|  |
| --- |
| **Eurofins Work-planner Project Report** |

**Florin Bordei, 280593**

**Jimmi C.B, 280091**

**Kenneth Jensen, 280269**

**Teakosheen Joulak, 280021**

**Supervisor: Mona Andersen, Michael Viuff**

**VIA University College**

**[Logo of companies included]**

**[Number of characters]**

**ICT Engineering**

**1st Semester**

**21/11/2018**

**Table of content**

[Abstract iv](#_Toc501021071)

[1 Introduction 1](#_Toc501021072)

[2 Requirements 2](#_Toc501021073)

[2.1 Functional Requirements 2](#_Toc501021074)

[2.2 Non-Functional Requirements 2](#_Toc501021075)

[3 Analysis 3](#_Toc501021076)

[4 Design 4](#_Toc501021077)

[5 Implementation 5](#_Toc501021078)

[6 Test 6](#_Toc501021079)

[6.1 Test Specifications 6](#_Toc501021080)

[7 Results and Discussion 7](#_Toc501021081)

[8 Conclusions 8](#_Toc501021082)

[9 Project future 9](#_Toc501021083)

[10 Sources of information 10](#_Toc501021084)

[11 Appendices 1](#_Toc501021085)

**List of figures and tables**

Optional

# Abstract

*An abstract is a shortened version of the report and should contain all information necessary for the reader to determine:*

1. *What are the aim and objectives of the project*
2. *What are the main technical choices*
3. *What are the results*

*Frequently, readers of a report will only read the abstract, choosing to read at length those reports that are most interesting to them. For this reason, and because abstracts are frequently made available to engineers by various computer abstracting services, this section should be written carefully and succinctly to have the greatest impact in as few words as possible.*

*Although it appears as the first section in a paper, most report writers write the abstract section last.*

Cf. (Dawson 2009, p.195).

# Introduction

Eurofins is an international network of laboratories, centered in the European countries. Their Danish office is located in Vejen, and its focus is to test food, dairy products and feed. (Eurofins Scientific, 2018)

Today, they have multiple excel-sheets for keeping track of their calendar, work assignments and vacations. The excel-sheets are updated manually for each employee by their team leader. In that sense, the team-leader organizes the work-load of each employee for the upcoming weeks, using a separate work-sheet, to have the right amount of people on each task. They keep another document to keep track of preferences, for example allergies, bad shoulder and so on. The team leaders need to keep track of whether an employee can perform a certain analysis (if they have been trained for it). (Eurofins Steins, n.d.)

The problem with the manual input of all the calendars where the team leaders juggle between 3 worksheets, is that they often make mistakes and the process consumes a lot of time. The mis-assignment of a resource will influence other projects, because of the inter-connectivity of projects and departments in modern organizations. Therefore, it is vital that team leaders establish an appropriate sequence and configuration of work, which is difficult to do manually. (Bendoly, et al., 2010)

Eurofins needs an application that can store a staff time template, a training sheet and a work plan in one place. They would like to be able to edit these, even when others are viewing them, and they want only team-leaders to be able to edit the files. The different functions need to be flexible to allow for comments, adding or removing different analyses, hiring new employees and hiding or removing old ones. They would like to be able to store the data for at least a year and they want to be able to view the plan without installing the application, for example online. (Eurofins Steins, n.d.)

The system will not contain server/client infrastructure, as this has not been taught as part of the curriculum. Because of this, the system will not be able to notify users of changes that has occurred, while they are away from the program. Data will not be stored for long periods of time, as that would require a dedicated location for the save files, which we do not have access to. The system will have a limited availability as it will only be able to run on a computer using java.

Following this section will go into detail about our Project Description, explain in detail the requirements for the system, walk through the steps of our analysis and then into the design of the program. Finally, the report will document how the system has been implemented, including the testing phases. Through the testing phases, the requirements have been tested individually, to ensure the functionality of the system.

# Requirements

We have two actors: an employee and a team leader.

Employees will only use the program for viewing their own or others’ schedule. As a part of that, they can search for a date, a fellow employee or team leader and types of analysis performed.

Eurofins needs one program for work plan, employee management, training and preferences. The program has two types of users: The team-leaders and employees. Employees can view the work plan and they can also search for specific date, schedule or analysis. While the team-leaders are able to add, change or delete a work plan or comment for a specific employee in a specific date. They can also add, hide or remove an employee. Moreover, they can accept vacations and make or accept a request form external team-leaders for an employee to do an assignment for the other team.

## Functional Requirements

1. The system must be able to track the current date
2. The system must be able to extrapolate current month and/or week from current date
3. The system must store the employees name, employee code (1), training info (2) and test preferences (3)
4. The system must be able to differentiate between employee and team leader (4), as only the team leader can make changes
5. Users must be able to search for specific schedules or type of analysis
6. The search function must be able to search using keywords, such as names, codes, analysis type and date
7. Team leaders must be able to mark days of a specific employee as vacation days
8. The system must be able to differentiate the vacation marks (see requirement 8) into 2 categories: Accepted vacations, and vacations pending acceptation
9. The team leader must be able to comment on specific dates of an employee
10. The system must be able to store a worksheet which records how many employees are needed, for any specific test in a specific week
11. Team leaders must be able to request for an employee on another team to be assigned on a task on another team
12. The system must be able to accept a request for an employee to do an assignment at another team, from an external team leader and record the employee’s temporary assignment to the other team
13. The system must be able to prevent illegal inputs, like a character in an age
14. The team leader must be able to add, change or delete a work-schedule on a specific date of a specific employee
15. The team leader must be able to add, hide or remove employees from the registry
16. The team leader must be able to put multiple analyses on a single day of a specific employee
17. The schedule must be available without installing an application
18. The system must keep track of employees tasked with the different assignments
19. The system should alert the team leader if the number of employees on a task is less than the minimum required (see requirement 10)

## Non-Functional Requirements

1. The system must be developed in Java
2. Scheduled vacations (see functional Req. 8) must colour accepted vacation red and vacations pending acceptation yellow
3. The stored training info (2) should be coloured:
   * Purple for approved employee
   * Red for employee needing retraining
   * Yellow for employee under training
4. The search function must return results within 2 second 95% of the time

# Analysis

The purpose of the analysis section is to outline an understanding of the problem domain and specifically WHAT the stakeholders want. Here, you elaborate on your background description.

