Eurofins planning tool

1. Semester project

Bors Dementie, 279948

Lucas Kaas Møller, 279967

Justinas Jancys, 280151

Pavliuc Bogdan, 280156

Sova Nicoleta, 267069

Supervisor: Mona Wendel Andersen

Michael Viuff

VIA University College Horsens

[Number of characters]

Software engineering

1. semester

2018

Table of content

Abstract 4

1 Introduction 5

2 Requirements 7

2.1 Functional Requirements 7

2.2 Non-Functional Requirements 7

3 Analysis 8

4 Design 12

5 Implementation 15

6 Test 19

6.1 Test Specifications 22

7 Results and Discussion 23

8 Conclusions 23

9 Project future 23

10 Sources of information 24

11 Appendices 25

List of illustrations, figures and tables

Illustration 1 – Current scheduling tool (Viuff, 2018) 5
Illustration 2 – GUI of the “Manage employees” page 14
Illustration 3 – Login interface 15
Illustration 4 – Login code 15
Illustration 5 – Employee schedule view 16
Illustration 6 – Team leader schedule view 17
Illustration 7 – Shift code 18
Illustration 8 – Login test input 19
Illustration 9 – Login test output 19
Illustration 10 – Date test input 19
Illustration 11 – Date test output 19
Illustration 12 – Vacation test input 19
Illustration 13 – Vacation test output 19
Illustration 14 – Analysis and AnalysisList test input 19
Illustration 15 – Analysis and AnalysisList test output 19
Illustration 16 – EmployeeFile and EmployeeList test input 1 20
Illustration 17 – EmployeeFile and EmployeeList test input 2 20
Illustration 18 – EmployeeFile and EmployeeList test output 20
Illustration 19 – Shift test input 21
Illustration 20 – Shift test output 1 21
Illustration 21 – Shift test output 2 21

Figure 1 – Use case diagram 8
Figure 2 – Activity diagram (Manage employees) 11
Figure 3 – Relations only 12
Figure 4 – “Shift” part class diagram 13
Figure 5 – “Work schedule” part class diagram 13
Figure 6 – “Entry point” part class diagram 14

Table 1 – Use case description 9
Table 2 – Requirements covering use cases 10
Table 3 – Use cases covering requirements 10
Table 4 – Test results 22

Abstract

The aim of the project is to create a program which eases the scheduling for Eurofins Chemistry Lab in Vejen. Currently, Excel is being used to create the work schedule, however, this is not a time efficient method. Thus, the program needs to analyze the requirements for each shift, which is dependent on the type of analysis and employees. Furthermore, the program should have a user-friendly interface, where the scheduling information is easy to locate.

The final implementation of the program did not fulfill all requirements, however, none of these requirements were crucial for the program's functionality. The implemented classes were tested and confirmed to work as intended.

1 Introduction

Eurofins is one of the world leaders within testing of food, environment and pharmaceutical products. With more than 650 laboratories and over 45,000 employees across 45 countries, Eurofins holds a staggering portfolio with over 150,000 different analytical methods (Eurofins, 2018). These numbers require an organization where all the organization is running smoothly and as efficient as possible and with that number of employees, scheduling is a large part of this.

Eurofins Steins Laboratorium A/S is analyzing dairy and feed, where upwards to 40,000 samples are analyzed daily. One of the departments is the Chemistry department, where around 60 employees are employed (Viuff, 2018). The department's team leader, Agnete Viuff, has requested a new and easier way to do the scheduling of the employees.

Currently, Viuff is using an Excel spreadsheet as a tool for creating the schedule, as it is a tool that is easy to use and understand, yet it lacks some functionalities making it efficient and timesaving. The spreadsheet consists of four different sheets; the work plan, staff time, training overview and preferences. When making the schedule, Viuff shifts between the sheets, remembering the information from one sheet and then manually inputs it in another sheet. The illustration below displays the four sheets used to scheduling a week. First sheet two is accessed to see the number of employees is required for an analysis, depending on the day of the week, and type of week. After this, sheet three is accessed to find the required personal that are trained within the analysis as well as checking to see if an employee has a preference regarding this analysis in sheet four. Finally, with all of this information, the shifts can be added in sheet one.

Sheet 1: Work plan - Major working document

2018		37	Dep meeting			S/L
Initials	Date	10-09-18	11-09-18	12-09-18	13-09-18	14-09-18
BB2	Bob	Fat	Fat	Fat	Fat	Fat
SN6	Susan	Starch training	Enzymes	reg day off	reg day off	Free
KN2	Ken	Fiber/ Sugar	Fiber/ Sugar	Fiber/ Sugar	Fiber/ Sugar	Fiber/ Sugar
BE7	Barbie	Protein	Protein (Off at 13.15)	Protein	Protein	Pending vacation
SE4	Sussie	Starch training	Cattle	Starch	Cattle	Protein
LO3	Leo	Milk	Milk	Pig	Pig	Pig
AE5	Alice	Fat	Fat	Fat	Fat	Fat
CL9	Carol	Vacation	Nitrogen	Nitrogen	Nitrogen	Chicken

Sheet 2: Staff time overview - Works as template for the work plan

			Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
Matrix	Week	Analysis	No. Persons	No. Persons	No. Persons	No. Persons	No. Persons	No. Persons
Food	L	Fat	2	1.5	1.5	2	2	0.5
Food/Feed	L	Protein	2	2	2	2	2	0.5
etc	L							
Food	S	Fat	3	2.5	2.5	2	2	0.5
Food/Feed	S	Protein	2	2	2	2	2	0.5
etc	S							
Feed	All	Fiber	1	1	1	1	1	0.5
Feed	All	Sugar	1	1	1	1	1	0.5
etc	All							

Sheet 3: Training overview - Look up to see who is trained for what

Name	Fat	Protein	Enzymes	etc
Bob	Trained	Need retraining		
Susan		Trained		
Ken	Trained		Under training	
Etc				

Sheet 4: Preferences from annual performance review (MUS)

Sussie	Wants to avoid Fat
Barbie	Wants to be trained in enzymes
Ken	Prefers to work on Fiber

Illustration 1 – Current scheduling tool (Viuff, 2018)

For employees though, the spreadsheet is easy to understand, a requirement that must carry over to the program. The purpose of the program is not only to ease the work planning for the team leaders, but also provide employees with an easy-to-understand presentation of their schedules. Thus, the object of the program is to provide the team leaders with efficient and easy way of managing employees, analyses work schedules etc. as well as a platform for the employees to access their schedules.

To ensure that the program has all the functionalities required to create a work schedule for the employees, all the requirements need has been listed, as a way to check if the project has been fulfilled.

2 Requirements

The following requirements to the program have been stated based on the presentation and interview with Eurofins. The requirements ensure that the needs and requests from Eurofins will be implanted into the final program, thus, fulfilling the demands for the scheduling program.

2.1 Functional Requirements

1. Every user needs to login before use.
2. The schedule for an employee can be created/edited/deleted only by a team leader.
3. Notes can be added by team leaders to shifts.
4. Team leaders can remove old employees.
5. Team leaders can add new employees.
6. The work schedule can be color coded by the team leader depending on the type of analysis or work.
7. Employees should be able to search for individual times in the schedule through a search bar.
8. The work schedule should have a button to go to current date.
9. Every user can request a vacation.
10. Team leaders need to verify (accept/reject) vacations.
11. Every user can view individual and team work schedule.
12. Vacation verification should pop up for the team leader.
13. There needs to be a legend for the colour codes.
14. When the team leader is creating a work schedule, individual employee preferences should be shown.
15. If an employee only works 3 days a week, then the employee shouldn't be to be booked the other days.

2.2 Non-Functional Requirements

1. The work schedule needs to be able to be filled at least 4 weeks in advance.
2. The system needs to save past work schedules, so that they could be reviewed if needed.
3. Being re-trained in different analyses take different times, therefore, this should be managed by the team leader.

3 Analysis

Problem domain:

The intentions of the program for Eurofins is to ease the way of scheduling shifts. Currently, excel is being used to create the work schedule as it has some features which can be used to ease the understanding of the schedule. However, using Excel is time-consuming for the team leader, as everything must be manually inputted. Furthermore, the method that Eurofins currently uses has no way of ensuring that every shift is filled out and that preferences of employees are taken into consideration. All of this, the team leader must account for when creating the work schedule, resulting in even more time being spend by the team leader creating the work schedule.

By implementing a program that ensures that all the requirements above are met when creating a shift, the time and effort needed by a team leader to generate a work schedule will be reduced. Additionally, it is crucial that the visual display of the schedule is easy for employees to understand and navigate through, ensuring that their time required to read their schedule is kept at a minimum.

Use case diagram

The Use case diagram shown below displays the functions an employee or a team leader can access through the program. An employee is restricted to only viewing the schedule and only having the option of requesting vacation. As a team leader, all the functionalities of the program are accessible. Through the program, a team leader can manage employees and analyses, creating shifts and approve vacations.

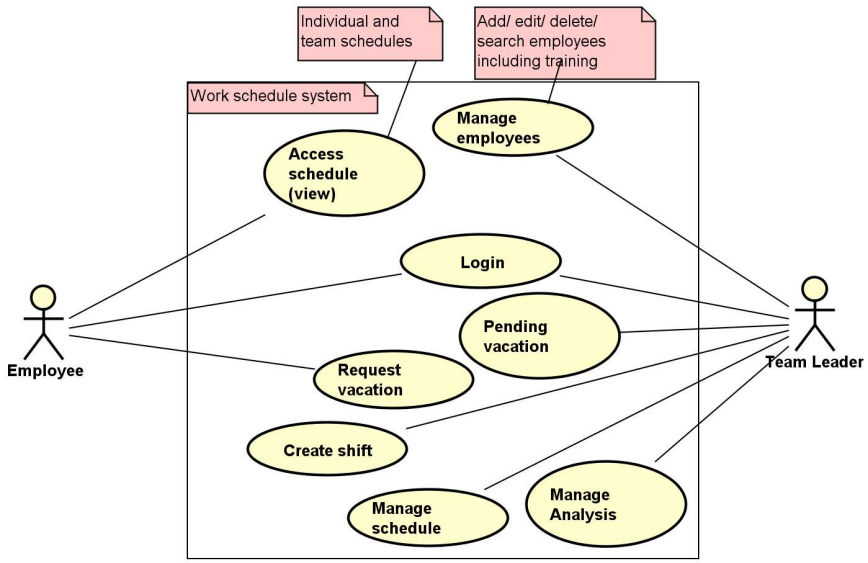


Figure 1 – Use case diagram

Use case descriptions

The description of how a team leader can manage employees is shown in the table below. The rest of the use cases can be found in *Appendix B*.

Item	Value
Use case	Manage employees
Summary	The team leader can add, edit and delete employees as well as search for employee data
Actor	Team leader
Precondition	The team leader must be logged in and accessed the "Manage employees" page.
Postcondition	Employee data has been changed
Base sequence	*To add an employee:* 1. Fill out the credentials (Name, surname, email, phone number, preferences and areas trained in) in the corresponding fields 2. Click "add" 3. The employee is added to the list of employees *To edit an employee:* 1. Use the search bar to locate the employee 2. Click on the "edit" button next to the employee 3. The fields will become editable and instead of the "edit" button, a "save" and a "cancel" button will be shown. 4. Change the desired credentials of the employee 5. Click "save" 6. The new data of the employee will be stored and shown on the list. *To delete an employee:* 1. Use the search bar to locate the employee 2. Click "Delete" next to the employee 3. The "delete" button will change to a "yes" and a "no" button. 4. Click "yes" 5. The employee will be removed from the list
Branch sequence	None
Exception sequence	If all the credentials are not filled out, an error will show (Step for add 3 and step 6 for edit will not be executed.)
Sub use case	None
Note	

Table 1 – Use case description

Link between requirements and use cases

To ensure that all requirements will be fulfilled, the tables underneath links the use cases and requirements. Table 2 lists all the requirements covering the individual use cases and table 3 is listing all use cases covering the individual requirements.

