 €' and 'Total duration for the task: 1 days'. At the bottom right of the form is a blue button labeled 'ASSIGN'.

Figure 37 Admin Add Simple Task Page

The admin has five options as to the type of task he can assign:

1. Simple Task
2. Waterfall Task
3. Iterative Task
4. Spiral Task
5. V-Model Task

The first task is the simple task which can be anything up to something complex. The other four are SDLC models and have been detailed above each one separately. In the shown figure 37 the Simple Task above is the first task the admin sees and can provide a title, description, employees who are assigned to it, and a start and end date. Based on the days the admin gives for a deadline, the cost of the task is calculated based on the salaries of the employees selected. The next task he can assign is the Waterfall Task shown in figure 38.

Waterfall Sequential Phases Dates

1) Requirement Gathering and analysis

Analysis Start Date: Phase Duration (Days): 2 Analysis End Date:

2) System Design

Design Start Date: Phase Duration (Days): 2 Design End Date:

3) Implementation

Implementation Start Date: Phase Duration (Days): 1 Implementation End Date:

4) Integration and Testing

Testing Start Date: Phase Duration (Days): 2 Testing End Date:

5) Deployment of system

Deployment Start Date: Phase Duration (Days): 2 Deployment End Date:

6) Maintenance

Maintenance Start Date: Phase Duration (Days): 2 Maintenance End Date:

Estimate Cost

Total cost for the task: 1124 €

Total duration for the task: 11 days

Figure 38 Admin Add Waterfall Task Page

The Waterfall Task has a title, description, assigned users and the six phases that comprise the SDLC model discussed in the Background section. Each phase has a start and end date, and the application calculates the total cost of the task based on the sum of the days of each phase and the employees selected. The next task is the Iterative task which is shown in Figure 39. It has the same phases as the Waterfall task but is iterative as analyzed in the Background chapter.

SIMPLE TASK
WATERFALL
ITERATIVE
SPIRAL
VMODEL

Assign Iterative Task

Task Name

Task Name
Create Airlines program

Description

Description
Create Airlines program for the next summer.

Assign To

Employees
chris stanley

First Stage Phases Dates

1) Requirement Gathering and analysis

Analysis Start Date: 27-07-2022	Phase Duration (Days): 2	Analysis End Date: 28-07-2022
------------------------------------	--------------------------	----------------------------------

2) System Design

Design Start Date: 29-07-2022	Phase Duration (Days): 2	Design End Date: 30-07-2022
----------------------------------	--------------------------	--------------------------------

3) Implementation

Implementation Start Date: 31-07-2022	Phase Duration (Days): 1	Implementation End Date: 31-08-2022
--	--------------------------	--

4) Integration and Testing

Testing Start Date: 02-08-2022	Phase Duration (Days): 2	Testing End Date: 03-08-2022
-----------------------------------	--------------------------	---------------------------------

5) Deployment of system

Deployment Start Date: 04-08-2022	Phase Duration (Days): 2	Deployment End Date: 05-08-2022
--------------------------------------	--------------------------	------------------------------------

6) Maintenance

Maintenance Start Date: 06-08-2022	Phase Duration (Days): 2	Maintenance End Date: 07-08-2022
---------------------------------------	--------------------------	-------------------------------------

Estimate Cost

Total cost for the task: 963 €

Total duration for the task: 11 days

Figure 39 Admin Iterative Task Page

The next task is the Spiral task which is shown in figure 40 and has four phases.

SIMPLE TASK WATERFALL ITERATIVE **SPIRAL** VMODEL

Assign Spiral Task

Task Name

Task Name
Create Hospital program

Description

Description
Create Hospital program for the patients.

Assign To

Employees
willie chris penny

Spiral Four Quadrants Dates First Stage

1) Identifying and understanding requirements

Requirements Start Date: 27-07-2022	Phase Duration (Days): 2	Requirements End Date: 28-07-2022
--	--------------------------	--------------------------------------

2) Performing risk analysis

Analysis Start Date: 29-07-2022	Phase Duration (Days): 2	Analysis End Date: 30-07-2022
------------------------------------	--------------------------	----------------------------------

3) Implementation

Implementation Start Date: 01-08-2022	Phase Duration (Days): 3	Implementation End Date: 03-08-2022
--	--------------------------	--

4) Integration and Testing

Testing Start Date: 04-08-2022	Phase Duration (Days): 3	Testing End Date: 06-08-2022
-----------------------------------	--------------------------	---------------------------------

Estimate Cost

Total cost for the task: 1592 €

Total duration for the task: 10 days

ASSIGN

Figure 40 Admin Add Spiral Page

The last task is the V-Model task which has eight phases and is illustrated in figure 41.

