

VIA University  
College

# IT Project Management System

## Project Report

Group 4

Group Members: Henrik Koster (305916), Kim Tranberg (172394),  
Aleksandrs Bistrovs (304542)

Supervisors: Astrid Hanghøj, Steffen Vissing Andersen

VIA University College, Campus Horsens

Character Count:

Software engineering  
Semester 1

Date: 18/12/2020

<b>Abstract</b>	<b>2</b>
<b>1. Introduction</b>	<b>3</b>
<b>2. Analysis</b>	<b>4</b>
2.1. Functional requirements	4
2.2. Non-functional Requirements	6
2.3. Use Case Diagram	7
2.4. Use Case Descriptions	8
2.5. Activity Diagrams	10
2.6. Domain Model	12
<b>3. Design</b>	<b>13</b>
3.1. Class diagram	14
3.2. Sequence diagram	16
3.3. GUI views	17
3.4 Website	20
<b>4. Implementation</b>	<b>20</b>
<b>5. Test</b>	<b>22</b>
5. 1. J-Unit test	23
5. 2. Use Case test	24
<b>6. Results and discussion</b>	<b>28</b>
<b>7. Conclusions</b>	<b>29</b>
<b>8. Project Future</b>	<b>30</b>
<b>9. Sources of information</b>	<b>31</b>

# Abstract

The aim of the following report is to go through the process of creating an IT project management system by going through the main stages of the report - analysis, design and implementation. It begins by the development team gathering and analysing factual data from the customer company "Colour IT" to document the needs of the customer, as well as, the requirements for the system. The analysis is continued by the design, where careful thoughts and considerations are put forward by the team in order to achieve the desired result and meet the customers expectations. When analysis and design are completed, and the team has made considerable amounts of thought behind the desired product, the implementation phase begins.

Once the program is implemented, a variety of tests are performed to ensure that the program functions as intended, while identifying the features which are defective or missing. At this point, the product is considered to be in its 'finished' state and the team assembles to discuss the results and determine whether the project was a success or not.

The system was implemented in Java with the help of JavaFX to create a GUI. Implementation of the code was chosen to be done in a git repository, hosted on the free git hosting service called GitHub

In the end, the project management system achieved basic functionality as intended.

# 1. Introduction

At the beginning of the first semester, software engineering students were contacted by a company named “Colour IT”. The company has provided the students with a task to design and develop a system for managing IT projects, as well as the website for their customers with the information about their projects. The customer - Mr. Colour has specifically asked for a system with a functionality to create projects, split the work between the team members and track the progress of the projects (Mr.Colour, 2020).

In order to gain a better understanding of the problem, the development team has decided to introduce the reader to what exactly is an IT management system, its importance in today's world, advantages/disadvantages and possible challenges to be encountered during the development of such a system.

IT management system is a tool to assist with the scheduling of the projects, task management and team coordination. The most important function of the mentioned system is to help the managers with day-to-day management responsibilities. It is especially important today in the age of technology to keep yourself within the range of competition, as nearly every industry relies on IT solutions (Kashyap Vartika, 2020).

The advantage of using such a system is to stay afloat with your competitors, reduce the time waste by eliminating the updating the schedules and timetables by hand, ensure that all the team members are on board with the project, as well as ensure that all the necessary information is in one place. (Mavenlink, 2020)

Disadvantages to consider are the high costs of implementation of an IT system in the workplace, adoption to the technology by the employees, as well as the time spent training the employees to use the system. It can all be very costly and time consuming, which in turn creates a challenge for the developers to make the system affordable and easy to use. The goal of the IT management system is to speed up the processes within the company instead of creating an extra burden. (Mavenlink, 2020)

Taking all of the above to consideration and, given little to-no experience and the time restrictions, the development team will be only able to carry out IT solutions for small company projects. It is important to mention that this project report will not include any budget considerations and extra occurring costs in case of not meeting the requirements of the customer.

The following report is split into seven parts:

- 1) Analysis, where the reader will be able to find an interpretation of the system by the authors based on the interview with Mr. Colour. Also, read through the requirements and use cases. This chapter will result in the Domain model of the system, which will be the base for the following part.
- 2) Design, where the reader will be able to get familiar with the Class diagram and Sequence diagram in order to get a more in-depth understanding of the methods and developed IT system itself.
- 3) Implementation, where the reader will be guided through the implementation of the system with the code snippets and their descriptions.
- 4) Testing, where the reader will be able to see if the content of the requirements have been fulfilled during the project. (do we need this part?)
- 5) Results and discussion, where the reader will get familiar with the outcome of the project and achieved results by the authors.
- 6) Conclusions, where the reader will see the compilation of results from each chapter of this project.
- 7) Project future, where the reader will see the authors reflections on the technical viewpoint of the project in the future.

## 2. Analysis

The analysis was the most time consuming stage of the project because it establishes the framework and context of the product. As such, a great deal of care went into deciphering exactly what the consumer wanted their program to do and determining which features were fundamentally required.

### 2.1. Functional requirements

Firstly, a list of requirements was formulated. The requirements were arranged in chronological order from the most critical features to the least critical ones. These requirements were continually revised over a long-period of time with changes being made even as far as the design and implementation stage. Lastly, a list of non-functional requirements were created. As the name suggests, these requirements

were not necessary for the general functionality of the product, however they would only enhance it further given enough development time. For this reason, they would only be implemented once all the functional requirements are fulfilled.

**Critical priority:**

1. As a project creator, I want to be able to create new projects in the system, so that our projects will become easier to manage.
2. As a project creator, I want to be able to assign team members to a project, such that they can work on projects.
3. As a product owner, I want to be able to add and remove requirements, to make sure the project will meet the customers specifications.
4. As a scrum master, I want to be able to add tasks to the project, such that the rest of the team members know what tasks they have to work on.
5. As a project creator, I want all information from the system stored in a single file, so that information will not be lost when the system is closed.

**High priority:**

6. As a project creator, I want to be able to assign roles to team members, so it will be clear what kind of responsibility they have.
7. As a product owner, I want to be able to prioritize requirements based on their importance(Low, Medium, High, Critical), such that features which are critical for the system to function will be developed first.
8. As a product owner, I want to change who is responsible for a requirement in case another team member is more suited for handling that requirement.
9. As a product owner I want requirements to automatically get marked with "Ended" when all tasks for a requirement are done, so that I can see what requirements I should be testing.
10. As a scrum master, I want change who is responsible for a task in case another team member is more suited for handling that task.
11. As a product owner, I want requirements to contain an id, user stories in who, what, why template, estimated time, a deadline, who is responsible, status, total hours spent, such that I can easily get an overview of all relevant information for a requirement.
12. As a product owner, I want to be able to approve or reject requirements, such that the project can reach a finished state, and make sure it meets the customers needs.

13. As a scrum master, I want each task to contain all information (Requirement ID, task ID, title, time estimation, deadline, responsible team member, status, hours spent) such that I can easily get an overview of all relevant information for a task.
14. As a team member, I want to be able to register a total amount of hours to the system whenever a task has been finished, so that I can keep track of its progress and see our productivity in regards to how well we can estimate time for tasks.
15. As a product owner, I want requirements to display total time spent, which should automatically be a total of time spent from all related tasks, so it will be possible to track the progress of a requirement.
16. As a product owner, I want the total estimated time of all tasks related to a specific requirement, be exactly the same amount as the estimated time for that requirement, such that the project will stay on schedule.
17. As a scrum master, I want to mark the status of each task in the form of: started, not started, finished; such that team members know which tasks should be worked on.

**Low priority:**

18. As a team member, I want to be able to search information regarding the projects by ID, responsible team members, deadlines, such that I can easily access specific information of a given topic, without having to go through the entire system.
19. As a project creator, I want to have the ability to change the roles of the team members, so they can work on more suitable tasks.
20. As a customer, I want the project's description, requirements and their status, displayed on a website, such that I can track its progress.
21. As a project creator, I want to be able to remove team members from a project, in case the team size needs to be adjusted.

## 2.2. Non-functional Requirements

22. The GUI should be implemented using Java/JavaFx.
23. The website has to function using Google Chrome (version 86.0.4240.193, release date 2020-11-10) Mozilla Firefox (version Firefox 82, release date 2020-10-20) Safari (version Safari 14.0, release date 2020-9-16).

## 2.3. Use Case Diagram

The use case diagram portrays the relationship between the system's use cases and their respective actors. This makes it easier for new users to understand the hierarchy of access within the project management system. The names of the use cases represent the user-goal for a particular actor. In other words, the shared purpose of a particular group of requirements. For instance, requirements 14, 17 and 19 all concern the updating of tasks. Therefore, their use case can be simply represented as 'Update Tasks'. The same principle goes for the rest of the use cases shown below.