Use case	***Covered requirements***
Login	1
Access schedule	7, 8, 11, 13
Request vacation	9
Manage employees	2, 4, 5, 14, 15
Manage analyses	2, 6
Pending vacations	2, 10, 12
Create shifts	2, 14, 15
Manage schedule	2, 3

Table 2 – Requirements covering use cases

Requirement	***Use case(s)***
1	Login
2	Create shift
3	Manage schedule
4	Manage employees
5	Manage employees
6	Manage analysis
7	Access schedule
8	Access schedule
9	Request vacation
10	Pending vacation
11	Access schedule
12	Pending vacation
13	Access schedule
14	Manage employees; create shift
15	Manage employees; create shift

Table 3 – Use cases covering requirements

Activity diagram

The diagram below shows the process of the “Manage employees” use case. The remaining Activity diagrams can be found in *Appendix C*.

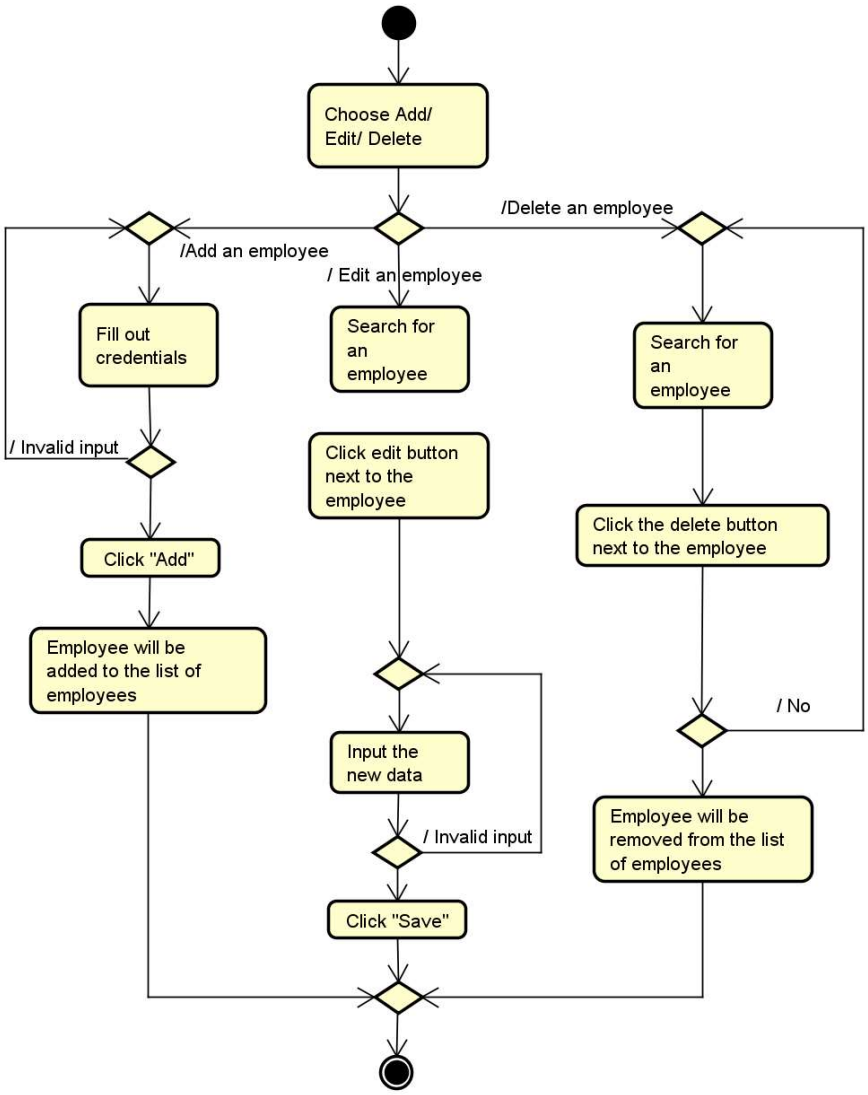


Figure 2 – Activity diagram (Manage employees)

4 Design

The class diagrams below explain the structure of the program. The Class Diagram has been split up to ease the understanding of it. The complete Class Diagram can be found in *Appendix D*.

"Relations only" – class diagram

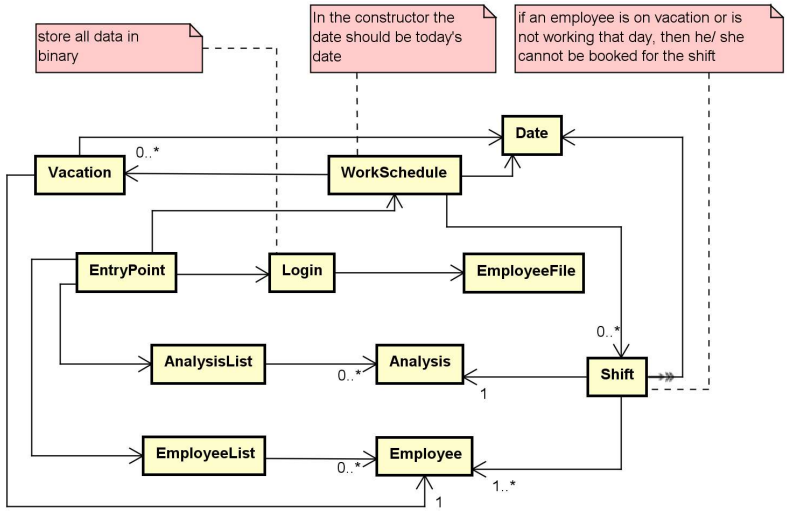


Figure 3 – Relations only

Shown above is the relations between all classes within the program. The individual classes will be explained in further detail in the following figures.

"Shift" part – class diagram

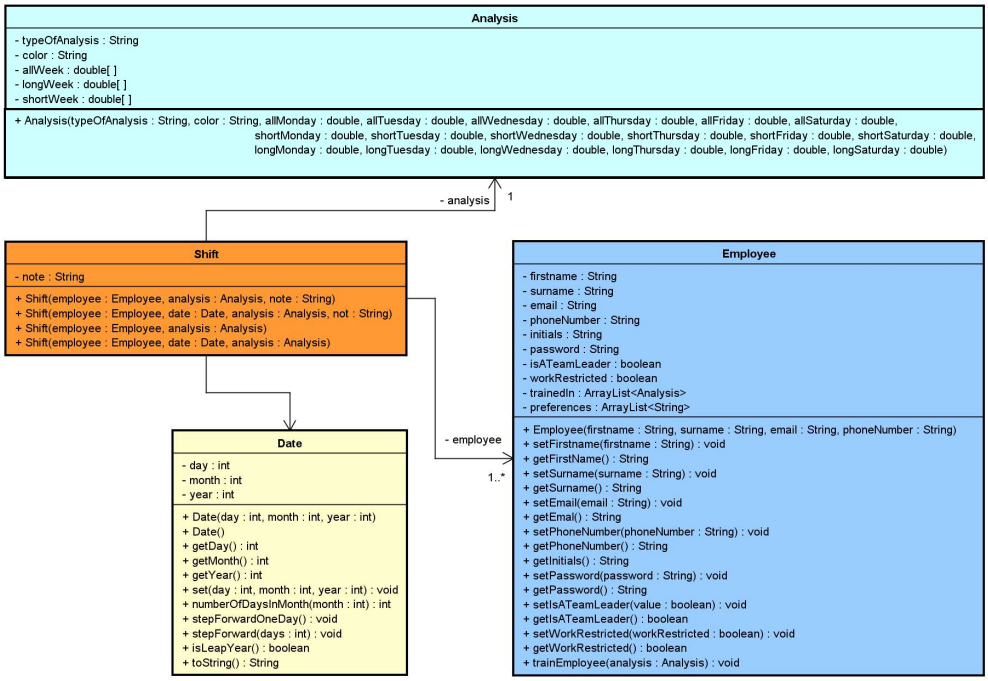


Figure 4 – "Shift" part class diagram

A *Shift* is made of an *analysis*, set number of *Employee*(s) and a date. When creating a *Shift*, an *Analysis* and a *Date* is chosen. Within the *Analysis* class the number of employees required for a large and short week as well as the general number for every week is specified. By having this specified, the program will state the number of *Employees* need for that *Analysis* on the given *Date*.

"Work schedule" part – class diagram

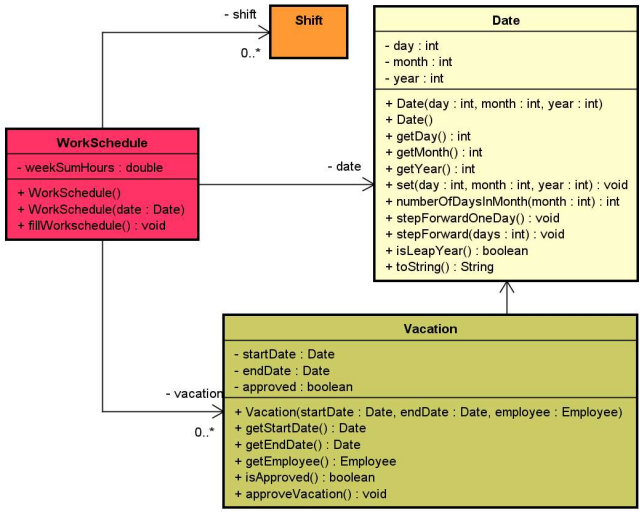


Figure 5 – "Work schedule" part class diagram

The *WorkSchedule* class is where all the *shifts* and *vacations* are stored. From here they will be displayed for the users of the program.

"EntryPoint" part – class diagram

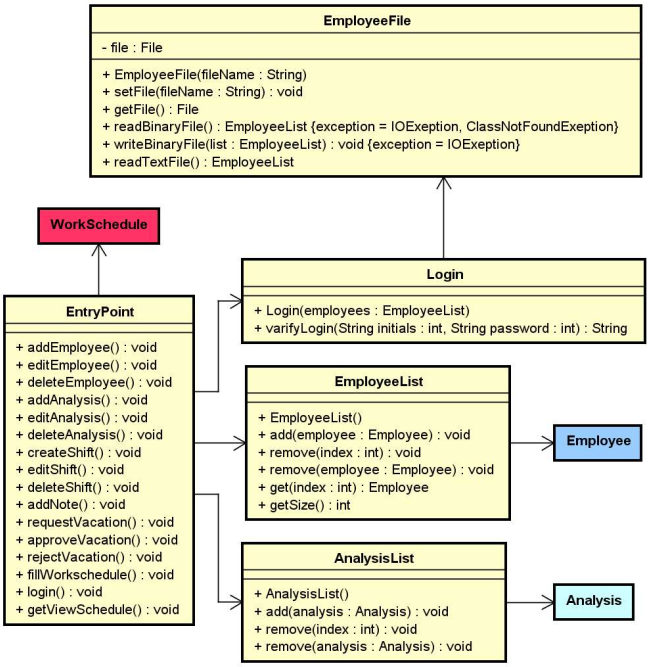


Figure 6 – "Entry point" part class diagram

The *EntryPoint* class is the entry point of the system. Here all the methods are available for the controller.

Graphical user interface

A part of the problem domain was for the program to be easily understandable for the user, thus, the GUI must be designed and implemented in such way. The image below shows the screen where a team leader can manage employees. It is designed in a simple and user-friendly way, ensuring that our requirement of it having a simple interface will be met. In *Appendix E* a complete User guide to the system can be found.