SIMPLE TASK
WATERFALL
ITERATIVE
SPIRAL
VMODEL

Assign Vmodel Task

Task Name

Task Name
Create Supermarket program

Description

Description
Create Supermarket program for products.

Assign To

Employees
stanley anita penny

First Stage Phases Dates

1) Business requirement analysis:

Requirements Start Date: 27-07-2022	Phase Duration (Days): 2	Requirements End Date: 28-07-2022
--	--------------------------	--------------------------------------

2) Acceptance Testing:

Acceptance Start Date: 30-07-2022	Phase Duration (Days): 2	Acceptance End Date: 31-07-2022
--------------------------------------	--------------------------	------------------------------------

3) System Design

Design Start Date: 03-08-2022	Phase Duration (Days): 3	Design End Date: 05-08-2022
----------------------------------	--------------------------	--------------------------------

4) System Testing

Testing Start Date: 06-08-2022	Phase Duration (Days): 3	Testing End Date: 08-08-2022
-----------------------------------	--------------------------	---------------------------------

5) Architecture Design

Architecture Start Date: 10-08-2022	Phase Duration (Days): 4	Architecture End Date: 13-08-2022
--	--------------------------	--------------------------------------

6) Component Testing

Component Start Date: 14-08-2022	Phase Duration (Days): 4	Component End Date: 17-08-2022
-------------------------------------	--------------------------	-----------------------------------

7) Low-Level Design

Low-Level Start Date: 18-08-2022	Phase Duration (Days): 3	Low-Level End Date: 20-08-2022
-------------------------------------	--------------------------	-----------------------------------

8) Unit Testing

Unit Start Date: 21-08-2022	Phase Duration (Days): 4	Unit End Date: 24-08-2022
--------------------------------	--------------------------	------------------------------

Estimate Cost

Total cost for the task: 4014 €

Total duration for the task: 25 days

ASSIGN

Figure 41 Admin Add V-Shaped Task Page

The next module that the admin role can choose is the calendar which is shown in Figure 42.

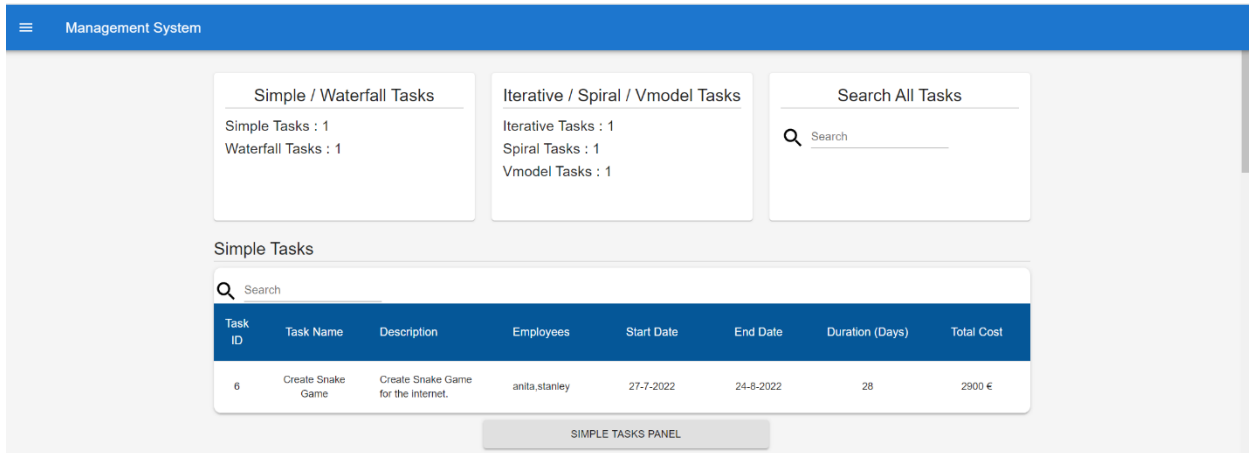


Figure 42 Admin Calendar Page

In the calendar, the admin is informed about the number of tasks that are active at the top of the screen and can search if he is looking for a specific task. At the bottom, each type of task is displayed separately in each table with information about the start and end date of each phase and the cost with the implementation duration. Figure 43 shows the table of Waterfall tasks and Iterative tasks.