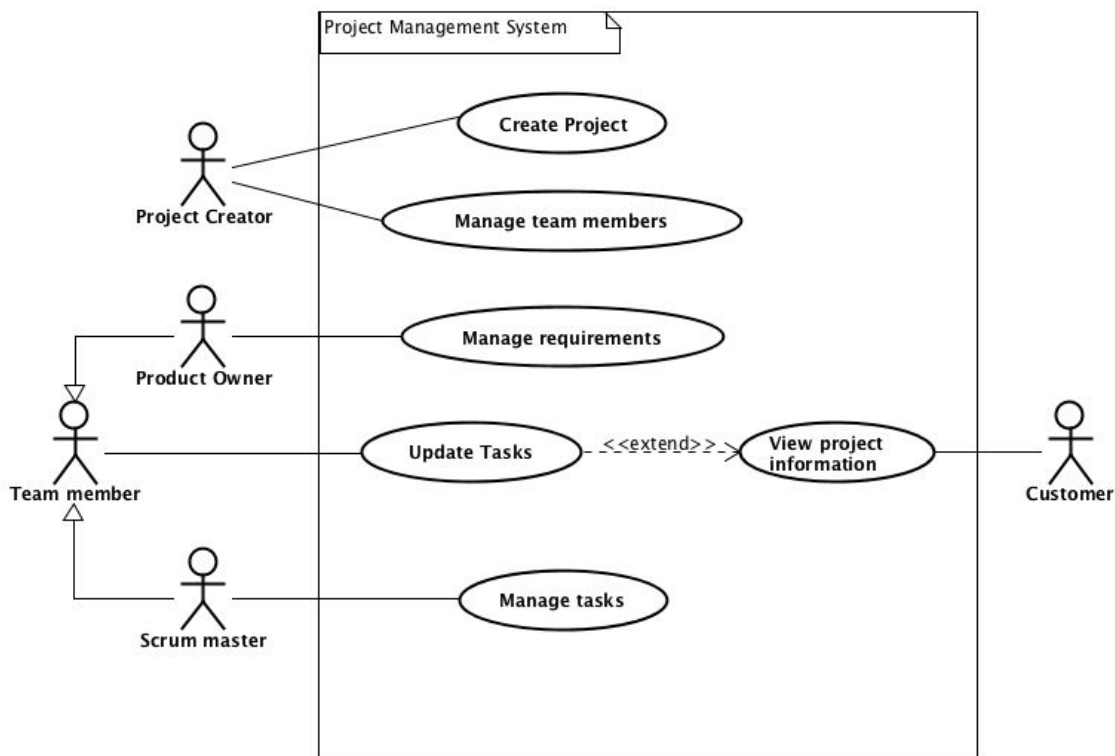


Fig. 1. Project management system - Use case diagram.



## 2.4. Use Case Descriptions

Following the formation of all requirements and use cases, a meticulous set of use-case descriptions was required. All the information one would need to know about a particular use case, such as preconditions and postconditions can be found in the use case descriptions. The base sequence is especially helpful because it establishes the step-by-step process a user of the system would go through to execute a particular function, as well as any possible exceptions and limitations. Much like the list of requirements, the use-case diagrams were a continuous work in progress. Initially, these were engineered based on how we envisioned the program to work. However, once the time came to implement it, the base sequence for each process was vastly different from the original. Although the intended users of the system would be trained beforehand, the use case descriptions help to accommodate new users and provide a form assistance for navigating the program. Many processes are only accessible by specific roles, for example only a project creator can create a project. For this reason, use-case descriptions help users of the system to distinguish which processes are relevant to them, thus preventing any confusion in regards to system access.

<b>Use case</b>	<b>Create Project</b>
<b>Summary</b>	Adding a new project to the system, with all relevant information(requirements, tasks, deadline, customer, etc.)
<b>Actor</b>	Project Creator
<b>Precondition</b>	Analysis should be accepted by the customer and all requirements and tasks should be specified
<b>Postcondition</b>	The new project is added to the system, and its information is saved to a file.
<b>Base sequence</b>	<ol style="list-style-type: none"> <li>1. Create a project</li> <li>2. Enter project title, customer name, project description and set a project deadline</li> <li>3. Save project</li> </ol>
<b>Exception sequence</b>	
<b>Note</b>	This use case covers requirements 1, 5

Fig. 2. Use case description table - Create project

<b>Use case</b>	<b>Manage requirements</b>
<b>Summary</b>	Add or remove requirements for a given project, changing their priority or responsible team member.
<b>Actor</b>	Product Owner
<b>Precondition</b>	A project should already be created in the system.
<b>Postcondition</b>	The requirements are now updated.
<b>Base sequence</b>	<ol style="list-style-type: none"> <li>1. Open Project. <ol style="list-style-type: none"> <li>a. Editing or removing a requirement go to step 6</li> </ol> </li> <li>2. Add Requirement.</li> <li>3. Give Requirement an estimated time, user story, deadline, priority, responsible team member. (edit requirement)</li> <li>4. Save requirement</li> <li>5. Go back to project <ol style="list-style-type: none"> <li>a. Adding a new requirement, go to step 2.</li> <li>b. Done with managing requirements go to step 12</li> </ol> </li> <li>6. Select the requirement to be edited. <ol style="list-style-type: none"> <li>a. If the requirement is to be removed go to step 11</li> </ol> </li> <li>7. Open requirement.</li> <li>8. Edit requirement information (priority, responsible team member, approve or reject)</li> <li>9. Save Requirement.</li> <li>10. Go to step 5</li> <li>11. Remove requirement <ol style="list-style-type: none"> <li>a. If more requirements needs to be removed go to step 6</li> </ol> </li> <li>12. Save Project.</li> </ol>
<b>Exception sequence</b>	
<b>Note</b>	This use case covers requirements 3, 6, 7, 8, 11, 12, 18

Fig. 3. Use case description table - Update tasks.

## 2.5. Activity Diagrams

Once use-case descriptions were finalized, activity diagrams were produced for each system process in order to demonstrate their logical operations. Activity diagrams provide a better way of visualizing a process as a whole. Instead of portraying a specific process in a linear fashion like a base sequence, they are represented as branching functions with multiple outcomes depending on what the user wants to do and in what order they do it. They are simple to follow and provide a means to evaluate all the possible outcomes of executing a function.

For example, the base sequence for the 'Create Project' use case involves three key steps: 1. 'Create a project' 2. 'Enter project information' 3. 'Save project'. This process is represented in the activity diagram directly below by three action nodes (Rectangle nodes). However between these action nodes are two decision nodes (Diamond-shaped nodes). In this particular example, they represent a decision, forming a loop. The condition being that the information entered must be correct. Therefore, if the information is wrong, the process must be repeated until the information entered is correct. This information would not be conveyed as effectively with use case descriptions alone. However, when combined with an activity diagram, the logical operation is clearly visible.