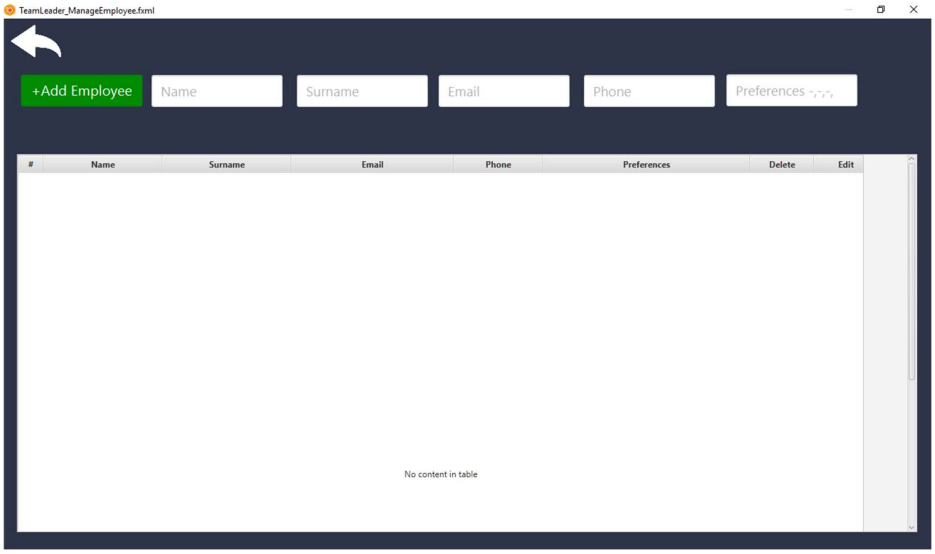


Illustration 2 – GUI of the "Manage employees" page

5 Implementation

Login

The login class is the first step to access our program.

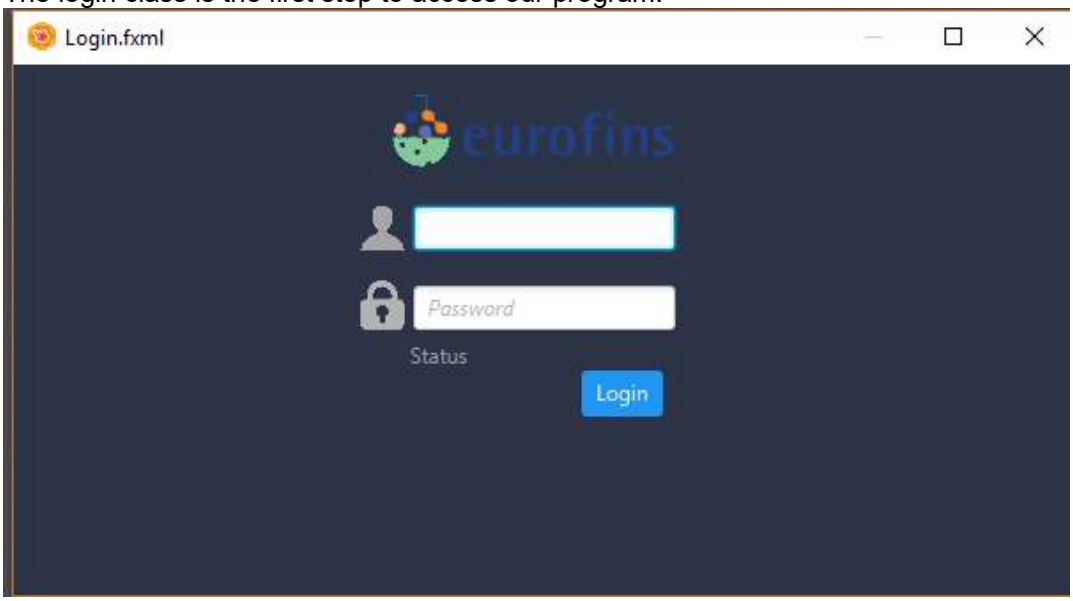


Illustration 3 – Login interface

From the GUI the program gets the users initials and the password. To check if the initials and the password are correct, the system reads the data from the database("Employees.bin"). For testing purposes, we start with a different database("EmployeesStart.txt"), so we could edit it and read it easily.

```
public String varifyLogin(String initials, String password)
{
   String text = "Initials or password is incorrect";
   for(int i = 0; i < employees.getSize(); i++)
   {
      if(employees.get(i).getInitials().equals(initials) && employees.get(i).getPassword().equals(password))
      {  //if the initials and password are correct
         text = "Login successful";
         if(employees.get(i).getIsATeamLeader())
            text += ".  Team Leader view";   //if the user is a team leader open team leaders view
         else
            text += ".  Employee view";      //if the user is an employee, then open employee view
      }
   }
   return text;
}
```

Illustration 4 – Login code

After logging in, the user would be redirected to another GUI, depending on what type of user is logging in and only then the user will be able to access the program.

WorkSchedule

Depending if a team leader or an employee is logged, then different pages will be open.

Employee

Employees can only view the work schedule. As a default, todays week schedule will be open, but if needed, then next week's schedule can be opened. Employees can request vacations, which will be accepted or rejected by team leaders.

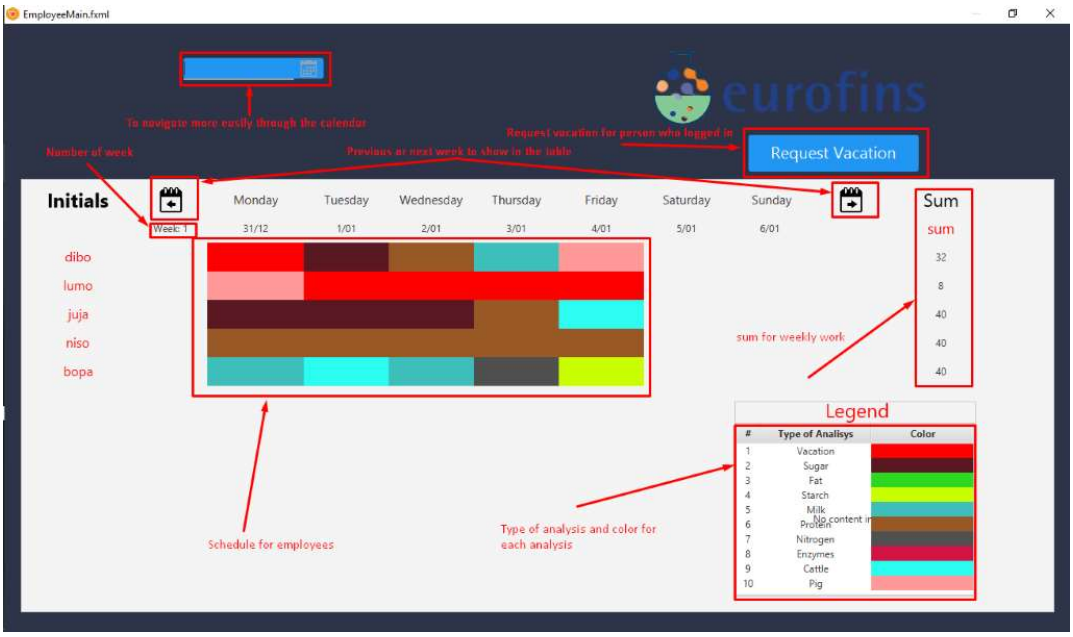


Illustration 5 – Employee schedule view

Team leader

Team leaders can view the work schedule and besides that they are able to create, edit and delete schedules. Besides that, team leaders must accept or reject vacation requested by their employees. The work schedule can be created to the future as well.

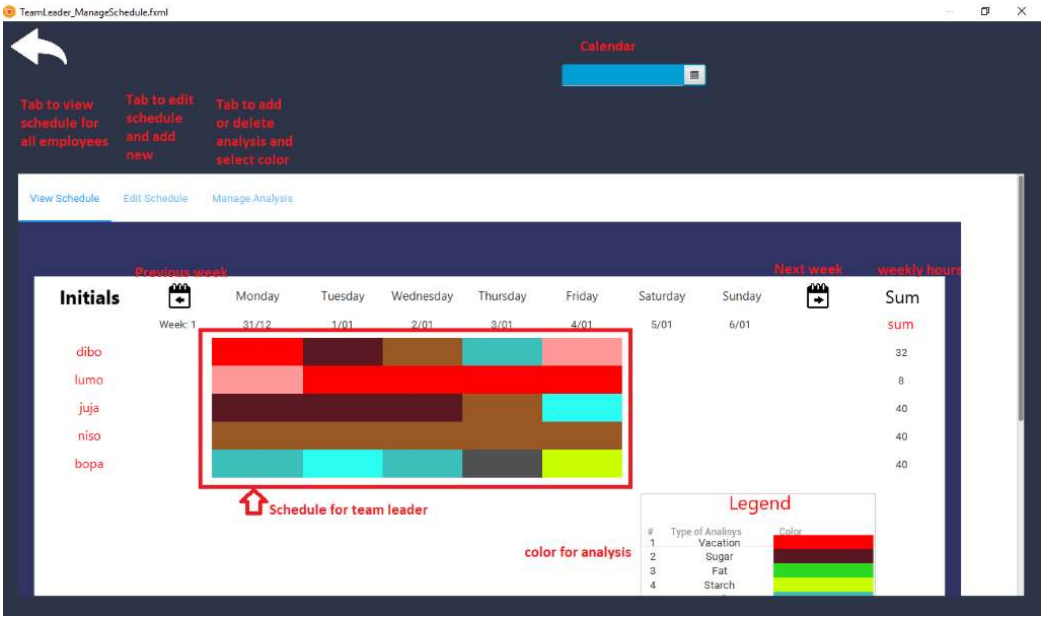


Illustration 6 – Team leader schedule view

Shift

Only team leaders can do anything with shifts. Team leaders can create, edit and delete shifts. To create a shift, the team leader must enter at least an employee and an analysis. For extra data, the team leader can enter a date, if not inserted, then todays date will be used. A note can be added if needed.

```
public Shift(Employee employee, Analysis analysis, String note)
{
   this.employee = employee;
   this.analysis = analysis;
   this.note = note;
   date = new Date();
}

public Shift(Employee employee, Date date, Analysis analysis, String note)
{
   this.employee = employee;
   this.analysis = analysis;
   this.note = note;
   this.date = date;
}

public Shift(Employee employee, Analysis analysis)
{
   this.employee = employee;
   this.analysis = analysis;
   date = new Date();
   note = "";
}

public Shift(Employee employee, Date date, Analysis analysis)
{
   this.employee = employee;
   this.analysis = analysis;
   this.date = date;
   note = "";
}
```

Illustration 7 – Shift code

6 Test

Login

```
Login login = new Login();
System.out.println(login.varifyLogin("MiVi", "4321")); //Login test
System.out.println(login.varifyLogin("DiBo", "4321"));
System.out.println(login.varifyLogin("DiBo", "1234"));
System.out.println("\n");
```

Illustration 8 – Login test input

```
Login successful.  Team Leader view
Initials or password is incorrect
Login successful.  Employee view
```

Illustration 9 – Login test output

Date

```
Date date = new Date();
Date date1 = new Date(12, 28, 2018);
System.out.println(date);  //for test class Date
System.out.println(date1);
System.out.println("\n");
```

Illustration 10 – Date test input

```
18/12/2018
28/12/2018
```

Illustration 11 – Date test output

Vacation

```
Vacation vacation = new Vacation(date, date1, list.get(2));
System.out.println(vacation);
vacation.approveVacation();
System.out.println(vacation);
System.out.println("\n");
```

Illustration 12 – Vacation test input

```
18/12/2018 - 28/12/2018
Employee: Justinas Jancys
Approved: false
18/12/2018 - 28/12/2018
Employee: Justinas Jancys
Approved: true
```

Illustration 13 – Vacation test output

Analysis and AnalysisList

```
Analysis analysis = new Analysis("Fat", "red", 1.5, 1.5, 1.5, 1.5, 1.5, 1.5, 1, 1, 1, 1, 1, 1, 3, 3, 3, 3, 3, 3);
Analysis analysis1 = new Analysis("Fiber", "green", 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 2);
AnalysisList anaList = new AnalysisList();
anaList.add(analysis);
anaList.add(analysis1);
System.out.println(anaList.get(0));
System.out.println(anaList.get(1));
System.out.println("\n");
```