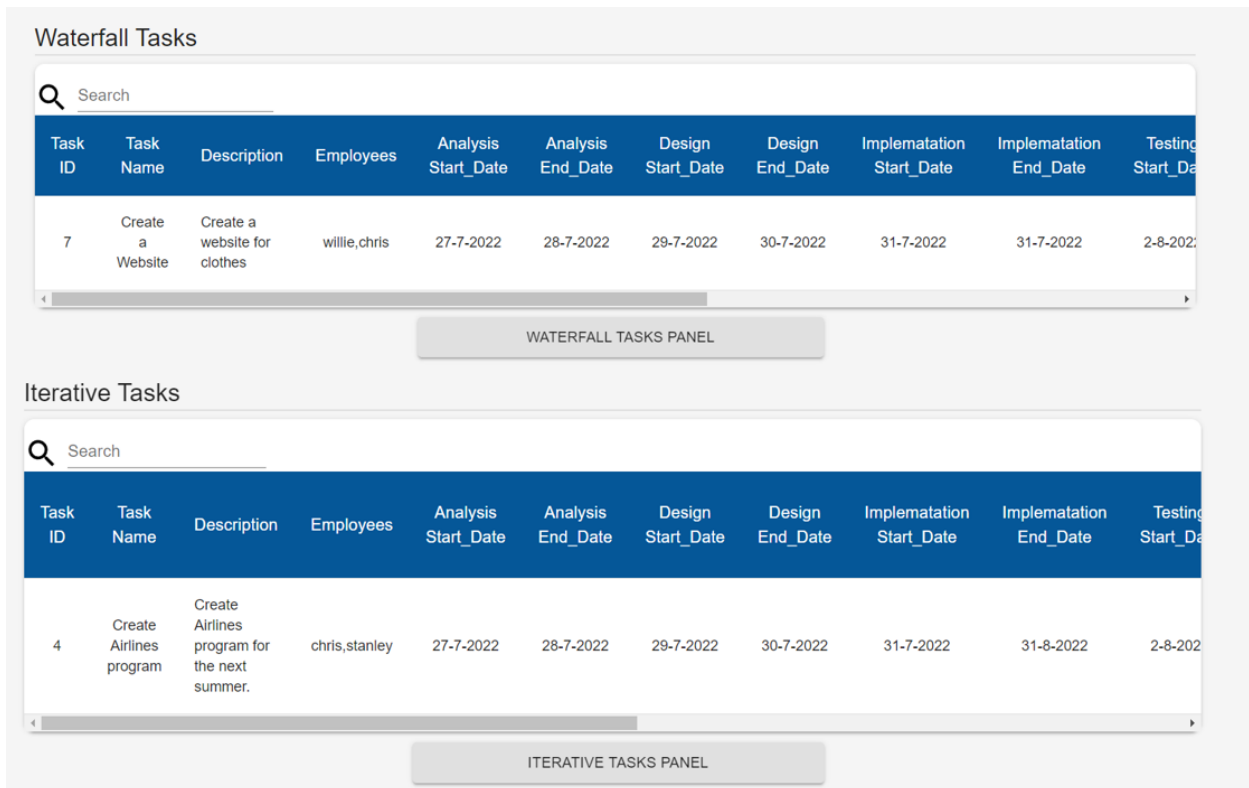


Figure 43 Admin Calendar with Tasks Page

The admin can edit the tasks individually by clicking on the button on each table. When he clicks on the Waterfall tasks, he can edit the data or complete the implementation as shown in figure 44.