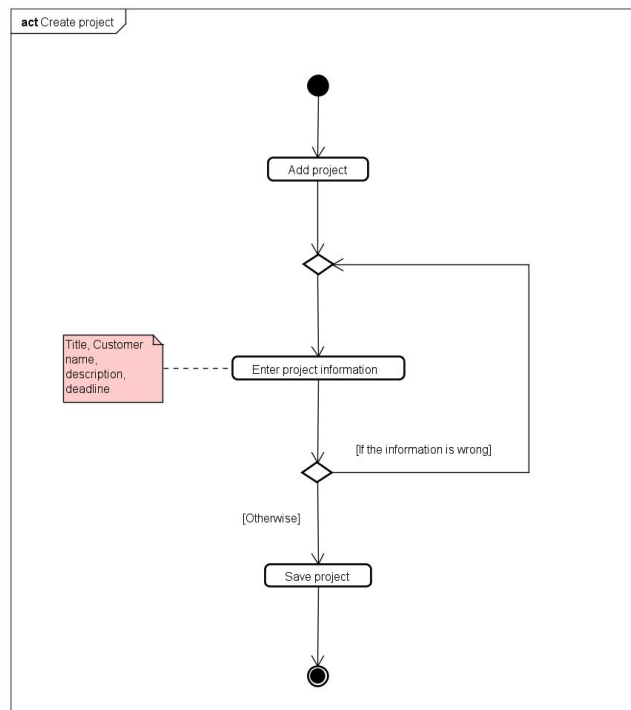


Fig. 4. Activity diagram - Create project

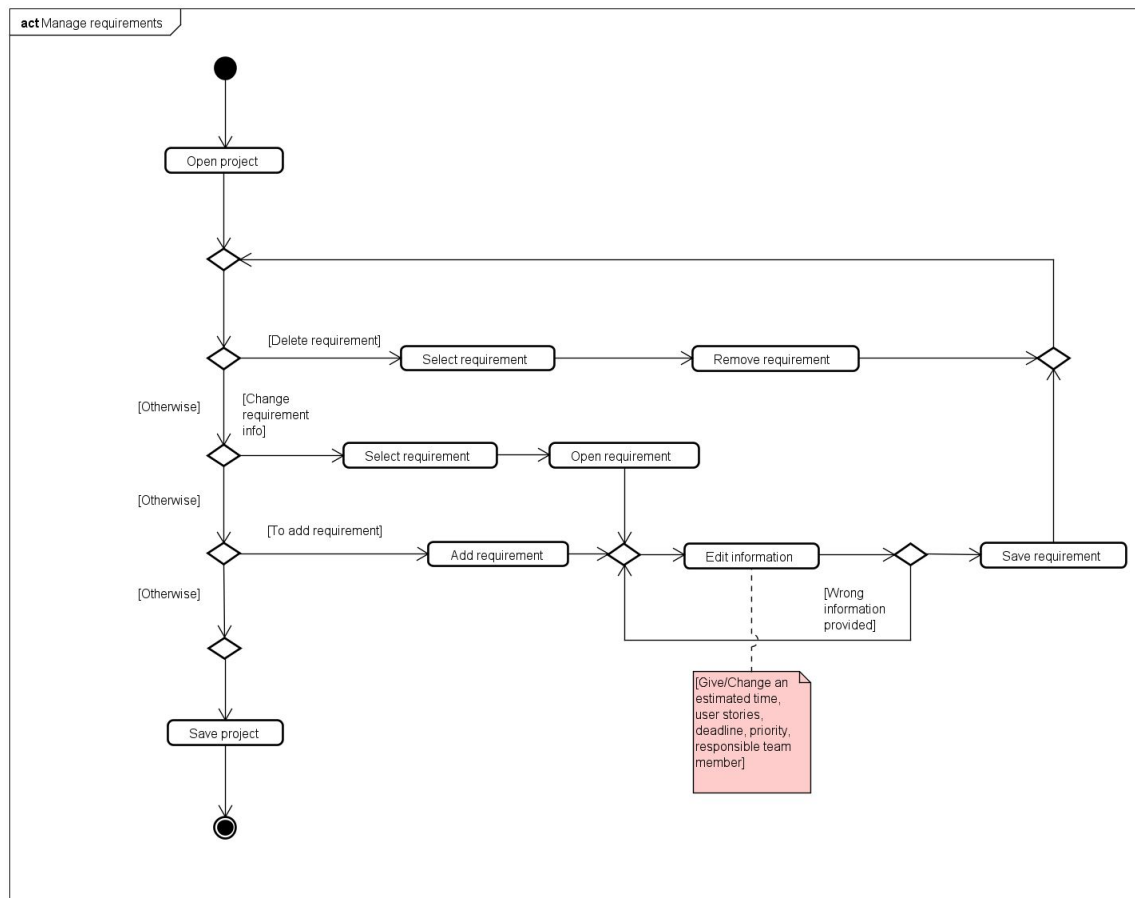


Fig. 5. Activity diagram - Manage requirements

In this activity diagram, a switch was employed to provide the user with multiple options - these options are called cases. Each new case is represented first by a decision node and they all lead back to the start of the switch. For this reason, these cases can be repeated as many times as the user wants considering they meet that case's conditions. For example, when a user wants to remove a requirement (Case A), after they have done so, they can choose to remove another requirement (Case A), change requirement info (Case B) or add a requirement (Case C). When the user is ready to save their project, they are essentially exiting the switch.

With closer examination of the activity diagram, one can see that when a user adds a requirement, the 'Add requirement' node leads to a merge node which is being intercepted by the 'Open requirement' node from Case B. The reason for this is that the user is editing the requirements information in both Case B and Case C. Therefore, only one 'Edit information' node is required as opposed to two. The merge node accomplishes this by merging the two cases into only one. Essentially, this makes the diagram simpler and helps prevent unnecessary repetition of nodes.

## 2.6. Domain Model

The final step in the analysis stage was creating a domain model. This establishes the relationship between the conceptual classes and knowledge of the problem, while showing the key concepts of the object behaviour (Philip Brown, 2014). This was challenging in its own right because classes had to be compiled into a single diagram in order to make sense as a union and come together in a cohesive manner. In the domain model shown in figure 6. *Domain model - Project management system* below, the reader is able to find the main objects that interact within the system, which helps in understanding and reaching a solution to a given problem. In the case of this report, the problem is an IT project management system to manage the projects and track the progress. In the model there are 9 conceptual classes or objects with the main one being - Project, which is standing in the middle of the model and which has a relationship to all the other objects shown in the diagram. The Project has a Date or deadline, it has a Customer who ordered the project, and a Team member who is working on the project. The project consists of Requirements, which in turn consist of Tasks. The Requirements also have Priority, that is to show which requirements are more critical for the project completion and need to be worked on first. Both tasks and requirements have Status, it tracks the progress of the project. By viewing this model, the user should have a better understanding of the description of the given problem.

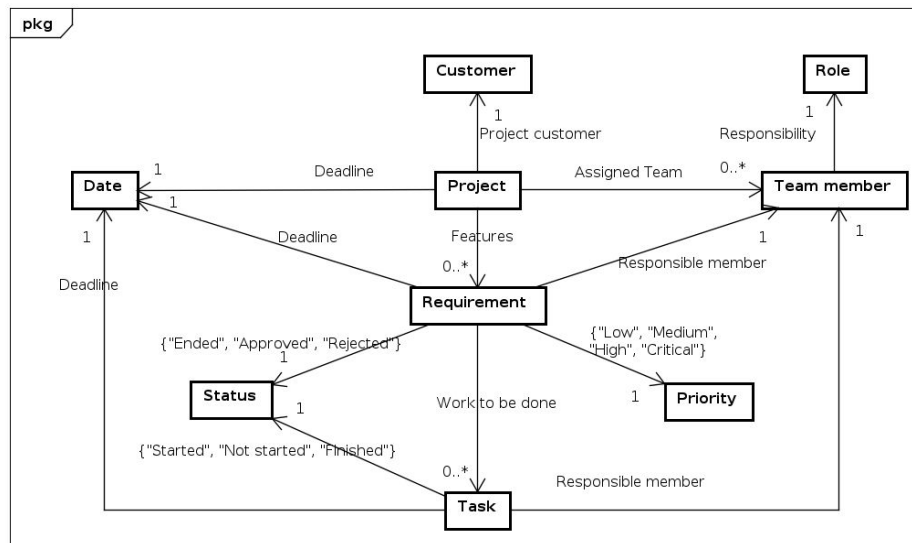


Fig. 6. Domain model - Project management system.

### 3. Design

Up until this point the development team has gathered and analyzed the data regarding the needs of the customer, that way getting an understanding of the requirements for the system and its functionality. The next step, which is a system design, was carried out in order to set the guidelines for the development team itself. Design acts as a set of instructions for the programmers, same way as the architectural design for the construction managers. Also, it is an effective way to communicate the product to both - customers and fellow employees. With the help of the design it is easier to split the work and envision what the product will look like in the future.

*“The purpose of the System Design process is to provide sufficient detailed data and information about the system and its system elements to enable the implementation consistent with architectural entities as defined in models and views of the system architecture.” (Odhiambo Didacus, 2018)*

The design of the current system consists of a Class diagram, Sequence diagram, and some of the GUI screenshots to grasp a deeper understanding of how the system will look like and how it is going to function.

When working with the diagrams and Graphical User Interface(GUI), the sketches of the system were brainstormed and used. In order to make the design presentable for the reader, the team required help from the software development tools, to be more specific, the programme called Astah were used. All the diagrams and models presented in this report were designed in the mentioned software modelling tool. While working on the systems GUI, the team has used yet another software development tool called Scene Builder. It made the work way easier, because the mentioned tool enables to design user interfaces without actual coding.

### 3.1. Class diagram

The Class diagram is the skeleton of the system - it is used to visualize and construct the system (Visual Paradigm, 2020). Here the reader will be able to see the Classes it consist of and the methods which are going to be implemented for each single Class. To make it more observable, the team has decided to split the Class diagram into several parts, this was done because of the quantity of the classes the team has agreed to implement for the full functionality of the system. The Class diagram was split into two parts, namely the model and the view part.

*(Note: in the figures below only the part of the both diagrams is visible, to access the full version of the Class diagrams see Appendix D)*

Following the waterfall approach in the beginning of the project, the development team quickly realized that it is not the best approach, when creating a functioning system. This was concluded when the team had to go back several times and re-model the Class diagram, because of the difficulty to get all the methods described in the older version of the Class diagram to be working properly. If the strict waterfall approach would be followed, there would be many flaws in the implementation phase, thus making it hard to achieve the desired result.

In the model part of the diagram, the reader is able to find the main classes of the system. This part is responsible for all the back-end operations and general functionality of the system. The figure 7. *Project management system - Model Class diagram* shows only a fraction of the implemented classes and methods. For example, in the figure the reader can see the Project class and its relation in regards to the other classes. The Project class has a customer, team member list, id, a date, and the requirement list. It also has a constructor called Project, which holds its own variables - title and description, variable from Customer class called customer, and variable from MyDate class called deadline. In addition, the Project class has composition type of relation to the Id class and also MyDate class, meaning that Project cannot exist without Id or deadline, and the Id and deadline cannot exist on its own. In this case both of the mentioned classes should have also had a .copy() method, so that information remains unchangeable.

Given the time pressure and constant re-modeling of the Class diagram, the development team did not manage to set the right relations between the classes, to be more precise, mention where are the Association, Aggregation or Composition to be found. However, it is something that the team is aware of and what could be changed in the future development of this project.

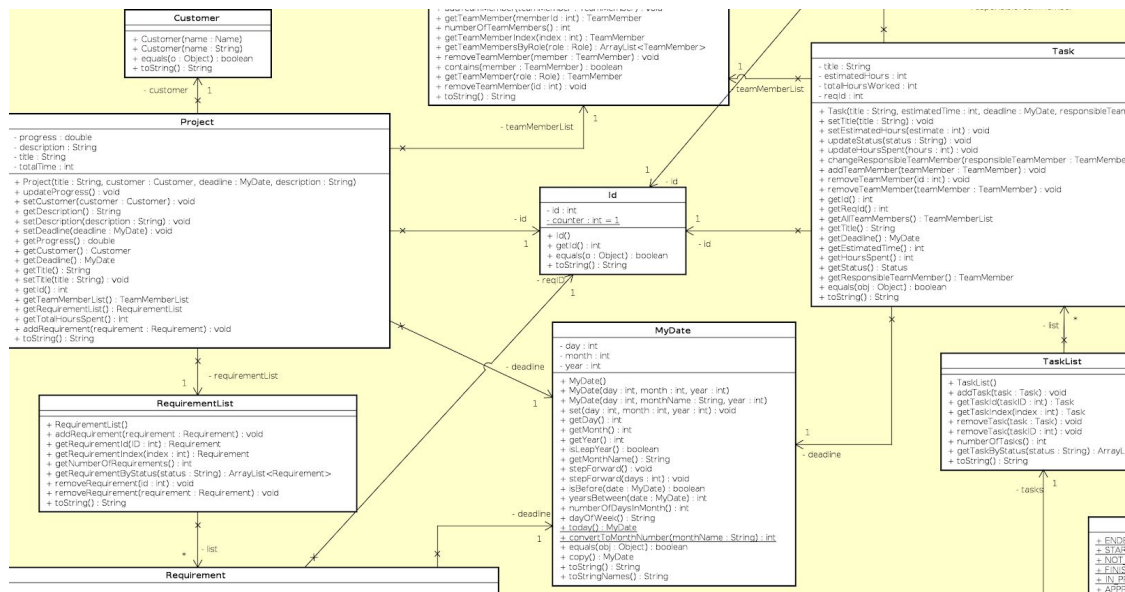


Fig. 7. Project management system - Model Class diagram.

