**Employees management system**

**Students**

Balkis Ibrahim 260092

Dziugas Austys 280144

Przemyslaw Regulski 280196

Ronald Johnson 279987

**Supervisor**

Troels Mortensen

**ICT ENGINEERING**

**SECOND SEMESTER 2019**

**Table of content**

[1 Introduction 1](#_Toc16797618)

[2 User stories and requirements 2](#_Toc16797619)

[2.1 User stories 2](#_Toc16797620)

[2.2 Functional requirements 3](#_Toc16797621)

[2.3 Non-functional requirements 3](#_Toc16797622)

[3 Analysis 4](#_Toc16797623)

[3.1 Scenarios 4](#_Toc16797624)

[3.2 use case diagram 4](#_Toc16797625)

[3.3 use case description 5](#_Toc16797626)

[3.4 Activity diagram 7](#_Toc16797627)

[3.5 Conceptual diagram 8](#_Toc16797628)

[3.6 Database 10](#_Toc16797629)

[4 Design 10](#_Toc16797630)

[4.1 sequence diagram 10](#_Toc16797631)

[4.2 class diagram and design pattern 10](#_Toc16797632)

[4.2.1 MVC Design pattern 11](#_Toc16797633)

[4.2.2 DAO Design pattern 12](#_Toc16797634)

[4.3 GUI 12](#_Toc16797635)

[4.4 Client/Server (RMI) 14](#_Toc16797636)

[5 Implementation 15](#_Toc16797637)

[6 Testing 17](#_Toc16797638)

[7 Conclusion 19](#_Toc16797639)

[8 References 19](#_Toc16797640)

Appendices

# Introduction

# User stories and requirements

## User stories

In this subchapter the user stories from the customer will be presented since the requirements for this report will be made based on them.

1. As an admin, I want to add employees to the system so that all the employees will have access to the system.
2. As an admin, I want to delete employees from the system so that all employees can no longer access the system.
3. As an admin, I want to edit employee’s information in the system, so that all employee’s data can be modified.
4. As an admin, I want to be able to assign shifts to employees, so that employees can view their work plan.
5. As an admin, I edit and remove shifts from employees work plan, so that employees can view their work plan.
6. As an admin, I want to be able to view employee’s data, so that I have access to pertinent information regarding employees.
7. As a user, I want to be able to view my work schedule so that the schedule can be adhered to.
8. As a user, I want to be able to modify my data, so that I can update my data with any future changes.
9. As a user, I want to be able to specify whether I want to work or not on a specific date so that admins can assign my shifts accordingly.
10. As a user, I want to able to denote my time of arrival and departure from work, so that my working hours can be recorded.
11. As a user, I want to be able to check my work-related statistics, so that I can calculate my income.

## Functional requirements

Based on the information above the following requirements have been created.

1. The system must be able to allow the admins to add employees.
2. The system must be able to allow the admins to remove employees.
3. The system must allow admins to modify employee’s data.
4. The system must be able to allow the admins to assign shifts for employees.
5. The system must be able to show the work plan for the employees.
6. The system must allow employees to modify their data.
7. The system must allow the admin to view employee’s data.
8. The system must allow the employees to specify the days they want to work.
9. The system must allow employees to view details regarding specific shift.
10. The system must allow employees to view their work statistics
11. The system must allow the employees to denote the time of arrival and departure from work.

## Non-functional requirements

1. The System must follow the Client/Server architecture (RMI).

2. The system must be implemented in Java.

3. The usability of the system must be tested by end-users.

4. The system must store information in a database.

# Analysis

## Scenarios

## use case diagram

A close up of a map

Description automatically generated

Figure 1use case diagram

The use cases methodology has been used to classify and organize the system requirements. As it is shown in the diagram there are two different actors each has it is own set of actions that needs to be performed as a part of their daily activities.

## use case description

Based on the use case diagram shown above, several use case descriptions were made for this subchapter. Only one-use case description will be shown as an example while the rest can be seen in Appendix 2.

The use case description is a collection of values, preconditions, postconditions and base sequences which form a detailed view of the actor’s actions while interacting with the system.

A screenshot of a cell phone

Description automatically generated

Figure 2use case description for adding employee

The diagram above shows, what steps the admin will take for the system to successfully complete the requested action and what steps does the system make.

The precondition of a use case means, that some steps must be achieved for the use case to be able to reach the end of the branch sequence. In this case, the admin must be logged in to continue with the use case. The actor (admin) will fill out the form with the employee info. In case one of the existing employees has the same email address the system will reject the newly added employee and display an error, otherwise, the system will save the added employee in the database.

## Activity diagram

The activity diagram was created during the inception face to get a clear idea of the certain actions that need to be executed by the actor to reach his goal.A close up of a logo

Description automatically generated

Figure 3Add Employee activity diagram

One of the Admin task is to add an Employee the activity diagram shows the steps needs to be taken to save the employee in the database.

## Conceptual diagram

Conceptual Diagram demonstrates how all of the classes in their packages interact with each other. This is important since it provides a graphical representation on how the system will be created and how each component will interact with another to make it functional.

A close up of a piece of paper

Description automatically generated

Figure 4conceptual diagram

Following the SOLID principles, the conceptual diagram is divided in three main packages Domain, Controller and View. In addition, all main packages have their sub packages. Domain carries Mediator and Model.

In Model Employees, shifts.., and most of the classes with their lists. In Mediator package there are model managers and databases for shifts, attendance and employees…. Controller and View have Add, Remove and Edit for Shifts and employee.

## Database

# Design

## sequence diagram

## class diagram and design pattern

### MVC Design pattern

### DAO Design pattern

## GUI

## Client/Server (RMI)

# Implementation

# Testing

# Conclusion

The project “Employee management system” has been completed, but due to lack of time, there are few delimitations.

It is required by the university that the project should follow the Client-Server and it also should have multiple design patterns. To follow these requirements, many user stories where drawn and it took a long time to make a correct class diagram that would be relevant to the user’s needs. Designing and Coding was challenging, and the reason was the summer holiday. It was not easy to get the whole team members focusing on the project

In conclusion, we did our best and we tried as much as it is possible to deliver a good product that can be possible for future development.

# References

Ken Schwaber, Jeff Sutherland, 2011; The Scrum Guide, [Last accessed 10/04/2018 ] via link: https://studienet.via.dk/Class/IT-SWE1XS18/Session%20Material/Scrum\_Guide.pdf

Project report, 2017 (Appendix 3) VIA Engineering Guidelines [Last accessed 10/04/2018] via link: https://studienet.via.dk/projects/Engineering\_\_project\_methodology/General/Guidelines /2017%20Project%20Report%20(Appendix%203)%20%20VIA%20Engineering%20Guidelines.pdf Project description, 2017 (Appendix 1) VIA Engineering Guidelines [Last accessed 27/02/2018] via link: https://studienet.via.dk/projects/Engineering\_\_project\_methodology/General/Guidelines /2017%20Project%20Description%20(Appendix%201)%20%20VIA%20Engineering%20Guidelines.pdf

Process report,2017 (Appendix 2) VIA Engineering Guidelines, [Last accessed 05/04/2018] via link: https://studienet.via.dk/projects/Engineering\_\_project\_methodology/General/Guidelines /2017%20Process%20Report%20(Appendix%202)%20%20VIA%20Engineering%20Guidelines.pdf

**Appendices**