WATERFALL TASKS

EDIT THE WATERFALL TASKS

Task ID
7

Task Name
Create a Website

Employees
wille chris

Description
Create a website for clothes

1) Requirement Gathering and analysis

Analysis Start Date: 27-07-2022	Phase Duration (Days): 2	Analysis End Date: 28-07-2022
------------------------------------	-----------------------------	----------------------------------

2) System Design

Design Start Date: 29-07-2022	Phase Duration (Days): 2	Design End Date: 30-07-2022
----------------------------------	-----------------------------	--------------------------------

3) Implementation

Implementation Start Date: 31-07-2022	Phase Duration (Days): 1	Implementation End Date: 31-07-2022
--	-----------------------------	--

4) Integration and Testing

Testing Start Date: 02-08-2022	Phase Duration (Days): 2	Testing End Date: 03-08-2022
-----------------------------------	-----------------------------	---------------------------------

5) Deployment of system

Deployment Start Date: 04-08-2022	Phase Duration (Days): 2	Deployment End Date: 05-08-2022
--------------------------------------	-----------------------------	------------------------------------

6) Maintenance

Maintenance Start Date: 06-08-2022	Phase Duration (Days): 2	Maintenance End Date: 07-08-2022
---------------------------------------	-----------------------------	-------------------------------------

Duration: **11 days**
Cost: **1124 €**

EDIT

COMPLETE Waterfall TASKS

Task ID
7

COMPLETE

CANCEL

Figure 44 Admin Edit Calendar Task Page

When the task can have more than one iteration cycle then an additional iteration cycle can be added by specifying dates and personnel who will or can process the current iteration cycle. The next section that the admin can select is Settings which is shown in figure 45. There is an option to change the password at the top of the screen or delete the account along with all employee accounts.

The screenshot displays two main sections on the Admin Settings page. The first section, titled 'Change Password', contains three input fields: 'Current Password' (with a placeholder 'Current Password *'), 'New Password' (with a placeholder 'Password'), and 'Confirm New Password' (with a placeholder 'Re-enter Password'). A blue 'SAVE' button is positioned below these fields. The second section, titled 'Delete Account', features a warning message: 'Once you delete your account, there is no going back. Please be certain.' Below this, a red text prompt reads 'To confirm this, type "DELETE" below:'. There are two input fields: a grey one with the placeholder 'TYPE DELETE *' and a red 'DELETE' button.

Figure 45 Admin Settings Page

4.1.2 Simple User Role

The simple user has fewer permissions within the application, so the screens should be simpler. When the admin registers the simple user and gives him the details to log into the application then the simple user sees the dashboard which is shown in figure 46.

The screenshot shows the Simple User Dashboard. On the left is a navigation sidebar with links for Dashboard, Financial, Calendar, Settings, and Log Out. The main content area has a blue header 'Management System'. The user profile for 'Anita Sayid' (Developer Employee) is displayed, including fields for Username (anita), Gender (Female), Birthday (12/11/1992), and Nationality (Greek). A green success message states 'Success You are logged in as — User!'. To the right, 'Organization Information' shows Name (Developers Company) and Location (Greece). 'Contact Information' lists Email (anita@gmail.com), Phone (6900000000), and Address (Athens 12). An 'UPDATE ACCOUNT' button is located at the bottom right of the contact information section. A footer note reads 'Copyright © Your Website 2022'.

Figure 46 Simple User Dashboard Page

The options that the simple user has for navigating the menu are five and they are:

1. Dashboard
2. Financial
3. Calendar
4. Settings
5. Log out

When the simple user selects the financial section then he can see the financial data concerning him as shown in figure 47.

EARNINGS FINANCIAL DATA	COMPANY COSTS
Monthly Earnings	Insurance Monthly Costs
1800 €	167 €
Annual Earnings	Equipment Monthly Costs
21600 €	30 €
Start Date	
27/7/2022	

Figure 47 Simple User Financial Page

The simple user is informed about his monthly and annual salary and the start date of work on the left side of the screen. In the right part of the screen, he is informed about the cost of his insurance and the cost of equipment per month. Figure 48 shows the screen when the simple user selects the calendar section.