In the figure 8. *Project management system - View Class diagram* below, there are four classes visible. Two of them are view controllers and the other two are view handler and view state. The 'ViewHandler' class is responsible for launching windows in the program. As a result, whenever a view controller needs to open a new view, it will need to access a method within the ViewHandler to do so. The 'ViewState' is responsible for how the system handles the states between the windows. Its main goal is to store the data and restore the results which impacted the user interface by user actions.

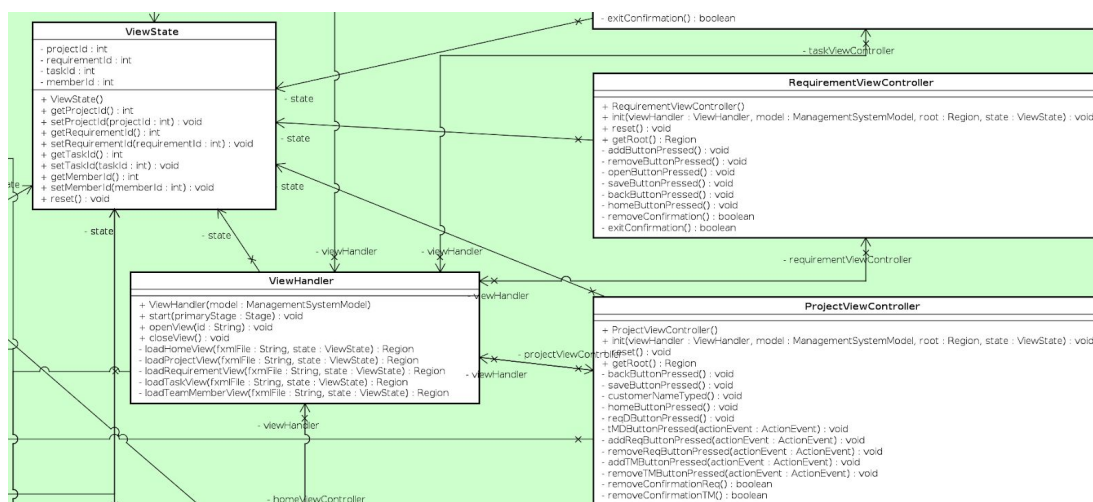


Fig. 8. Project management system - View Class diagram.



### 3.2. Sequence diagram

The sequence diagram was carried out for one of the more difficult methods by the opinion of the group. It depicts the sequence of events how the objects interact in order to achieve the desired functionality of the method to cover the system's requirement (Lucidchart, 2020). Especially useful for the programmers to understand the logic behind the given method.

The sequence diagram in figure 9 (*Note: for the bigger sized picture see Appendix E sequence diagram, or appendix Sequence diagram updateProjectProgress.svg file*) describes the process of updating a project's progress. First the updateProjectProgress is called inside the ProjectViewController, where it is passed an id of a project to update. Then the ManagementSystem class asks the ProjectList class to return the project with the given id, where after updateProgress is called on said project. Project loops through all related requirements and updates their status, if current status of a requirement is different from approved. A requirement has to check the status of all related tasks before it can update its own status. All related tasks are then looped through. If any of them has a different status than finished, the loop ends, which means the requirements status is in progress. In the other scenario where all tasks are finished, the requirements status will be changed to ended. Finally the project will then count the amount of requirements that has the status of ended or approved, and updates its progress as a percentage of requirements that are finished/approved.

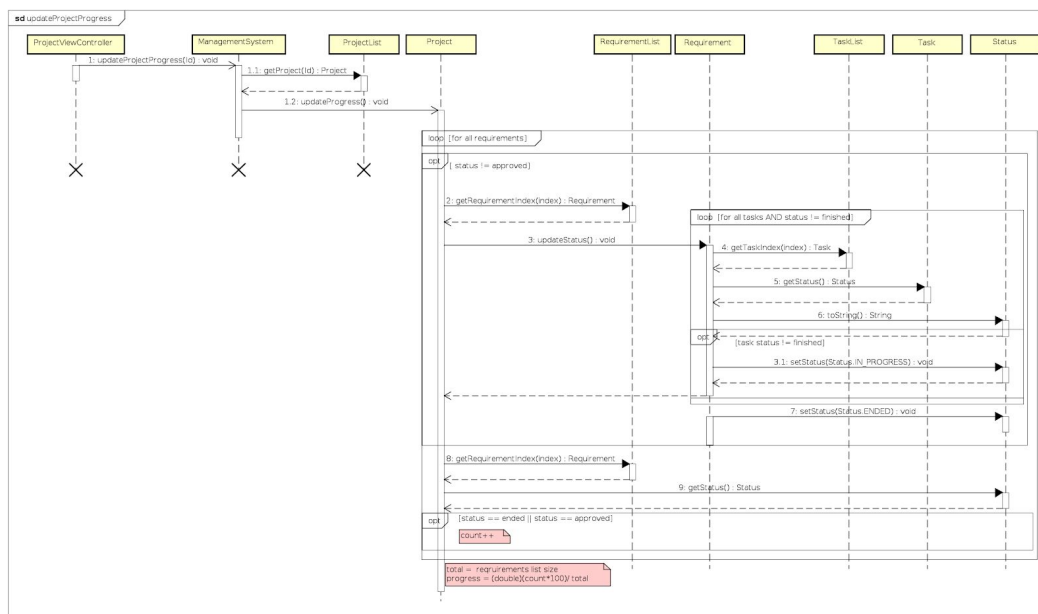


Fig. 9. *updateProjectProgress* method - Sequence diagram.

### 3.3. GUI views

The goal of the following section was to design and create a Graphical User Interface, which should include all the functionality for the system and at the same time should be easy to use. Focus is put here mostly on a visual aspect of the system. The key is that the navigation through the system should be understandable and intuitive, thus making the system user friendly. As mentioned in the introduction of this report, adoption to the technology by the employees, as well as the time spent training the employees to use the system can be very time consuming and costly. Therefore, a careful thought and considerations should be put behind the design of any system.

When planning on how the system would look, the development team went through a rather long process. Things to consider were the layout, the number of windows, if the views should include embedded menus or tabs, what happens when the displayed windows get resized, pop-up windows and more. It is important to mention that during the development process, the team had to cardinaly change the look of the system. It was done due to impracticality of the previously designed GUI. It consisted of twelve windows in total, which defeated the purpose of the system being user friendly and easy to navigate through. Having realized that during the implementation phase of the project, the team had to quickly change and readjust the GUI of the system. It resulted in the five different view windows and solved the impracticality issue. Further below the reader will be able to find the comparison between the two designs. (refer to the *Appendix B - Old GUI Sketch* to see the old design)

The following two figures are the screenshots of the new GUI design. In the figure 10. *Project - current system view*, the reader can see how the team decided to merge several windows together and make it one whole. In comparison to the old sketch, the newly designed view contains three windows of the old design in just one window, to be more precise, it is the list of requirements, the list of the team members and the project information itself. By making such a design the team got rid of the unnecessary amount of windows, thus making the system easier to navigate through and less confusing for the customer. Also, it minimized the view controller class count and allowed for the less written code. The layout of the current window is simple to understand and gives a good overview of the information like what project is being worked on, who is the customer, the description of the project, when is the deadline, what is the progress (given in percentage), total hours spent working on it and ID to be easily distinguishable among the other projects. In the same window on the right side, the user can see everyone who is working on the project by having an access to the team member list with a possibility to add, edit or remove the team members for the current project. In addition, all the requirements for the project are listed on the left side of the window with the same possibility to add, remove or edit the requirements. To access more information, the

user has a possibility to open the requirement by clicking on “Requirement details”, that in turn opens the next window shown in figure 11. *Requirement - current system view*, where he/she will be able to see and edit the list of tasks for a chosen requirement, as well as edit the requirement itself. This layout allows the user to have access to all the required information within just one click, whereas when compared to the old design, the user needed to go through multiple windows to find the same information, while forgetting what the information was related to.

The screenshot shows a web application interface for a project management system. The main heading is "Project". Below it, there are input fields for "Title:", "Customer:", "Description:", "Deadline:", "Progress:", "Hours spent:", and "ID:". The "Description" field is a larger text area. Below the input fields, there are two tables. The "Requirements" table has columns "Estimated time", "Used time", "Status", and "Deadline", and it displays "No content in table". The "Team members" table has columns "Name" and "Role", and it also displays "No content in table". At the bottom of the interface, there are several buttons: "Requirement details", "Add requirement", "Remove requirement", "Add Team member", "Edit team member", "Remove Team member", "Back", and "Save". An "errorLabel" is visible at the bottom center.