You identify objects in the problem domain that will be involved in the solution and how these objects cooperate. The result of this analysis is a Domain Model (Larman 2004, chap.9) and other relevant diagrams.

Use the UML standard for all diagrams where relevant.

Note: Remember that all implementation dependent objects are not part of the domain model only conceptual classes related to the requirements and the domain.

# Design

The purpose of the design section is to outline HOW the system is structured; i.e. to transform the artefacts of the analysis into a model that can be implemented. The design section is relevant for the programmer, whereas the analysis is relevant for the stakeholder.

Elements that may be relevant in this section:

* Architecture: Find architecture patterns here (Leszek Maciaszek 2004, chap.9).
* Technologies: Describe technologies used, also alternative technologies. Argue for choice of technology according to the project aim.
* Design Patterns: Describe which design patterns (GoF (Gamma et al. 2002) etc.) you are using and why.
* Class Diagrams
* Interaction Diagrams
* UI design choices
* Data models, persistence, etc.

You must explain all diagrams in the report. These diagrams including descriptions are the blueprints for the implementation.

Hint: One way to figure out which objects/classes are needed in the design is to apply the General Responsibility Assignment Software Patterns/principles (GRASP) (Larman 2004, chap.17).

Hint: Consider how to design your system to make it testable.

# Implementation

The purpose of the implementation section is to explain interesting code snippets. An idea is to explain the complete path through your system from UI to database etc.

Remember that your implementation must be consistent with your design (Larman 2004, chap.20).

Which standard libraries are used? How are design patterns implemented, etc.

Hint: Implement your code in a testable manner.

# Test

The purpose of the test section is to document the result of your testing; to verify if the content of the requirements section has been fulfilled. How is the system tested, which strategy has been used; e.g. White Box (Unit Test), Black Box, etc.

## Test Specifications

For functional requirements, test specifications must be listed. These test specifications can be described as soon as the functional requirements have been completed (Use Cases including descriptions).

IEEE can be used as a template for test specification (IEEE Computer Society 2008). VIA Library can give you access to this standard.

# Results and Discussion

The purpose of the results and discussion section is to present the outcome and achieved results of the project.

# Conclusions

The purpose of the conclusion section is to compile the results from each section in the report. What is the conclusion? Did the project fulfil the requirements? Etc.

You can only comment on report contents, no new topics or content can be introduced in this section.

# Project future

Reflect on your project from a technical viewpoint and describe what you would change if you could.

Suggest how the project could be improved or made ready for production. Discuss scalability, suggest possible spin offs, what is needed, missing, etc.?

# Sources of information

**Note: Use the standard reference method: Harvard Anglia. A very good reference tool is Mendeley** (Mendeley.com 2016), **ask VIA Library if you need help.**

Banger, D., 2014. A Basic Non-Functional Requirements Checklist « Thoughts from the Systems front line.... Available at: https://dalbanger.wordpress.com/2014/01/08/a-basic-non-functional-requirements-checklist/ [Accessed January 31, 2017].

Business Analyst Learnings, 2013. MoSCoW : Requirements Prioritization Technique — Business Analyst Learnings. , pp.1–5. Available at: https://businessanalystlearnings.com/ba-techniques/2013/3/5/moscow-technique-requirements-prioritization [Accessed January 31, 2017].

Dawson, C.W., 2009. *Projects in Computing and Information Systems*, Available at: http://www.sentimentaltoday.net/National\_Academy\_Press/0321263553.Addison.Wesley.Publishing.Company.Projects.in.Computing.and.Information.Systems.A.Students.Guide.Jun.2005.pdf.

Gamma, E. et al., 2002. *Design Patterns – Elements of Reusable Object-Oriented Software*, Available at: http://books.google.com/books?id=JPOaP7cyk6wC&pg=PA78&dq=intitle:Design+Patterns+Elements+of+Reusable+Object+Oriented+Software&hl=&cd=3&source=gbs\_api%5Cnpapers2://publication/uuid/944613AA-7124-44A4-B86F-C7B2123344F3.

IEEE Computer Society, 2008. *IEEE Std 829-2008, IEEE Standard for Software and System Test Documentation*,

Larman, C., 2004. *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development*,

Mendeley.com, 2016. Homepage | Mendeley. Available at: https://www.mendeley.com/ [Accessed February 2, 2017].

YourCoach, S.M.A.R.T. goal setting | SMART | Coaching tools | YourCoach Gent. Available at: http://www.yourcoach.be/en/coaching-tools/smart-goal-setting.php [Accessed August 19, 2017].

# Appendices

The purpose of your appendices is to provide extra information to the expert reader. List the appendices in order of mention.

Examples of appendices

* Project Description
* User Guide
* Source code – source documentation
* Diagrams
* Data sheets
* Etc.

**Appendix A Project Description**

Insert the original Project Description here