Simple / Waterfall Tasks

Simple Tasks : 1
Waterfall Tasks : 0

Iterative / Spiral / Vmodel Tasks

Iterative Tasks : 1
Spiral Tasks : 0
Vmodel Tasks : 0

Search All Tasks

Simple Tasks

Task ID	Task Name	Description	Employees	Start Date	End Date	Duration (Days)
6	Create Snake Game	Create Snake Game for the internet.	anita,willie	27-7-2022	25-8-2022	29

Iterative Tasks

Task ID	Task Name	Description	Employees	Analysis Start Date	Analysis End Date	Design Start Date	Design End Date	Implementation Start Date	Implementation End Date	Test Start
4	Create Airlines program	Create Airlines program for the next summer.	anita,penny,willie	27-7-2022	27-7-2022	27-7-2022	27-7-2022	27-7-2022	27-7-2022	27-7-2022

Figure 48 Simple User Calendar Page

It is informed about the tasks assigned to it and can also search for a specific one. The next section that the simple user can select is settings which are shown in figure 49.

Change Password

Current Password

New Password

Confirm New Password

SAVE

Figure 49 Simple User Settings Page

The only setting he is allowed is to change the password. The simple user was much simpler to implement but enough research was done to ensure that he could not affect the workflow of the rest of the team even if there was malicious activity on his part. He is given the necessary permissions to be aware of the tasks assigned to him and his financial details.

5. Conclusion & Future Work

5.1 Conclusion

SDLC models are a very basic tool for the proper creation and modification of a reliable high-quality software system. Many complex systems run the risk of being implemented incorrectly or leaving a schedule when there is no proper forethought and steps in their implementation. The choice of any SDLC model depends largely on the deadline, available developers, and budget. Many companies spend a lot of money to properly implement their systems and manage their staff at the same time. By properly managing the staff and the implementation project, the risk of failure is drastically reduced.

Based on all the above, the idea of this master thesis was based on the aim to make the system as useful as possible to the groups that need it and to be free of charge to anyone who wishes to extend it, use it or study how to implement it. It is not a simple project as it combines many different fields of computer science on a technical and theoretical level to be able to offer the user the right tool he needs. Its strongest feature is the technologies used as they are new technologies with a lot of community support and usefulness to many companies. The project was challenging to implement because it needed all the technologies to be developed in parallel so that they could work together. It has been a source of knowledge because it can give a comprehensive view of the requirements and obstacles that a developer may have even in a large-scale commerce application. In conclusion, the purpose of the application developed in this master thesis is to be a basic tool for someone who needs to manage a team of developers and have a complete image of the tasks he assigns based on the SDLC models that are the state of the art of the whole application. Still to be able to add and delete members from his team whenever he needs and to be able to cost the projects undertaken by the team based on the salaries of the employees to whom it is assigned. Finally, employees will also be able to be informed about the tasks assigned to them and the deadline of each phase separately for each SDLC model.

5.2 Future work

The system has a lot of prospects for development and can be adapted to other systems or expanded on its own in many different directions. During its implementation, the emphasis was put on using technologies that are very famous for this purpose and that in the future a developer can easily extend the system using help from various open-source communities. The future changes and additions that can be made to the system are:

- Support additional SDLC models in addition to the ones it already has. The application supports Waterfall, Iterative, Spiral, and V-Shaped SDLC models but someone can add as many more as desired, provided that one can graphically represent the phases of the new SDLC model and add the functions needed to calculate various data. Among the most famous SDLC methodologies that can be added are Agile and Scrum.
- Extension to HRMS to allow for better staff management. The application has all the essentials for personnel management such as salaries and basic data. It was implemented this way because it was not intended to be considered an HRMS. It can however be extended to fully functional HRMS if additional screens are added that contain much more information for employees such as schedules, uploading files, statistics, and more.
- Adding additional roles to the application. The application supports two roles, admin and simple user. A role could be added that would have the role of a project supervisor assuming a new SDLC metadata requires it. Another role could be an accountant that would oversee the financials of the company.
- Support for additional languages to make it available to a wider range of users and companies.
- Additional functions for the admin to have a more complete image of his team. He will have the ability to statistically view the performance of each of his employees and will be able to view the financial statistics of each employee individually for the long period that he has been contributing to the team.

- Create a mobile or desktop application. With the mobile app, users will be able to enter and view data without having to open the browser. With the desktop application companies could use the application on a closed network locally greatly reducing the risk of malicious activity
- In-app chat support between roles. When Admin will want to inform an about a change or event to a simple user, he can send a message from the application and a notification will be displayed on the dashboard
- Notifications that a deadline for a task is approaching. The user will receive a notification at the dashboard or an email that a deadline is approaching for a task that has been assigned and needs to be completed.