Fig. 10. Project - current system view.

The requirement window seen in figure 11. *Requirement - current system view* consists of two main sections, where one of them is the information about the given requirement in the upper side of the view, and all the related tasks for that requirement in the bottom side of the view. This again gives a quick access to a handful of information, which is concentrated in one place in comparison to the old design. Here the user can find all the data regarding the requirement like its ID, responsible team member, description of the given requirement, deadline, its status, total hours spent working on it, priority and estimated time for the completion of that specific requirement. Most of the information is also editable, except ID and hours spent (ID is assigned automatically by the system, and hours spent are calculated based on the time spent

working on tasks). As it was already mentioned above, at the bottom of the window the user is presented with the list of tasks, where the tasks can be added, removed or edited.

In addition, to make the system more user friendly, the development team decided to add pop-up windows. When there has been changes done to any of the information, and the user wishes to quit or go back without saving that information, the pop-up window comes up, which is asking for the confirmation of the chosen action. Also, when the wrong information is entered in any of the fields, the program gives an error and specifies what it is, that way ensuring the user to put the correct type of information in the specific fields.

The screenshot displays a web-based form titled "Requirement". The form is organized into two primary sections. The top section, labeled "Requirement", contains several input fields: "ID:" (a single-line text box), "Responsible:" (a single-line text box), "Description:" (a multi-line text area), "Deadline:" (a date picker), "Status:" (a dropdown menu), "Hours spent:" (a single-line text box), "Priority:" (a single-line text box), and "Estimate:" (a single-line text box). Below this section is a table titled "Tasks". The table has two columns, "Title" and "Status", and is currently empty, displaying the message "No content in table". At the bottom of the form, there is a row of buttons: "Add task", "Remove task", "Open", "Home", "Back", and "Save".

Fig. 11. Requirement - current system view.

### 3.4 Website

One of the requirements with a low priority was to make a website for the customers of the “Colour IT” with a possibility to be able to view the information regarding their ordered projects and track the progress. To be more precise, it was a requirement number 20 from the requirement list presented in the Analysis part of the project. Due to several reasons this task could not be accomplished. First and foremost, the development team has put their full focus on critical and high priority requirements, as without them there would be no chance to make the system at all. Second, followed by the time pressure to meet the deadline, the team needed to choose what is of most importance in the current project, which obviously was the IT management system and the reports. On top of all, the constant running into the problems while making the project and not receiving the full support from all the team members, all resulted in the website being left out of the current project.

## 4. Implementation

The following chapter shows how the above discussed system has been implemented. Here the reader will be able to find the path that the team has followed in order to make the system come to reality. To guide the reader through the implementation process, the development team will present and explain some of the most important code snippets below.

First the model was implemented, since the functionality of it is the foundation of the entire system. The model class diagram(*Appendix D.1. Model class diagram*) was used as a template, by exporting it as Java classes from Astah and importing them into an IntelliJ project. Implementation started at the “edges” by implementing simple classes, with few and simple methods. Like the Role class in figure 12

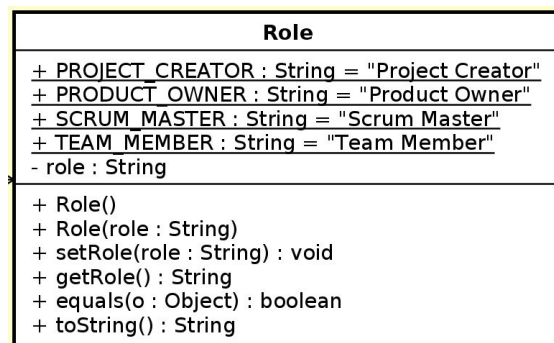


Fig. 12. Role - class diagram.

In the next phase where implementation began on some of the more complex classes, flaws in the design started to appear. A redesign of the class model diagram had to be made. When the changes had been implemented, work continued on more complex classes and methods. The last part of the model was implementing functionality of interface class, that handles communication between the view and model.

Implementation was approached by basing it on the model class diagram design, but quickly realized it was not very well designed, so a decision was made to finish implementing the view to figure out what methods actually would be needed. After the GUI was implemented, the next phase was dealing with implementation of functionality in view controllers.

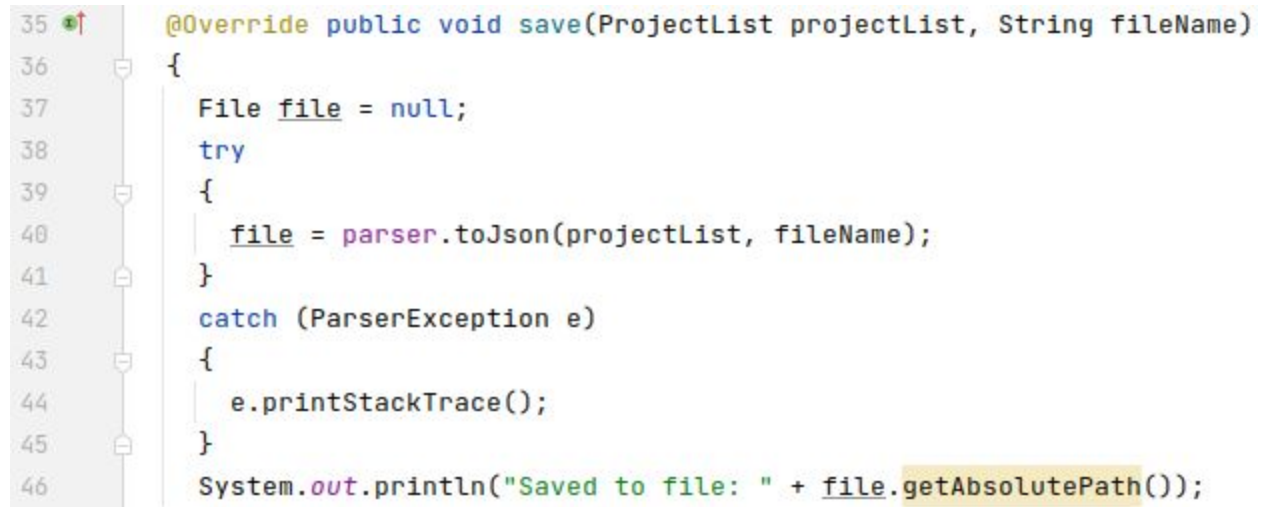
Finally, a way to store the systems information would be implemented. A choice of saving all information in the form of a .json file was made, based upon one of the requirements described a website, and importing information from a .json file with javascript would be easy. With a finished system here is an explanation of how one path through the system should function, in theory. A scenario where a user wants to add a team member to a task, and how the system handles this. User clicks add team member in task window, code checks if task is saved, if not user is told to save first.

```
@FXML private void addPressed() {  
    if (state.getTaskId() < 0) {  
        errorLabel.setText("Save task before adding team members");  
    }  
    else {  
        state.setMemberId(-1);  
        viewHandler.openView( id: "teamMember");  
    }  
}
```

Fig. 13. Event handling when add button is pressed

When a task is saved, the viewhandler is told to open the team member window, where the initialization of the window changes depending if it was accessed from a task or project. Team member view controller, checks what team members are assigned to the related project, while at the same time excluding team members already assigned to the

task, and adds them to an observable list. The list is then assigned to the drop down selection, where the user can choose who to add to the task. When the choice is made, the user clicks save. Code then adds the selected team member to the tasks team member list. Task view is now loaded by the viewhandler and the team member list is updated, with the newly added team member. User clicks on save, code then saves all information from the system to a json file.

A screenshot of a code editor showing a Java method named `save`. The method signature is `@Override public void save(ProjectList projectList, String fileName)`. The code is as follows:

```
35  *  
36  {  
37      File file = null;  
38      try  
39      {  
40          file = parser.toJson(projectList, fileName);  
41      }  
42      catch (ParserException e)  
43      {  
44          e.printStackTrace();  
45      }  
46      System.out.println("Saved to file: " + file.getAbsolutePath());
```

The code is highlighted with a light blue background. The line numbers 35 through 46 are visible on the left side of the editor.

Fig. 14. Saving to a json file

This scenario and many others will be touched upon in the next section, where the system will be tested, to make sure everything is functioning as expected.

## 5. Test

After the implementation of the system, the team needed to make sure that everything is working and all the requirements are met. This was being done by making J-Unit tests for some of the class methods, as well as, testing the use cases. J-Unit tests are made to test the functionality of the method, to be more precise, if the method is doing what was intended by the programmer. Testing use cases on the other hand, are useful for broader testing of the system - it covers step by step activities from start to finish of a specific action and tests if the system functions as requested by the customer.