Illustration 14 – Analysis and AnalysisList test input

```
Fat
red
All week:
Monday - 1.5
Tuesday - 1.5
Wednesday - 1.5
Thursday - 1.5
Friday - 1.5
Saturday - 1.5
Fiber
green
All week:
Monday - 1.0
Tuesday - 1.0
Wednesday - 1.0
Thursday - 1.0
Friday - 1.0
Saturday - 1.0
```

Illustration 15 – Analysis and AnalysisList test output

EmployeeFile and EmployeeList

```
EmployeeFile fileWork = new EmployeeFile();
EmployeeList list = new EmployeeList();
try
{
   list = fileWork.readTextFile();  //Test to check if file input from a text file works(works)
}
catch (FileNotFoundException e)
{
   // TODO Auto-generated catch block
   e.printStackTrace();
}

fileWork.setFile("Employees.bin");   //checks if data can be written to a binary file(works)
try
{
   fileWork.writeBinaryFile(list);
}
catch (IOException e)
{
   // TODO Auto-generated catch block
   e.printStackTrace();
}
System.out.println(list.get(0));
System.out.println("\n");
```

Illustration 16 – EmployeeFile and EmployeeList test input 1

```
list.remove(1);
list.remove(0);
try
{
   list = fileWork.readBinaryFile();   //Checks if the binary file can read(works)
}
catch (ClassNotFoundException e)
{
   // TODO Auto-generated catch block
   e.printStackTrace();
}
catch (IOException e)
{
   // TODO Auto-generated catch block
   e.printStackTrace();
}

System.out.println(list.get(1));//for test class employeeList and employee(works)
System.out.println("\n");
```

Illustration 17 – EmployeeFile and EmployeeList test input 2

```
Dima
Bors
04123123
DiBo@eurofins.dk


Micheal
Viuff
87465132
MiVi@eurofins.dk


Justinas
Jancys
16769423
JuJa@eurofins.dk
```

Illustration 18 – EmployeeFile and EmployeeList test output

Shift

```
Shift shift = new Shift(list.get(0), anaList.get(0));
Shift shift1 = new Shift(list.get(1), anaList.get(1), "Leave work early");
Shift shift2 = new Shift(list.get(2), date1, anaList.get(0));
Shift shift3 = new Shift(list.get(1), date1, anaList.get(0), "Will be late 30min");
System.out.println(shift);
System.out.println("\n");
System.out.println(shift1);
System.out.println("\n");
System.out.println(shift2);
System.out.println("\n");
System.out.println(shift3);
System.out.println("\n");
```

Illustration 19 – Shift test input

```
Employee: Dima Bors
Date: 18/12/2018
Analysis: Fat
red
All week:
Monday - 1.5
Tuesday - 1.5
Wednesday - 1.5
Thursday - 1.5
Friday - 1.5
Saturday - 1.5
Note:


Employee: Micheal Viuff
Date: 18/12/2018
Analysis: Fiber
green
All week:
Monday - 1.0
Tuesday - 1.0
Wednesday - 1.0
Thursday - 1.0
Friday - 1.0
Saturday - 1.0
Note: Leave work early
```

Illustration 20 – Shift test output 1

```
Employee: Justinas Jancys
Date: 28/12/2018
Analysis: Fat
red
All week:
Monday - 1.5
Tuesday - 1.5
Wednesday - 1.5
Thursday - 1.5
Friday - 1.5
Saturday - 1.5
Note:


Employee: Micheal Viuff
Date: 28/12/2018
Analysis: Fat
red
All week:
Monday - 1.5
Tuesday - 1.5
Wednesday - 1.5
Thursday - 1.5
Friday - 1.5
Saturday - 1.5
Note: Will be late 30min
```

Illustration 21 – Shift test output 2

6.1 Test Specifications

#	Requirement	Implemented/ Partly implemented/ Not implemented
1	Every user needs to login before use	
2	The schedule for an employee can be created/edited/deleted only by a team leader	
3	Notes can be added by team leaders to shifts	
4	Team leaders can remove old employees	
5	Team leaders can add new employees	
6	The work schedule can be color coded by the team leader depending on the type of analysis or work	
7	Employees should be able to search for individual times in the schedule through a search bar	
8	The work schedule should have a button to go to current date	
9	Every employee can request a vacation	
10	Team leaders need to verify (accept/reject) vacations	
11	Every user can view individual and team work schedule	
12	Vacation verification should pop up for the team leader	
13	There needs to be a legend for the color codes	
14	When the team leader is creating a work schedule, individual employee preferences should be shown	
15	If an employee only works 3 days a week, then the employee shouldn't be to be booked the other days	

Table 4 – Test results

7 Results and Discussion

Within the program, nine of the requirements has successfully been implanted and tested; 1, 2, 3, 4, 5, 6, 9, 10 and 13. Requirement 7, 11 and 12 has only been partially implemented. Requirement 8, 14 and 15 has not been implemented.

8 Conclusions

The requirements that was stated were not all met. The system failed to have a button to go to today's date in the schedule view, have employee preferences shown when creating shifts and to recognize if a work restricted employee has had the allowed work hours through a week. Furthermore, the search bar was not completely implemented, an employee can only view the team schedule not his individual and the Vacation button displayed on the team leaders page does not emphasize when there's a pending vacation request. However, the rest of the requirements have been fulfilled.

All the cases within the use case diagram has been fulfilled, where both the employee and team leader can perform all the cases displayed. Furthermore, the steps described in all the use case descriptions goes a long with the GUI implemented, except for those including the requirements that were not met.

The class diagram was implemented as shown and later tested within the test class. Through the test the missing and partially implemented implementations were found and accounted for.

In conclusion, the program has been partially completed, however, the main functionalities of the program have successfully been implemented.

9 Project future

Since our project is not complete, the first thing for the project's future is to complete it. Finish up on the controller, finish the workSchedule class and to connect GUI with the controller.

If we would desire to make the project ready for use, we would need to use databases instead of files, put everything on a server, modify the program, that the system could be accessed on a computer that has the program installed. Maybe even create a project on the website, so that the only software you need is a browser. Remake the design of the project, make it more general, so the project could be used by more than one company.

10 Sources of information

Eurofins, 2018. *About us.* [Online]
Available at: https://www.eurofins.com/about-us/
[Accessed 18 December 2018].

Eurofins, 2018. *Vores ydelser.* [Online]
Available at: https://www.eurofins.dk/foedevarer/vores-afdelinger/
[Accessed 20 September 2018].

NISO, 2010. *Scientific and Technical Reports -,* Baltimore: National Information Standards Oganization.

VIA Engineering, in preparation. *Confidential Student Reports,* s.l.: s.n.

Viuff, 2018. *Presentation of work planning tool.* [Online]
Available at: http://ict-engineering.dk/Course/SEP1-A18/WorkPlanToolPresentation.pdf
[Accessed 18 December 2018].

Viuff, A., 2018. *ItLearning.* [Online]
Available at: http://ict-engineering.dk/Course/SEP1-A18/WorkPlanToolPresentation.pdf
[Accessed 8 October 2018].

Viuff, A., 2018. *WorkPlanToolSpreadsheet.* [Online]
Available at: http://ict-engineering.dk/Course/SEP1-A18/WorkPlanToolSpreadsheet.xls
[Accessed 18 December 2018].

11 Appendices

Appendix A Project Description

Project description

Work schedule

ICT ENGINEERING IT-SSE1Z-A18

Lucas Kaas Møller, 279967
Bors Dementie, 279948
Pavliuc Bogdan, 280156
Justinas Jancys, 280151
Sova Nicoleta, 267069

Number of characters: 4946

Supervisors:
Michael Viuff
Mona Wendel Andersen

Table of Contents

1 Background description ..... **Error! Bookmark not defined.**

2 Definition of purpose 28

3 Problem statement 28

4 Delimitation 28

5 Choice of models and methods 29

6 Time schedule 30

7 Risk assessment 30

8 Sources of information 31

Project description

Background description

Eurofins Scientific is an international company which provides pharmaceutical testing of the products, environmental and agroscience service. (Eurofins, 2018). Eurofins has a worldwide network of laboratories, so it is one of the global market leaders in testing field and laboratory services.

Currently, Eurofins uses Microsoft Excel, as a template for organization- an old-fashioned way of providing information about every employee and tests for such a big company. Moreover, the problem is that their work is hard and slow, with little possibilities, which do not come to their necessities. Excel sheets are updated manually by team leaders and consists of a bunch of information added and edited for each employee separately, which is why they spend much time introducing all the necessary information, when it could be used in a more efficient way. (Viuff, 2018)

The first table is actual work plan. It assignes duties for every worker during the week and also where they can manage their vacation and days off. The sheet number 2 shows all departments depending on season or week .The sheet number 3 dispays the training overviw (trained up to date, training in process, need of training).

Also, all the approvals and changes in schedule are made just by the managers, and this changes are made using diffferent colours in order to undestand the sheets.

For the vacation yellow colour means that not yet approved and red is for ones that are approved, green is for trainning and blue means that the emploeey works in other department. For the table number 3, colours are used too, pink means training in process and red indicates the need of training .

Color code is an important feature for the future application as it is much viewable and gives information in an interactive way. Eurofins wants a single user system and would like a tool that easily allows to fill out the work week template, by having all the data present in one program, taking into account the preferences of each technician. Also, Eurofins would like to be able to store loads of data about their employees and workplace and to have

an easy access to see training areas of employees or their period of vacation with a foreseeable interface and flexible regarding system input possibilities.

2 Definition of purpose

The purpose of this project is to create a tool that will solve the customer's time schedule problem.

Delivering this system will help the managers to maintain their time schedule more organize, and make their work much easy and comfortable, so that the employees of Eurofins could easily know what needs to be done every day, and the managers would have an easy platform to create the time schedule.

3 Problem statement

Main problem:

What are the main needs in order to make a system that will help to organize a schedule?

Sub problems

- What do we need to know about the areas that employees are trained in?
- What do we need to know about the vacation request and days off?
- What will be the way of accessing and displaying the schedule?
- What type of information the managers/employees can input into the system?
- What is needed to be done for the software to be flexible for the user?

4 Delimitation

- All data will be stored locally
- Only one user can be logged in at a time

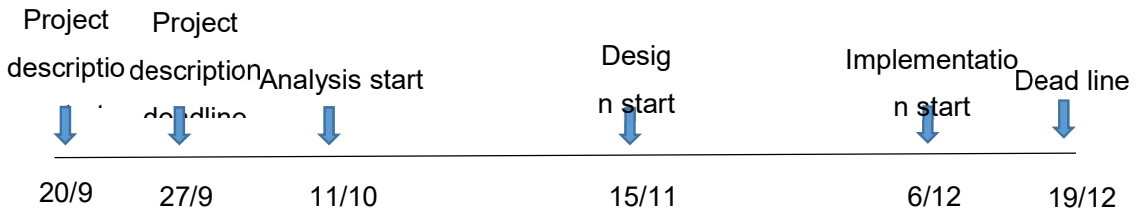
5 Choice of models and methods

What **- partial problem**	**Why** **- Study the problem**	**Which** **- methods / models / theories**	**Who** **- has the main responsibility for this point**
What do we need to know about the areas that employee is trained in?	The system won't allow an employee to be put on a station if the employee has not completed training for the station in question and a record of this has been made.	Comparing the employee's skills in order to create teams.	Justinas
What do we need to know about the vacation requests and days off?	Different types of vacations should be offered as a choice for the employee. In the program, it should be displayed whether the request has been approved, rejected or is pending.	Knowledge and literature from SDJ1. Analyze the different types of vacations, and the right to so.	Dementie
What will be the way of accessing and displaying the schedule?	The schedule will be accessed and displayed using the offered tool.	Literature firm SDJ1	Lucas
What type of information the managers or employees can input into the system?	The information that will be input into the system depends on either are you a manager or an employee.	Knowledge and literature from SDJ1 will be used to solve this problem.	Bogdan
What is needed to be done to the software to be flexible?	To provide an easy and comfortable way of making the schedule.	Knowledge and literature from SDJ1 will be used to solve this problem.	Nicoleta

		Create an interface that allows changes when it is a need.	