6. References

- [1] Amlani, Radhika D. "Advantages and limitations of different SDLC models." *International Journal of Computer Applications & Information Technology* 1.3 (2012): 6-11.
- [2] Rastogi, Vanshika. "Software development life cycle models-comparison, consequences." *International Journal of Computer Science and Information Technologies* 6.1 (2015): 168-172.
- [3] Shylesh, S. "A study of software development life cycle process models." *National Conference on Reinventing Opportunities in Management, IT, and Social Sciences*. 2017.
- [4] Kute, Seema Suresh, and Surabhi Deependra Thorat. "A Review on Various Software Development Life Cycle (SDLC) Models." *International Journal of Research in Computer and Communication Technology* 3.7 (2014): 778-779.
- [5] Sharma, Manish. "A Survey of project scenario impact in SDLC models selection process." *International Journal of Scientific & Engineering Research* 2.7 (2011): 1-4.
- [6] Alshamrani, Adel, and Abdullah Bahattab. "A comparison between three SDLC models waterfall model, spiral model, and Incremental/Iterative model." *International Journal of Computer Science Issues (IJCSI)* 12.1 (2015): 106.
- [7] Kumar, Madhup. "A Comparative Study of Universally Accepted SDLC Models for Software Development." *vol 4* (2018): 31.
- [8] Balaji, S., and M. Sundararajan Murugaiyan. "Waterfall vs. V-Model vs. Agile: A comparative study on SDLC." *International Journal of Information Technology and Business Management* 2.1 (2012): 26-30.
- [9] Κούκλαρης, Παναγιώτης. *Project management methodology and tools*. MS thesis. Πανεπιστήμιο Πειραιώς, 2019.
- [10] Miljanic, Mirko, and Nikola Zaric. "Review of collaborative software applications and integration with standard collaboration tools." *2020 24th International Conference on Information Technology (IT)*. IEEE, 2020. RAJESH SAHA "Human Resource Management System (HRMS) " *New Horizon College of Engineering, Department of Master of Computer Applications*, 2018-2019

- [11] RAJESH SAHA "Human Resource Management System (HRMS) " New Horizon College of Engineering, Department of Master of Computer Applications, 2018-2019
- [12] Alshamrani, Adel, and Abdullah Bahattab. "A comparison between three SDLC models waterfall model, spiral model, and Incremental/Iterative model." *International Journal of Computer Science Issues (IJCSI)* 12.1 (2015): 106.
- [13] Kute, Seema Suresh, and Surabhi Deependra Thorat. "A review on various software development life cycle (SDLC) models." *International Journal of Research in Computer and Communication Technology* 3.7 (2014): 778-779.
- [14] Rather, Manzoor Ahmad, and Mr Vivek Bhatnagar. "A comparative study of software development life cycle models." *International Journal of Application or Innovation in Engineering & Management (IJAIEEM)* 4.10 (2015): 23-29.
- [15] Jirava, Pavel. "System development life cycle." *Scientific papers of the University of Pardubice. Series D Faculty of Economics and Administration.* 9 (2004) (2004).
- [16] Acharya, Biswamohan, and Prabhat Kumar Sahu. "Software development life cycle models: A review paper." *International Journal of Advanced Research in Engineering and Technology (IJARET)* 11 (2020): 169-176.
- [17] Jindal, Tanu. "Importance of Testing in SDLC." *International Journal of Engineering and Applied Computer Science (IJEACS)* 1.02 (2016): 54-56.
- [18] Iqbal, Syed Zaffar, and Muhammad Idrees. "Z-SDLC model: a new model for software development life cycle (SDLC)." *International Journal of Engineering and Advanced Research Technology (IJEART)* 3.2 (2017): 8.
- [19] Rastogi, Vanshika. "Software development life cycle models-comparison, consequences." *International Journal of Computer Science and Information Technologies* 6.1 (2015): 168-172.
- [20] Tuteja, Maneela, and Gaurav Dubey. "A research study on importance of testing and quality assurance in software development life cycle (SDLC) models." *International Journal of Soft Computing and Engineering (IJSCE)* 2.3 (2012): 251-257.