## 5. 1. J-Unit test

Ideally, J-Unit tests are done for all the methods, as it helps to identify the exact problem within the method if there are any and gives the opportunity to fix it. However, for this exact project the development team has only done a few J-Unit tests to present the newly acquired skill during the semester of studies. In addition, given that the coding program automatically checks for the errors itself, and the development team is always testing the system on the go, there is a slim chance of having a major error. Besides that, the mentioned errors could be fixed during the Maintenance stage of the system development, which the development team will go through in their studies in the near future. Below the reader will be able to find some of the J-Unit test screenshots followed by a short description and explanation.

```
class IdTest
{
    Id id = new Id();
    Id id1 = new Id();
    Id id2 = new Id();

    @Test void getId()
    {
        assertEquals( expected: 1, id.getId());
        assertEquals( expected: 2, id1.getId());
        assertEquals( expected: 3, id2.getId());
    }
}
```

Fig. 15. J-Unit test for the method getId()

In the figure 12. *J-Unit test for the method getId()* above, the reader can see a J-Unit test done to check if the method, which was created to automatically assign the ID to the projects, requirements and/or tasks is functioning as intended by the development team. The intention was to automatically assign an integer, based on the order in which the ID was created (If it was the first ID, then the ID of it would be number 1, If it was the 206. ID, then the ID of it would be number 206). In the test there were 3 IDs created, therefore expected results would be 1, 2, and 3. In case of not correct implementation of the code, the numbers would not match, which in turn would specify the incorrect test case. This enables the programmers to go back and check the mistake and find different solutions for the previously implemented code.



In the figure 13. *J-Unit test for the method stepForward()* below, the reader is able to see if the date goes up one day to the future. As in the example, if it is the last day of the month, by calling the stepForward method it should set the day to 1, set the number of the next month, as well as, set the number of the next year (if the date was 31/12/2020, then the new date should result in 1/1/2021). In this case the development team has written the code correctly, however if the opposite would be true, the date could result in 32/13/2021, which is impossible given that there are 31 days in the month of December and there are 12 month in a year. If any of the expected numbers in the example would change, the test would fail, while also printing out the message “Not correct ...” depending in which case the mistake was found.

```
@Test void set()
{
    date.set( day: 1, month: 3, year: 2020);

    assertEquals( expected: 1, date.getDay(), message: "No correct day");
    assertEquals( expected: 3, date.getMonth(), message: "No correct month");
    assertEquals( expected: 2020, date.getYear(), message: "No correct year");
}
```

Fig. 16. J-Unit test for the method stepForward()

## 5. 2. Use Case test

Use case tests were performed to evaluate whether our functionality works as the use cases intended. These are assessed with a simple PASS or FAIL depending on whether they were accomplished with success. The program is started and a particular use case is attempted, afterwhich observations of the functionality and what order is required are made. These observations are then compared side by side with the expected results to determine whether they do in fact match the desired result. If they do, they are marked as ‘PASS’ and if they are not, they are marked as ‘FAIL’.

After a use case test has been performed for the main scenario of the use case, another one is performed for the exception scenario. In other words, the outcome of an incorrect execution. The assessment is the same as with the main scenario, either a ‘PASS’ is awarded or a ‘FAIL’ depending on whether it meets the expected criteria.

Use case under test	Use case "Create project"			
Scenario	Main scenario			
Precondition	System should be running			
Step	Action	Expected result/observation	Actual result/observation	Assessment(PASS/FAIL)
1	Click on add project	New project window will open	New project windows opens	PASS
2	Enter project title, customer name, project description and set a project deadline, then save project	Information will be saved, the project window should now be updated with entered information. The project should also have been given an id, progress off 0% and 0 hours spent	After entering information and saving, the window is updated with entered information and progress is 0%, hours spent 0 and id is updated	PASS

Fig. 17. Use case test (Main scenario) - Create project

In figure 14, a use case test for the 'Create project' use case has been performed. For step 1, the observed results match the desired results and so, a 'PASS' is awarded. The same can be said for step 2, meaning this use case functions as intended.

Use case under test	Use case "Create project"			
Scenario	Exception scenario			
Precondition	System should be running			
Step	Action	Expected result/observation	Actual result/observation	Assessment(PASS/FAIL)
1	Click on add project	New project window will open	New project windows opens	PASS
2	Enter customer	Error will be displayed, explaining	Nothing happens	FAIL

	name and nothing else	what information is missing		
--	-----------------------	-----------------------------	--	--

Fig. 18. Use case test (Exception scenario) - Create project

In figure 15, a use case test for the exception scenario of the 'Create project' use case has been performed. For step 1, the observed results match the desired results and so, a 'PASS' is awarded. However, the observed results for step 2 differ greatly from that of the expected result and so, step 2 is assessed as a 'FAIL'.

Use case under test	Use case "Manage requirements"			
Scenario	Main scenario			
Precondition	System should be running, and a project should already be created in the system. The project in question should have at least one team member assigned			
Step	Action	Expected result/observation	Actual result/observation	Assessment(PASS/FAIL)
1	Select the project and open it	Project information will be displayed	Project window opens and information is displayed	PASS
2	Click on add requirement	New requirement window will be displayed	New requirement window is opened	PASS
3	Enter estimated time, user story, deadline, priority, responsible team member and save the requirement	Requirement window will be update with entered information, and will be given a unique id	Requirement window is updated, but window title is still "New Requirement"	FAIL
4	Click on back and choose ok in confirmation box	Project window will open, the newly added requirement will be displayed in the requirement list.	Project window opens, and the requirement is listed	PASS
5	Select the requirement and click on	Requirement window will open and show information from selected requirement	Requirement window opens and displays information of	PASS

	requirement details		selected requirement	
6	Edit estimated time, user story, priority, responsible team member and save the requirement	Requirement window will be update with entered information	Requirement window is updated with edited information	PASS
7	Click on back and choose ok in confirmation box	Project window will open, the requirement will be displayed in the requirement list, with the edited information	Project window opens, and the requirement is listed with edited changes displayed	PASS
8	Select requirement and remove it	Confirmation box will appear and ask if you want to remove selected requirement	Confirmation box appears	PASS
9	Choose Ok	Selected requirement will now be removed from the projects requirement list	Requirement is removed from the list	PASS

Fig. 19. Use case test (Main scenario) - Manage requirements

In figure 16, a use case test was performed for the 'Manage requirements' use case. The observed results of steps 1,2,3,4,5,6,7,8 and 9 satisfy their respective expected results, and are therefore marked as 'PASS'. However, the observed result for step 3, although very similar to their expected result, is not sufficient to be considered successful, and is therefore assessed as a 'FAIL'.

Use case under test	Use case "Manage requirements"
Scenario	Exception scenario
Precondition	System should be running, and a project should already be created in the system, with at least one requirement. The project in question should have at least one team member assigned

Step	Action	Expected result/observation	Actual result/observation	Assessment(PASS/FAIL)
1	Select the project and open it	Project information will be displayed	Project window opens and information is displayed	PASS
2	Select a requirement and click on details	Requirement window will be displayed with information from selected requirement	Requirement window is opened and displays information about selected requirement	PASS
3	Enter characters into the estimate time box	Information will not be saved and an error text will be displayed with information about what is wrong	No information is save, error text is displayed, but it does not explain in an understandable way what went wrong	FAIL

Fig. 20. Use case test (Exception scenario) - Manage requirements

In figure 17, a use case test for the exception scenario of the 'Manage requirements' use case has been performed. For the first two steps, the observed results match the desired results and so, a 'PASS' is awarded to both of them. However, the observed results for step 3 differ greatly from that of the expected result and so, step 3 is assessed as a 'FAIL'.

## 6. Results and discussion

The goal of this project was to develop a project management system which enables its users to create projects, manage various requirements, tasks and their team members. This was clearly achieved as the final product allows users to do all of those things with all the detail one would expect from a standard management system. It is also easy to use and lets its users work collaboratively without any complications.

As the use case tests show, the basic functionality is sound. Furthermore, the system aids its users by informing them when they have skipped a particular step or when they are missing required inputs.

However, the biggest shortcoming of the project was undoubtedly the lack of a website for displaying the project information. As such, customers would not have a way of directly checking their project's progress themselves.

Furthermore, the option to search for projects by their unique ID was never finalized and was ultimately discarded. This would have made it easier for users with a large number of projects to search for particular projects quickly.

Nevertheless, the program functions as it was expected and has a standard of quality consistent with what we envisioned.

## 7. Conclusions

All aspects and stages of the project were challenging in their own rights. However, the success of the project is undoubtedly attributed to the meticulous planning that went into it. This can be seen most prominently in the Analysis and Design stages. These were the stages that presented the most problems because they required a considerable amount of foresight. Subsequently, the slightest mistake could cause problems down the line. As a result, a great deal of care went into these stages. They were revisited continuously, with more and more corrections and additions being made the further development progressed.

Structurally speaking, the design was solid from the beginning, with only minor changes being made towards the end. This careful planning facilitated implementation considerably because there was no need to go back and redo any aspects of the fundamental design. Ultimately, this led to there being more time for ensuring the program worked as intended.

Compromises were made in the final stages of the project in regards to the final look of the program, with some features being stripped down for better functional quality. However, none of these changes affected the final product in any negative way. Therefore, they should be seen as improvements rather than downgrades.

When assessing the project as a whole, it is evident that the goals which were set out were achieved with consistency.

## 8. Project Future

There are a variety of which would have improved the systems functionality and final look. However, due to several factors, primarily a lack of time, these ideas were inevitably discarded. The majority of these are non-functional requirements, as such, they were of low priority and ultimately overlooked.

One of these regards how to system handles exceptions. Initially, methods were meant to test and throw exceptions based on if input is wrong. This was worked on for a brief amount of time before work was directed to more important functionality.

Another possible feature would have been to save the date on which a project and its requirements and tasks is created and started. Technically speaking, this is easy to achieve but like most the features in this section, they were only thought of in the very final stages of development. Therefore, it was simply never attempted.

Additionally, an option to backup project files was another possible feature. Essentially, whenever a new project was created, a copy of that project's file is saved in another location so in the case of missing or accidentally deleted projects, a project can be easily recovered.