6 Time schedule

The time scope is estimated at 500 hours. The time schedule is estimated as followed:



The project description will need to be handed in on 27/9/2018 and the final deadline is for 19/12/2018.

7 Risk assessment

Risks	Description	Likelihood Scale: 1-5 5 = high risk	Severity Scale: 1-5 5 = high risk	Product of likelihood and severity	Risk mitigation e.g. Preventive & Responsive actions	Identifiers	Responsible
Risk 1	Lack of time before hand-in	4	4	16	Control of time schedule, work on weekends	Making excuses, blaming others,	Bogdan
Risk 2	Making the program work	3	4	12	Ask the supervisors for help	Blaming others	Justinas
Risk 3	Not meeting our own expectations	2	4	8	Work on the weekends, group meetings	Lack of knowledge	Lucas

Risk 4	Less meeting then expected	4	5	20	Control of time schedule, communication in the group	Absence, laziness	Bogdan
Risk 5	Delays	3	5	15	Communication in the group, working together, asking for help	Laziness, lack of knowledge	Justinas
Risk 6	Meeting deadlines	4	5	20	Control of time schedule	Stress, lack of knowledge	Lucas
Risk 7	Wrong priorities	3	4	12	Communication in the group, assessing individual strengths	Lack of experience	Dementie

8 Sources of information

Eurofins, 2018. *Vores ydelser.* [Online]

Available at: https://www.eurofins.dk/foedevarer/vores-afdelinger/

[Accessed 20 September 2018].

Viuff, A., 2018. *ItLearning.* [Online]

Available at: http://ict-engineering.dk/Course/SEP1-A18/WorkPlanToolPresentation.pdf

[Accessed 8 October 2018].

ICT Engineering. (2012). Project Description guidelines. Retrieved from Studienet:

https://studienet.via.dk/projects/Engineering__project_methodology/General/Guidelines/2018%20Project%20Description%20(Appendix%201)%20VIA%20Engineering%20Guidelines.pdf

Appendix B Use Cases

Item	Value
Use case	Login
Summary	User logs into an account
Actor	Employee Team leader
Precondition	The user must have an authenticated account
Postcondition	The user can access the program
Base sequence	1. The user's interface will show a login screen consisting of a "username" field and a "password" field as well as a "login" button. 2. The user inputs the login data. 3. Click "login" 4. The user will be taken to the program. Employees will see the schedule and team leaders will see their start-up page. 5. The user can log out when done using the program.
Branch sequence	None
Exception sequence	If the login input is invalid, return to step 2
Sub use case	
Note	Once an employee has been removed by a team leader, their login credentials are no longer valid.

Item	Value
Use case	Access schedule (view)
Summary	Employees can check their work schedule
Actor	Employee
Precondition	User must be logged in
Postcondition	None
Base sequence	1. User clicks on the “date” bar 2. User selects a date 3. The week where the selected date is within is shown. 4. By clicking on the “Today” button, the current week will be shown.
Branch sequence	None
Exception sequence	None
Sub use case	None
Note	None

Item	Value
Use case	Request vacation
Summary	Employees can request vacation
Actor	Employee
Precondition	Users must be logged in
Postcondition	Vacation has been requested and sent for approval by team leader
Base sequence	1. Click the “Request vacation” button 2. Type in the start and end date of the vacation 3. Click “Submit request” 4. The request has been sent to the team leader for approval
Branch sequence	None
Exception sequence	If the date is prior to todays date, the request cannot be submitted (Step 4 will not be executed.)
Sub use case	None
Note	None

Item	Value
Use case	Manage analyses
Summary	The team leader can create, edit and delete the analyses that are required.
Actor	Team leader
Precondition	The team leader must be logged in and access the “Manage analyses” page
Postcondition	Analyses data has been changed
Base sequence	To add an analysis: 1. Fill out credentials (Type of analysis, color, number of employees for all weeks, small and large week). 2. Click "Add". 3. The analysis will be displayed in the analysis list. Edit: 1. The team leader finds an analysis. 2. Clicks "Edit" button next to the analysis. 3. Line will be editable and instead of the "edit" button, "Save" and "Cancel" will be shown. 4. The team leader will change the data needed. 5. Click "Save" 6. Store the data in the system Delete: 1. The team leader finds an analysis. 2. Click "Delete" button next to the analysis. 3. The "delete" button will change to "yes" and "no" buttons. 4. The analysis is removed from the list.
Branch sequence	None
Exception sequence	If all the credentials are not filled out, an error will show (Step for add 3 and step 6 for edit will not be executed.)
Sub use case	None
Note	None

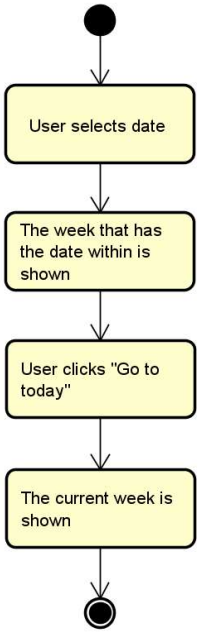
Item	Value
Use case	Pending vacations
Summary	Team leader accepts or declines the requested vacation of an employee
Actor	Team leader
Precondition	The team leader must be logged in.
Postcondition	Vacation request has been either approved or rejected.
Base sequence	1. If any vacation requests are pending, the “Vacation” button will be emphasized. 2. Click the “Vacation” button. 3. A list of requested vacations will be shown, with the employee and requested dates. 4. The team leader can approve or reject the request by clicking the corresponding buttons. 5. Once the vacation has been approved, it will be shown in the work schedule for both employees and team leader. 6. If the vacation has been approved, it will be shown in the work schedule for both employees and team leader. 7. Once a request has been rejected or approved, it will be removed from the list.
Branch sequence	None
Exception sequence	None
Sub use case	None
Note	None

Item	Value
Use case	Create shift
Summary	Team leader can create a shift
Actor	Team leader
Precondition	The team leader must be logged in and accessed the "Manage schedule" page. Employees and analysis must have been created.
Postcondition	Shift has been created and added to the work schedule.
Base sequence	1. Team leader clicks "Add shift". 2. A new window opens. 3. The team leader chooses type of analysis. 4. The team leader chooses the day of the shift. 5. The program displays number of employees required for 6. the shift. 7. The team leader chooses employees 8. The team leader clicks" save". 9. The shift will be added to the work schedule.
Branch sequence	None
Exception sequence	
Sub use case	None
Note	Employees who has a preference not to work at this type of analysis will be marked with color. Employees who are not trained in the type of analysis or has requested vacation will not be displayed as an option.

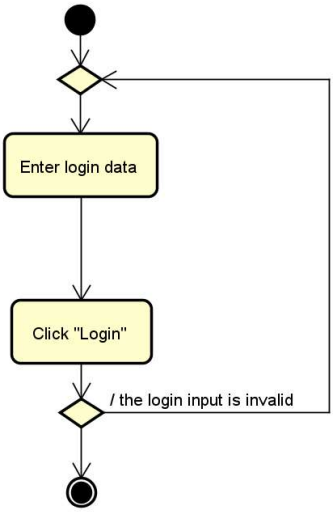
Item	Value
Use case	Manage schedule
Summary	Changes that a team leader can make on employees' schedules.
Actor	Team Leader
Precondition	User need to be logged in and accessed “Manage Schedule”. There must have been created shifts.
Postcondition	None
Base sequence	1. The schedule is shown on the screen. 2. The team leader can “double click” on a shift to add a comment to the shift. 3. The team leader can right click on a shift to edit it.
Branch sequence	None
Exception sequence	None
Sub use case	None
Note	The changes can be cancelled any time