- [21] Ruparelia, Nayan. "Software Development Lifecycle Models, Nayan B. Ruparelia, Hewlett-Packard Enterprise Services." *ACM SIGSOFT Software Engineering Notes* (2010).
- [22] Amlani, Radhika D. "Advantages and limitations of different SDLC models." *International Journal of Computer Applications & Information Technology* 1.3 (2012): 6-11.
- [23] Dora, Sujit Kumar, and Pushkar Dubey. "Software development life cycle (SDLC) analytical comparison and survey on traditional and agile methodology." *Natl. Mon. Ref. J. Res. Sci. Technol* 2.8 (2013): 22-30.
- [24] Kute, Seema Suresh, and Surabhi Deependra Thorat. "A review on various software development life cycle (SDLC) models." *International Journal of Research in Computer and Communication Technology* 3.7 (2014): 778-779.
- [25] Khurana, Gourav, and Sachin Gupta. "Study & comparison of software development life cycle models." *International Journal of Research in Engineering & Applied Sciences* 2.2 (2012): 1513-1521.
- [26] Saravanan, T., et al. "Comparative Analysis of Software Life Cycle Models." 2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN). IEEE, 2020.
- [27] Obeidat, Jasour A., and Hebah HO Nasereddin. "A new vision for information technology project management through selecting SDLC model." *American Academic & Scholarly Research Journal* 5.4 (2013): 183.
- [28] Mishra, Apoorva, and Deepty Dubey. "A comparative study of different software development life cycle models in different scenarios." *International Journal of Advance research in computer science and management studies* 1.5 (2013).
- [29] Alshamrani, Adel, and Abdullah Bahattab. "A comparison between three SDLC models waterfall model, spiral model, and Incremental/Iterative model." *International Journal of Computer Science Issues (IJCSI)* 12.1 (2015): 106.
- [30] Kute, Seema Suresh, and Surabhi Deependra Thorat. "A review on various software development life cycle (SDLC) models." *International Journal of Research in Computer and Communication Technology* 3.7 (2014): 778-779.

- [31] Salve, S. Madhukar, S. Neha Samreen, and Neha Khatri-Valmik. "A Comparative Study on Software Development Life Cycle Models." *International Research Journal of Engineering and Technology (IRJET)* 5.2 (2018): 696-700.
- [32] Kyeremeh, Kwadwo. "Overview of System Development Life Cycle Models." Available at SSRN 3448536 (2019).
- [33] Shylesh, S. "A study of software development life cycle process models." *National Conference on Reinventing Opportunities in Management, IT, and Social Sciences*. 2017.
- [34] Amlani, Radhika D. "Comparison of different SDLC models." *International Journal of Computer Applications & Information Technology* 2.1 (2013): 1-8.
- [35] Ateeq, S. A. D. A. F., and M. Shuaib. "Comparison of various SDLC models." *Global Journal of Multidisciplinary Studies* 3.11 (2014): 176-181.
- [36] *Jurnal Ilmiah Penelitian dan Penerapan Teknologi Sistem Informasi* Vol. 4 No. 1 August 2019
- [37] Buchori, Achmad, et al. "Mobile augmented reality media design with waterfall model for learning geometry in college." *International Journal of Applied Engineering Research* 12.13 (2017): 3773-3780.
- [38] SIMAANYA, Mweemba. *Employee Management System*. 2015. PhD Thesis.
- [39] Alabdulkareem, Areej Abdulrazq Mohammed, and Sarah Mustafa Eljack. "Human Resources Management system."
- [40] Ibrahim, Rosziati, et al. "Development of staff management system using UML-based object-oriented approach." *Proceedings of the International Conference on Computing Technology and Information Management*. 2014.
- [41] Hause, Matthew. "The SysML modelling language." *Fifteenth European Systems Engineering Conference*. Vol. 9. 2006.
- [42] Dinku, Zerihun. "React. js vs. Next. js." (2022).
- [43] Reddy, Prasad, and K. Siva. "Introduction to Spring Boot." *Beginning Spring Boot 2*. Apress, Berkeley, CA, 2017. 1-20.

- [44] Bannon, Ryan, et al. "Mysql conceptual architecture." Technical report, University of Waterloo (2002).
- [45] Dorette Jacob, Jennifer. "Comparing Agile XP and Waterfall software development processes in two start-up companies." (2011).
- [46] Balmelli, Laurent. "An overview of the systems modeling language for products and systems development." *Journal of Object Technology* 6.6 (2007): 149-177.