To improve the structure of projects, it would have been helpful if requirements, tasks and team members had their own files as opposed to having all of a project's information on a single file.

Due to a lack of time, the website for displaying project information to customers was not made. Had there been more time, a website with all the appropriate information would have been made. This would have accessed information stored on the project's file and displayed it in the form of an interactive table.

Lastly, the program would have benefited from an option to search for projects by their unique ID. This was not possible within the time provided because of the way the system handles new states.

## 9. Sources of information

1. Mr. Colour, September 2020. *Colour IT–The interview*. [pdf] Available at: <<https://via.itslearning.com/ContentArea/ContentArea.aspx?LocationID=14220&LocationType=1>> [Accessed 9 December 2020]
2. Kashyap Vartika, 2020. *Project Management System: Definition & Features*. [online] <<https://www.proofhub.com/articles/project-management-system>> [Accessed 9 December 2020]
3. Mavenlink, 2020. *What is project management software?* [online] <<https://www.mavenlink.com/resources/what-is-project-management-software>> [Accessed 9 December 2020]
4. Odhiambo Didacus, 2018. *System Design in Software Development*. [online] <<https://medium.com/the-andela-way/system-design-in-software-development-f360ce6fcb9>> [Accessed 12 December 2020]
5. Visual Paradigm, 2020. *UML Class diagram tutorial*. [online] <<https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/>> [Accessed 12 December 2020]
6. Lucidchart, 2020. *UML Sequence diagram tutorial*. [online] <<https://www.lucidchart.com/pages/uml-sequence-diagram#:~:text=A%20sequence%20diagram%20is%20a,to%20document%20an%20existing%20process>> [Accessed 12 December 2020]
7. Philip Brown, 2014. *What is the Domain Model in Domain Driven Design?* [online] <<https://culttt.com/2014/11/12/domain-model-domain-driven-design/>> [Accessed 17 December 2020]



# IT Project Management System

## Process Report

Group 4

Group Members: Henrik Koster (305916), Kim Tranberg (172394),  
Aleksandrs Bistrovs (304542)

Supervisors: Astrid Hanghøj, Steffen Vissing Andersen  
VIA University college

Character Count:

Software engineering  
Semester 1

Date: 17/11/2020

<b>1. Introduction</b>	<b>2</b>
<b>2. Group description</b>	<b>3</b>
<b>3. Project initiation</b>	<b>5</b>
<b>4. Project description</b>	<b>6</b>
<b>5. Project execution</b>	<b>7</b>
<b>6. Personal reflections</b>	<b>8</b>
Aleksandrs	8
Henrik	11
Kim	12
<b>7. Supervision</b>	<b>13</b>
<b>8. Conclusions</b>	<b>14</b>
<b>9. Sources of information</b>	<b>15</b>

# 1. Introduction

The project is composed of four primary stages: Project Description, Analysis, Design and Implementation.

This stage provides the foundation for the rest of the project since it entails the description of the problem and how we will go about solving it. We tried to make this as clear and concise as possible while still describing all the critical aspects of the project in the form of a background description, definition of purpose, problem statement and methodology. Due to the late formation of our group, we only had a few days to complete the project description. However, we were able to use our work from our previous groups and formulate that in our own words. For this reason, we were able to complete the project description in no more than two days.

Of all the project's stages, the analysis was the most time consuming, taking a total of 22 days to complete, making up roughly 22% of the entire project duration. This is largely in part due to the substantial amount of planning required in the form of: functional and non-functional requirements, a use case diagram, use case descriptions, activity diagrams and a domain model. Over this period, we had several meetings to discuss who was working on which aspects of the analysis. The most important meeting being after our first hand-in as we received a considerable amount of constructive feedback from both our supervisor and partner group. Our last meetings involved compiling all our work into one clear document ready for hand-in.

The design stage was the shortest stage, taking only 11 days to complete and making up roughly 11% of the entire project duration. Nevertheless, it was critical that we did this as accurately as possible since a solid design will always make implementation easier and vice versa. Design involved the formation of classes and sequence diagrams. Over this period, we had several meetings in which we worked collaboratively. Unlike the previous stages where we could work on separate aspects in our own time, it was critical that we formulated the classes and sequence diagrams together as these are all interlinked in some way and the slightest mistake could cause problems down the line. This helped us to avoid confusion and ensure all our work was consistent and met a high standard.

The Implementation stage took 17 days to complete, making up roughly 17% of the entire project duration. The majority of this stage was done remotely and meetings were carried out on a daily basis. We used Git and Github because it allowed us to work on our program in unison without the hassle of continuously merging our work which would inevitably cause a variety of problems. In hindsight, this was a great decision because it allowed for a more professional workflow. Had we done this without Git, we would probably spend as much time fixing errors as we did coding.

The project as a whole had a total duration of 3 months and 5 days and throughout this time, our dedication to consistent work as well as our commitment to meeting regularly helped us produce a functional program with very few complications and disagreements along the way.

## 2. Group description

Our current group consists of three members: Aleksandrs, Henrik and Kim.

Aleksandrs is 28 years old, who comes from Latvia. Previously has been studying International Hospitality management in Odense, where he gained some experience in project writing. However, since it is a very different field, the previously gained experience did not prepare him for the requirements here.

I am Henrik and I am 19 years old. I was born in Portugal in January 2001. I have Swedish and Portugese nationality. I studied a general engineering course in Loughborough College, United Kingdom. In my last year, me and 3 other classmates designed a bike trailer from scratch. As such, we were required to document the whole process from start to finish, including research. For this reason, I already have experience writing project and process reports.

Kim is 30 years old, of Danish nationality. Has previously been studying in two different higher educations, but did not complete any of them. He has experience with problem based learning, and project work in general, since highschool.

In the very beginning the team consisted of four members, namely - Aleksandrs, Kim, Henrik and Laurentiu. Later on, that number has been reduced to three, since one of the students decided to stop the programme due to personal reasons. The current group - Aleksandrs, Kim and Henrik come from different cultures, have different backgrounds and nationalities, which are all contributing factors to the work as a group. Therefore, it is important to take the previously mentioned factors into consideration when reflecting upon the period of the project. To help understand the differences and provide the reader with the background of each student, the aid of different cultural models, and personality profiles are discussed further below.

The cultural theories like Hofstede and Hall's theory helped our team to realize and take into account our cultural differences. Coming from different places, we have noticed that we have different ways of thinking, preferred ways of working, and it also affects the way we communicate with each other. For example, if we take a look into a Hall's theory about low and high-context cultures, we can see that low-context cultures are very different from high-context cultures. In our team we have

experienced a slight problem, where the members had a conflict due to different ways of communicating. To be more precise, in the very beginning, when we were still four members in the group - one of the members from a high-context culture (Romania) had a hard time accepting the critique, which was given by a member from a low-context culture (Denmark), where “saying what you mean and mean what you say” is a preferred way of communicating, rather than being indirect in high-context cultures. Becoming aware of this, we learned that we may run into such problems in the future, and there is no point in getting angry with each other.

*(Note: this paragraph was written during the SSE classes)*

Looking at the cultural model of Hofstede, to be more specific, to the dimension of Individualism (see figure 1. *Geert Hofstede cultural model - country comparison*), we noticed that some of the group members were more able to work on their own after splitting up the tasks (members with the highest scores in the mentioned category), whereas some of the members still required some sort of guidance in a form of collective thinking - no work could be done without discussing it first with the other members first. (members who score low in the mentioned category). Furthermore, looking to the rest of the dimensions in the model, we can see that some cultures are more similar than others. For example, it is clearly visible that Romanian and Portuguese culture is more alike, whereas Danish and Latvian culture also have more in common than the first two. This could also be felt in the group, members of these countries were better at cooperating with each other.

*(Note: this paragraph was written during the SSE classes - modified and added information)*

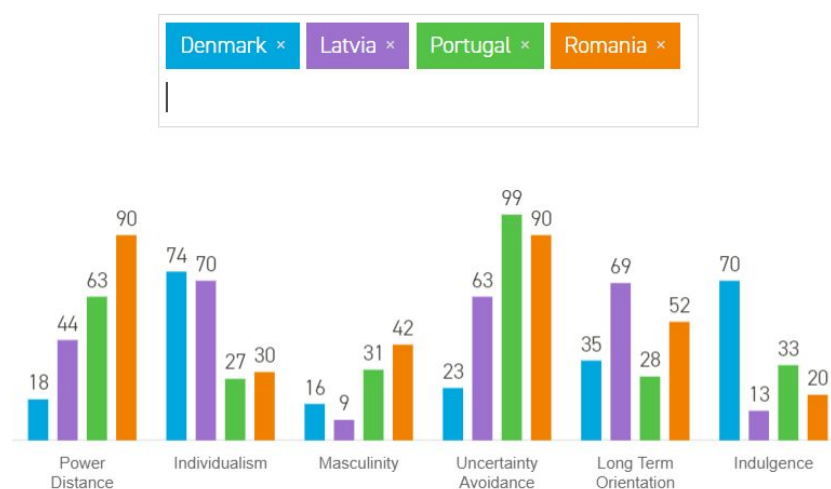


Fig. 1. Geert Hofstede cultural model - country comparison (Hofstede Insights, 2020)

Something important to consider was also our personalities. During the semester, we did personality tests in the lessons of SSE. The profile test divided the respondents into 4 categories with a different colours: red, yellow, green and blue. Each colour

represents different characteristics, for example, red is more of a leader, yellow is creative and social, green is more of a friendly type, and blue is detail oriented (E-stimate Personal Profile, 2019). Based on our results from the mentioned test, our personalities match up very well. In general we all have a mix of the same colours, mostly blue, green and a bit of yellow. This means that we can come to an agreement for all of our discussions without them reaching a personal level, and turn into a serious conflict. A downside to our group composition is that our discussions can sometimes take longer than needed, since we are lacking a bit of the red colour, which is more of a leading type and would try to reach an agreement way faster.