Appendix C Activity diagrams

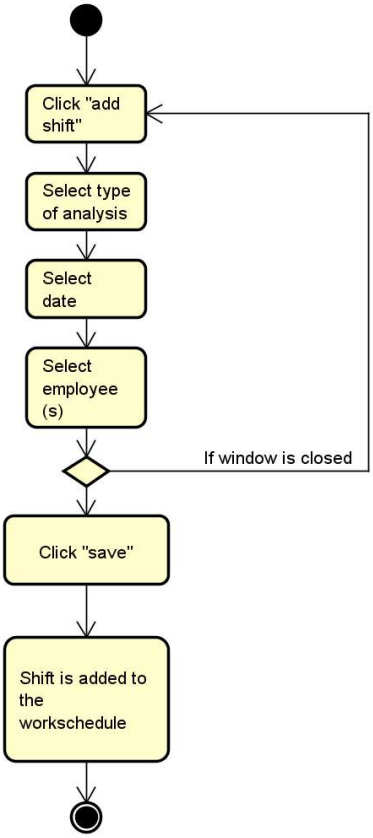
Access schedule activity diagram



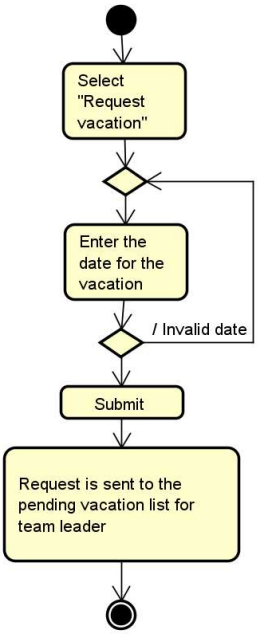
Login activity diagram



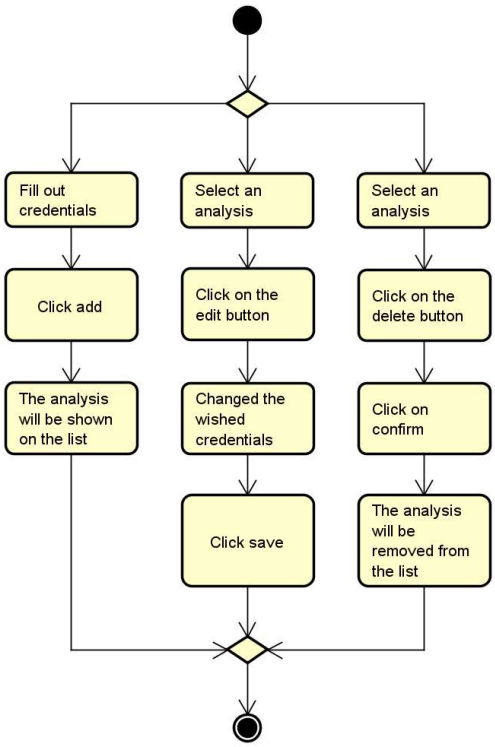
Create shift activity diagram



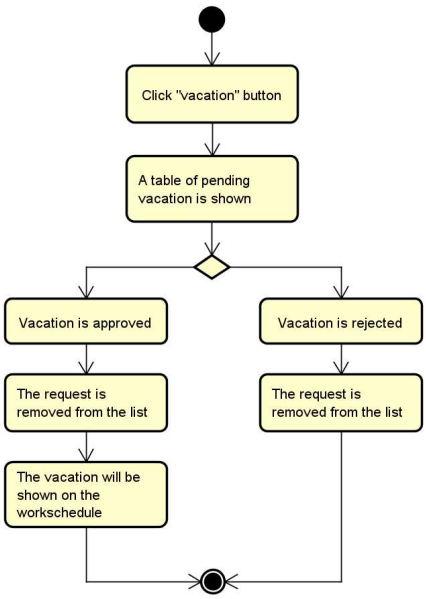
Request vacation activity diagram



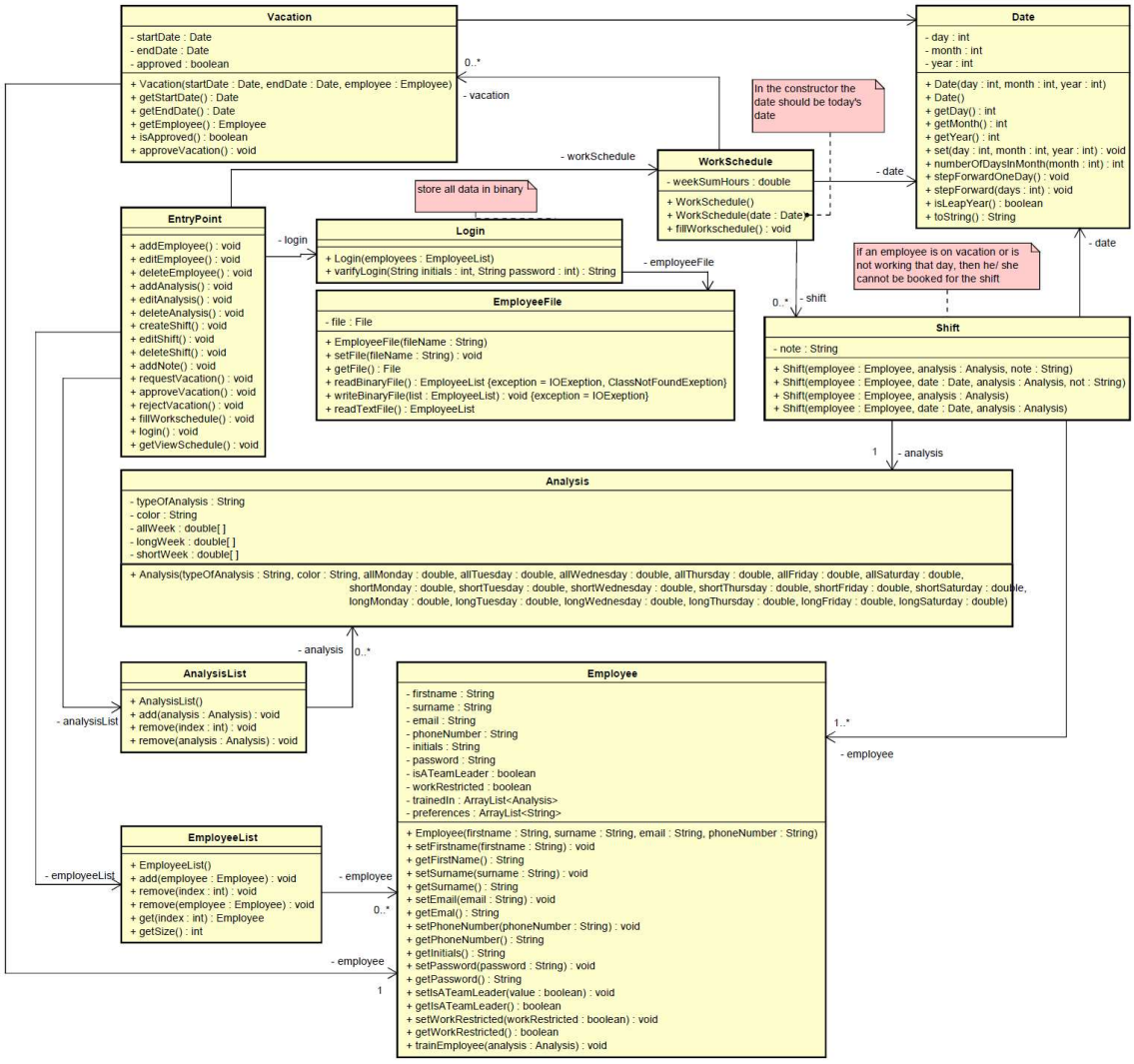
Manage analysis diagram



Pending vacation activity diagram



Appendix D Class diagram



Appendix E User guide

1. Overview of the program

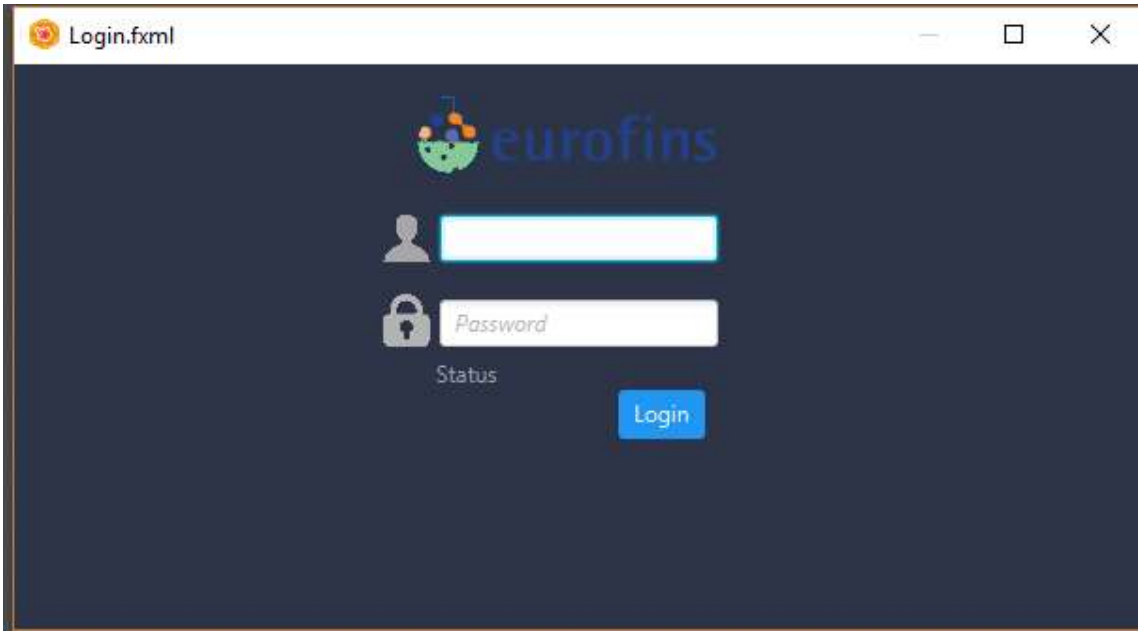


Figure 7

Description:

This screen is the start of the program. There is a login interface, when you need to put "*Initials*" and "*password*" to login as a Team Leader or as an Employee.

2. Login as a team leader

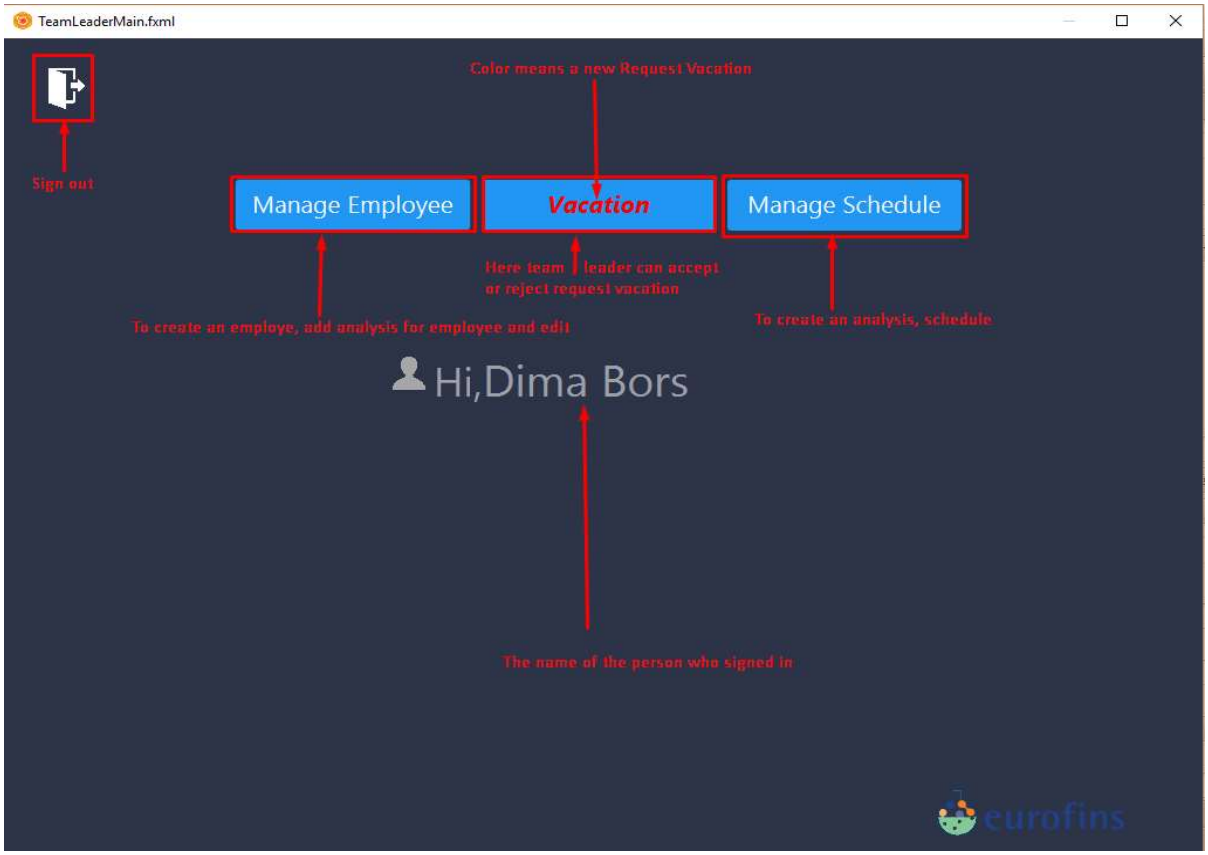


Figure 8

On first page for team leader will show 3 buttons.

User (Team Leader) - Manage Employee – To create, delete, edit an employee, the Team Leader need to go to "Manage Employee" (*Figure 2*);

Figure 9

Create an employee: fill out credentials (*Figure 3*), and press the button "+Add employee"

Edit an employee: press the button "Edit" next to the Employee, line will be editable, change text, then save (*Figure 3*).

Delete an employee: press the button "Delete" next to the employee (*Figure 3*), press "Yes" or "No" to confirm.

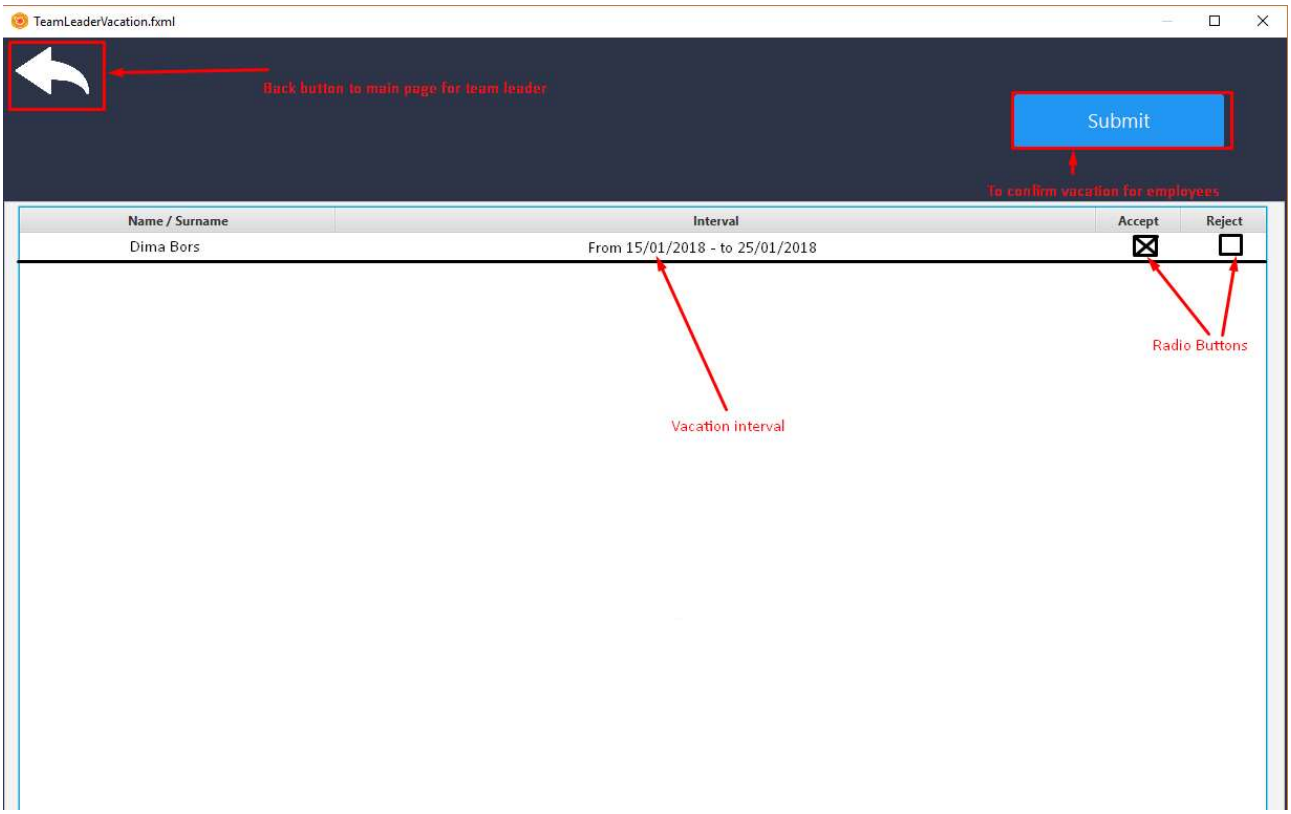


Figure 10

User (Team Leader) – Vacation – If any vacation request is pending, the "Vacation" (*Figure 2*) button will be emphasized. Click the "Vacation" will open a new window with the employee and requested dates (*Figure 4*). Team Leader can "Accept" or "Reject" the request by clicking the corresponding buttons, then press button "Submit".

User (Team Leader) – Manage Schedule – To work with schedule the Team Leader need to click "Manage Schedule" then will open a new window (*Figure 5*). On this page the Team Leader will be able to see schedule for all employee **Legend** for all

type of analysis with custom colors. Schedule when Team Leader can go to previous week or next week clicking on icon to navigate through the calendar. (*Figure 5*).

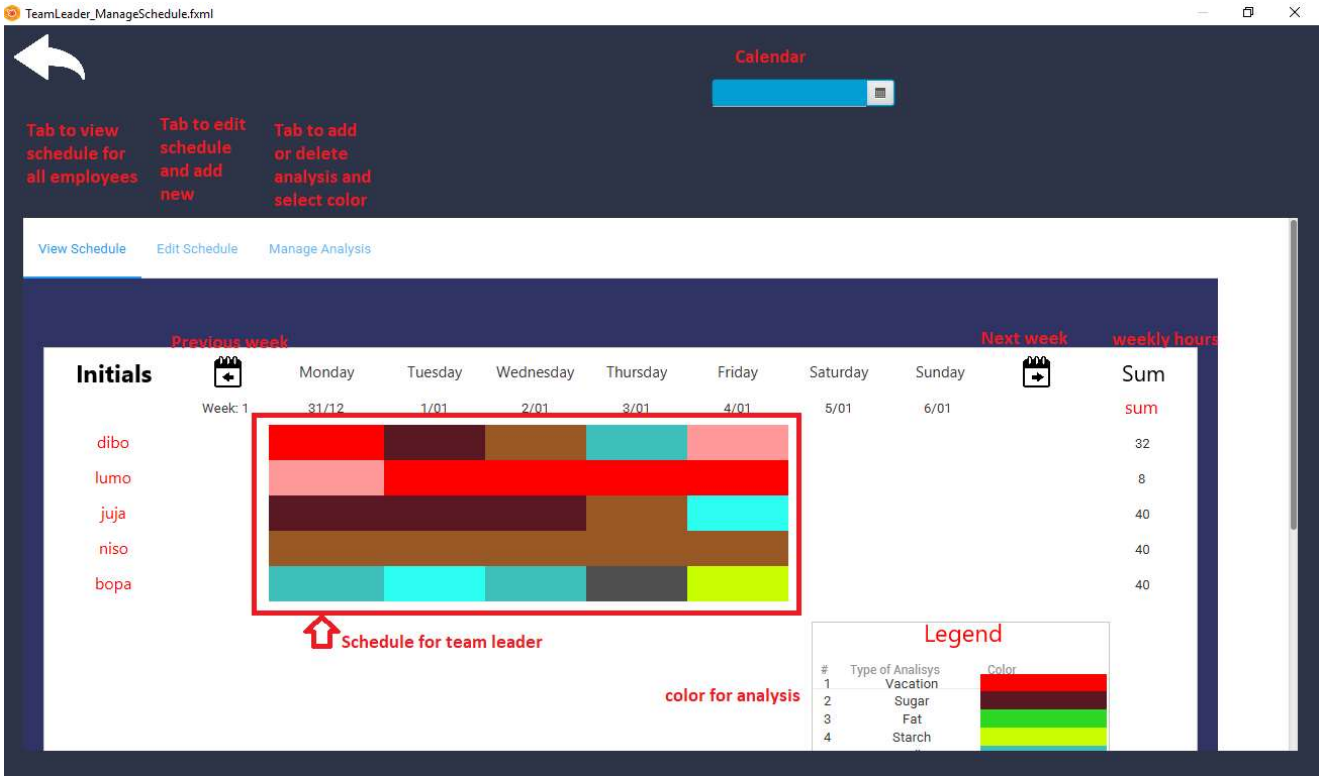


Figure 11

Edit Schedule: To edit schedule the Team Leader need to click on "Edit Schedule" chose tab "Edit" selected the employee by initials click "Edit" (*Figure 6*) and edit schedule.

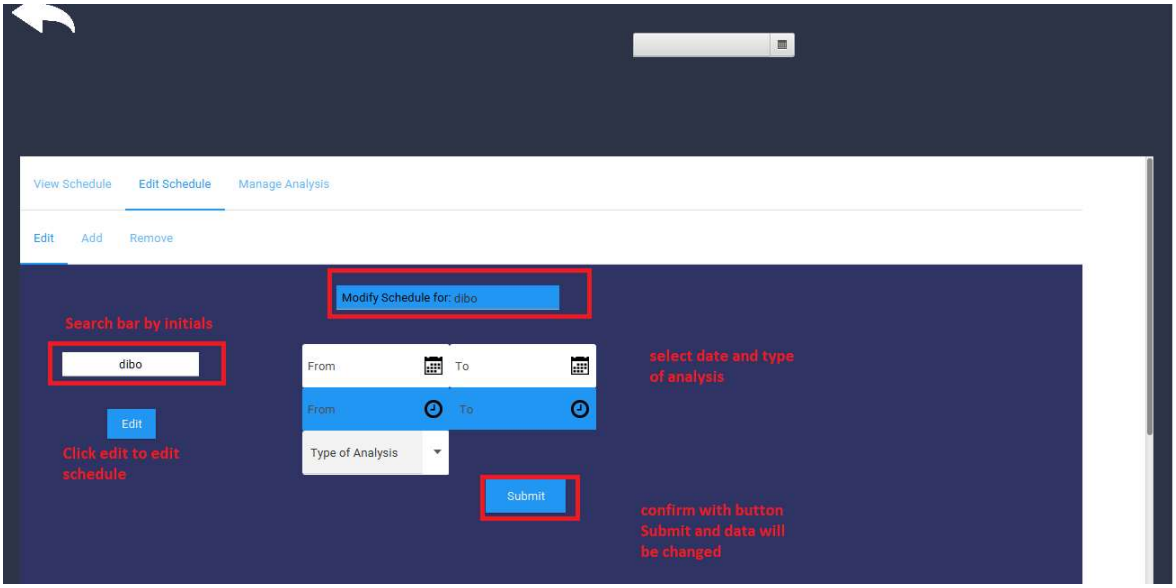


Figure 12

Add Schedule: To add a new schedule the Team Leader need to click on "Edit Schedule" and "Add". The Team Leader will choose date, type of analysis, and employee, then the program will display number of employees required for the shift, choose employee and "Submit" (Figure 7)

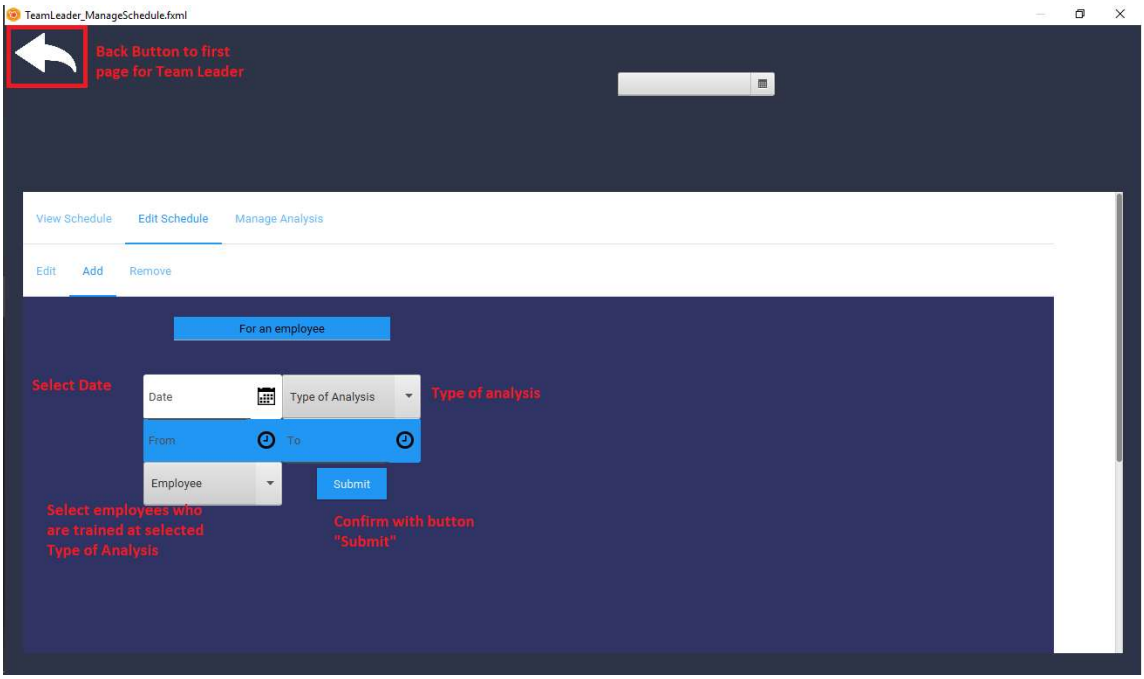


Figure 13

Remove Schedule: To remove the schedule the Team Leader need to click on "Edit Schedule" and choose tab "Remove" search the employee in the search bar, select start and end date, click "Delete" (*Figure 8*).

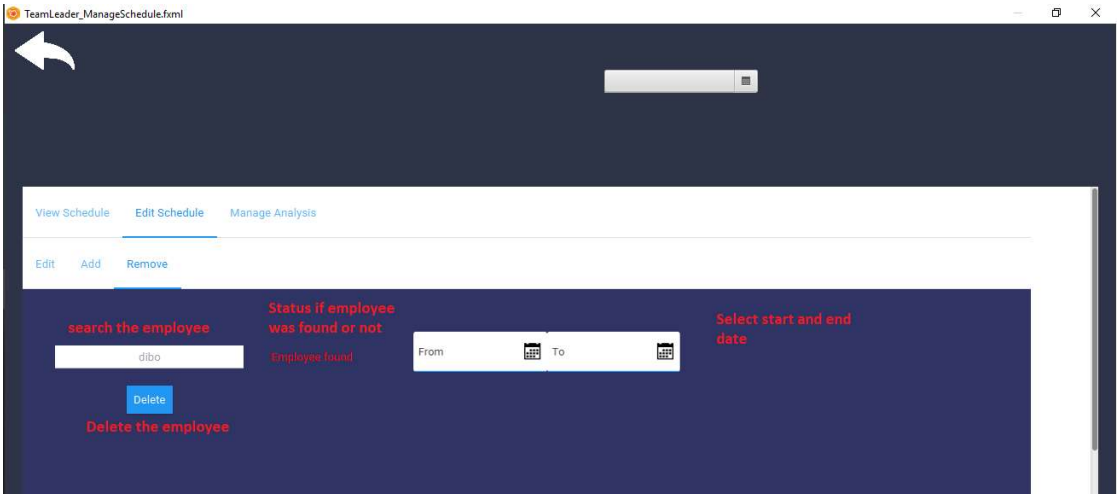


Figure 14

User (Team Leader) – Manage Schedule – Manage Analysis – The Team Leader will choose "Manage Analysis" to add, edit or delete analysis (*Figure 9*).

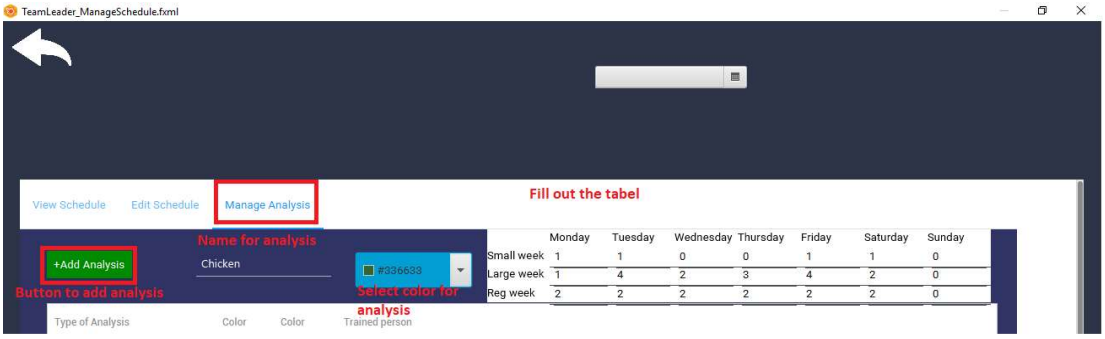


Figure 15

Add Analysis: Fill out the credential Type of analysis, color, number of employees for large, small and regular week after that click "+Add Analysis" to save (*Figure 9*).

Edit Analysis: Click "Edit" button next to the analysis to edit the selected analyses. To save click "Save" to stop editing click "Cancel" (*Figure 10*).

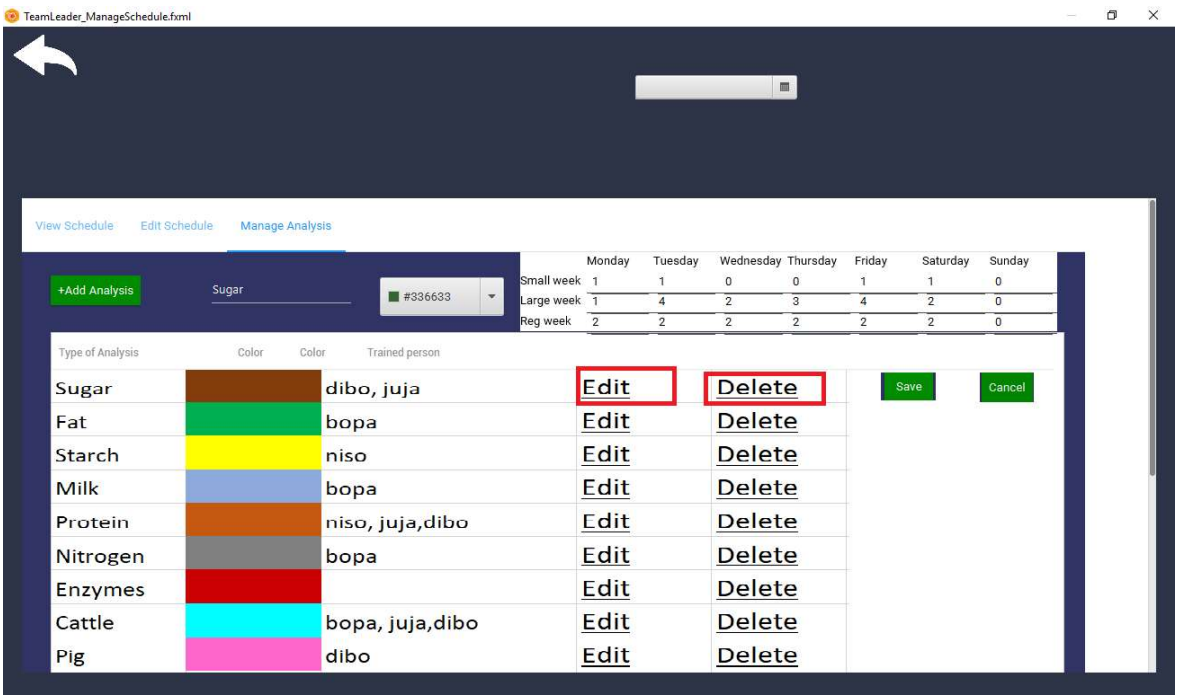


Figure 16

Delete Analysis: Click "Delete" button next to the analysis and confirm with "Yes" or "No"

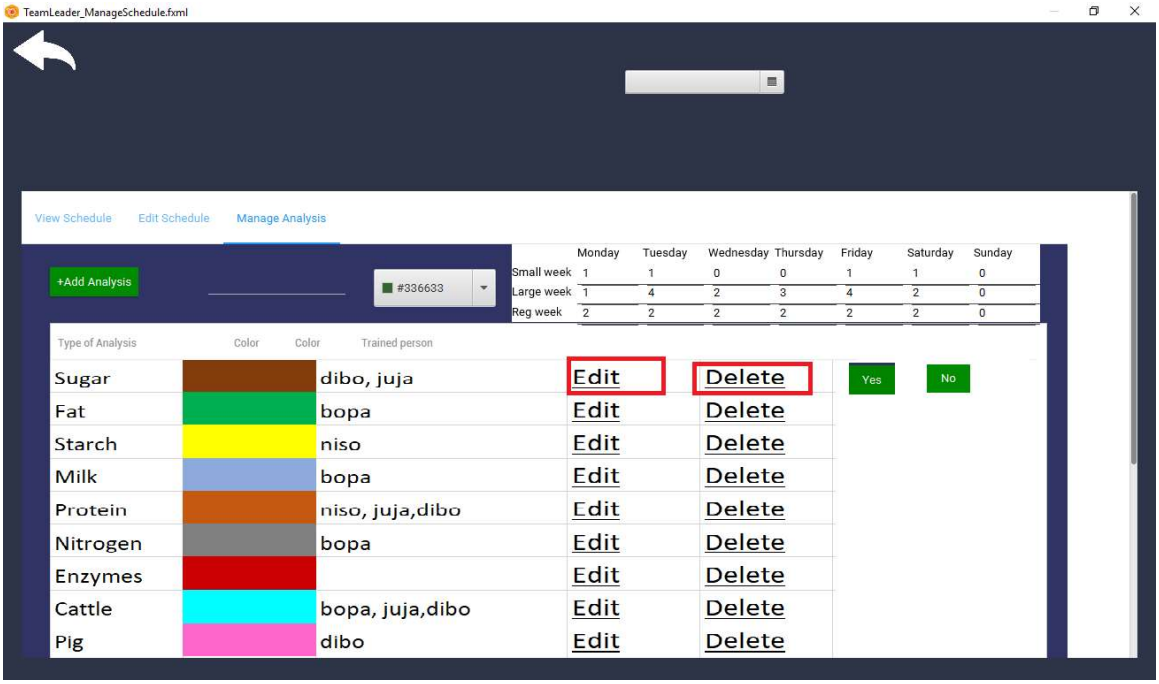


Figure 17

3. Login as an Employee

User (Employee) – on first page for employees will show schedule (Figure 12). If employee want to see next/previous week schedule, he/she need to click or . Sum column will show work weekly hours. Bellow will be **Legend** with Type of analysis color. (*Figure 12*).

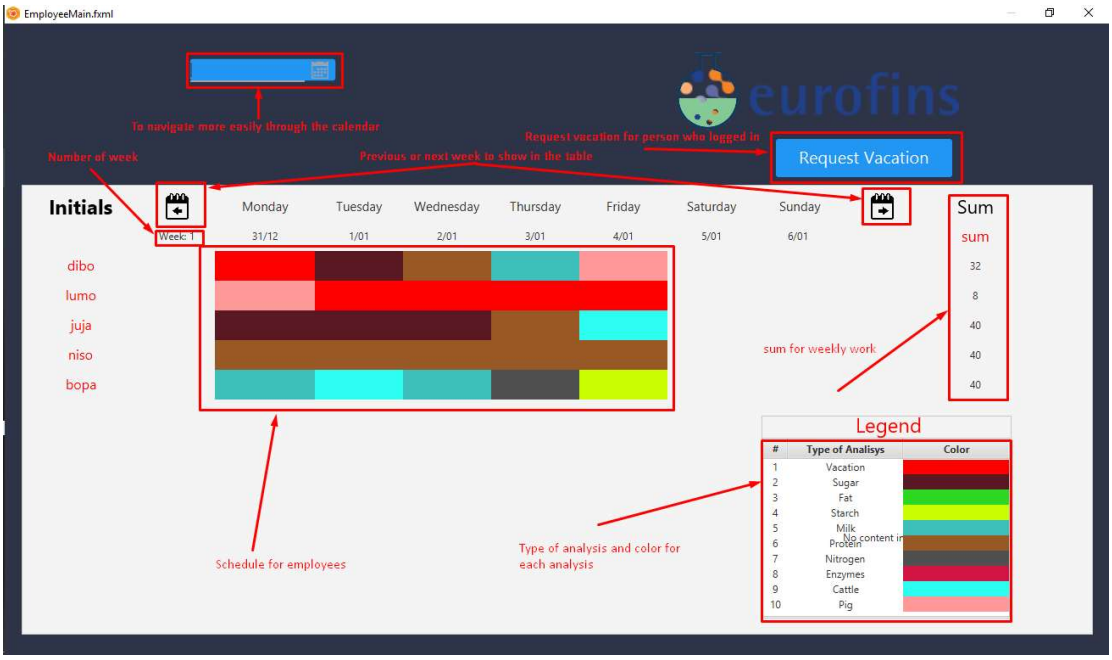


Figure 18

Request vacation: Click "Request Vacation" button, will open a new window (*Figure 13*). Select a start and end date of vacation. Click "Submit" to send a request vacation, Team Leader will have a notification (*Figure 2).*

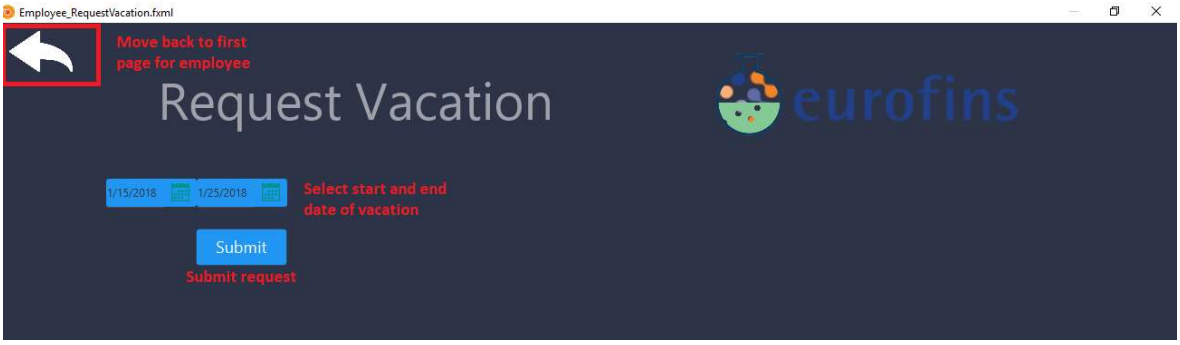


Figure 19