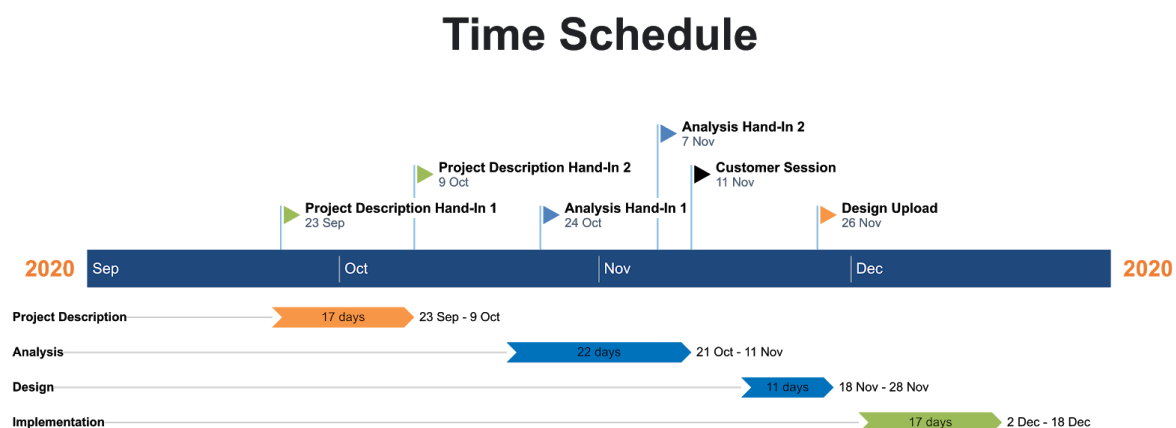
*(Note: this paragraph was written during the SSE classes - modified and added information)*

### 3. Project initiation

In the beginning of the semester, the students were given a task to design and implement a single user system, based on the interview with "Colour IT" given by the supervisors. During the SEP (Semester project) lessons, the students were learning step by step how to analyze such an interview, how to make a list of requirements using the SMART model, how make a Use Case diagram using the software called Astah (diagram modelling program) and how to write Use Case descriptions. Later on, we were introduced to the design part of the project, where students learned how to make Activity diagrams, as well as the Domain model using already previously mentioned software - Astah. Combined with the skills from other subjects, the students could slowly begin the process of working in their study groups while still learning the above mentioned topics.

The group has formed rather late during the semester, that is due to the fact that all of the team members have experienced different problems in the previous groups they were members of. For some members it was the difficulty communicating to their former group, for some it was lack of motivation due to the dysfunctionality and lack of participation of their former group in the lessons. That led to solving the problem and seeking out a new group. When finally the current group has formed (around the beginning of November), the project began and slowly gained pace towards the end of the month. The positive outcome of changing the group has allowed us to quickly adapt and begin performing, thus skipping the norming and storming stages described in the group dynamics model by Bruce Tuckman (MindTools, 2020). Because of the similar beginning phase of the group members and the common goal followed by the time pressure, the team could quickly accept each other, establish norms and harmony, and skip the conflicts.

In order to accomplish all the tasks and manage to be on time for the deadline, the team had to plan ahead. In the norming stage of the group, the team has made a contract (see *Appendix A - Group Contract*), where we had to sit down and discuss all the issues we might encounter in the future and how to deal with them, in the contract we also mentioned how invested are we in the topic and how much time we are willing to spend for the project, as well as the general guidelines for showing up on meetings and respecting each others time and deadlines. Here the team also used a tool called Gantt chart, which illustrates the schedule of the project. It is a great tool to divide the project into easily viewable sections, to which the approximate time is assigned, that way helping the user of such a tool to do the project on time or meeting the deadline.



Total Duration: 3 months and 5 days

Fig. 2. Gantt chart time schedule.

## 4. Project description

Before starting any project, the team had to make sure that the problem is clearly understood and defined, that is where the project description came in handy. Because the current team formed later in the semester, the project description phase was written twice - once in the previous groups, and the second time in their current group. It cost time and slowed down the process of getting to an actual project

writing, but enabled the team to revise and think about the problem more, thus making it more clear.

The problem was defined by sitting down with the team and analyzing the interview, while also thinking in the broader perspective. Here the team needed to understand why would companies have such needs in general and how it would better their efficiency at the workplace. These questions kickstarted the research and enabled the team to become more knowledgeable about the given topic.

The goal which was set during the project description resulted in the ready product followed by the report, where the product was an IT management system and the report describing it. It was a challenge to make it reality due to several reasons: one of them being the lack of knowledge and expertise in the field, since we are the first semester students. We spent way too much time on things, which would not take as long given more experience. Second, the lack of time - uncertainty followed by the time pressure resulted in quick and heedless decisions, which in turn made it hard to achieve the intended results. One of the features that the system should have included was the website. Unfortunately, by spending too much time on all the other aspects of the project, the team did not have time to implement this feature.

## 5. Project execution

The main plan for our project was mostly pre determined by the different deadlines set by our supervisors. We did not follow a time schedule or really utilize the one that was made.

We mainly used a waterfall approach, which was really great for a first semester project, since we had to learn all the different stages of developing a system, in chronological order. If we had to use an iterative approach, we would first have to know all the stages of system development. Since in an iterative approach you look at one requirement at a time and do all the steps for that one requirement, e.g. analysis, design, implementation and testing. However, at the same time the waterfall approach was not that great for us, since we did not have much previous experience with system development. In the end, this meant that when implementation was done, we had to go back and redo previous steps, like analysis and design. This was also a reason for our poor time utilization in the two project weeks. But for next semester an iterative approach will be much more relevant, since we now have more experience with the process of developing a system.



We did not really have a thorough plan for our project, so we could not follow up on it.

The main goal of the project was to design and develop an IT project management system, which was successfully achieved. However, given the time limitation, the team was not able to “polish” the system, to be more precise, to make it perfect by achieving the full potential with all the functionality included. In addition, the team did not manage to make a website for the system. Despite it, we are still satisfied with the results we got, as in the beginning of this semester nobody thought it was possible.

The main risks we foresaw were not meeting deadlines, unequal workload and crunch time. Fortunately, we were able to complete all our work on time. This was due in part to our regular meetings to ensure all the work was completed before handin dates. However, we didn't have any measures in place for unequal workload, this led to some people doing more than others in particular stages, although this was most prominent during implementation. Lastly, the only crunch time we experienced was towards the very end when we were working on project and process reports. This could have been avoided if we simply started documentation earlier. Better yet, we could have worked on the project and the report simultaneously, which would have allowed for more time to polish the final product itself.

Organization was poor from the very start of the project. We did not assign roles to tasks and as such, we were not able to coordinate who was working on what. Although, this did not prevent us from completing the necessary tasks, the work could have been carried out more effectively if more planning went into distributing the workload. Our organization improved towards the end of development as we realized how important it was to achieving a structured workflow.

## 6. Personal reflections

### Aleksandrs

Our group had very different levels of understanding, as well as, the skills. I feel like everyone was doing whatever they were best at. Due to my lack of expertise in programming, I felt like I could not contribute as much as I wanted to during the Implementation phase of the project, because of the mentioned differences above. Some of the group members had more skill in programming, therefore it was a little difficult to work together in some of the parts of the project. As a specific example I can mention that for me it was easier to write both reports, while for our team member Kim it was easier to code. I also felt different levels of engagement and

overall time investment in the project. In my opinion, the age difference could play here a role too, since it was clearly visible that the older team members were more serious from the very beginning of the project. Also, we had very different workflows, while some members tried to work consistently throughout the time period, some members contributed a little more right before the deadline of the project. Overall, the time investment and general contribution to the project by the team members were satisfactory, however, there were several times when it felt like there could be some more involvement and time contribution to the project.

Motivation was fluctuating throughout the process of the semester project. While in the beginning it felt like the group was very motivated, because of the previous failures in the former groups, then in the middle and closer to the end motivation slowly faded away. In the beginning the members were happy that they got to change their previous groups, because there the work was stagnant, and we were all to share the same feeling, that is why the motivation was there. However, later on it became harder and harder to work together, and this could be attributed to several factors: different levels of engagement and time investment, difficulties to meet in person (later on even online), the fact that one of the team members suddenly dropped out (decided to leave the studies), the fact that half of the group went away to their home countries in the first week of December when we were supposed to have the most time investment for our semester project.

I felt responsible for the project by actively taking part in the project writing, in the sections I felt the most confident in. In this case it was the project and process reports. I was always showing up on the agreed meetings and available for my team members in case they needed help.

Looking back at the contract and the expectations I had before it is nothing like it turned out to be. I would say that by previously put forth guidelines our group failed to bring them into reality. Our contract was divided into 5 different sections: participation, communication, meetings, conflicts and deadlines. In the participation section we agreed to strive for the best result, invest 2-3 hours daily in the project, divide the work equally. During the project not everyone was investing the equal amount of time, thus striving for the best result did not happen. Also, investing every day 2-3 hours did not work, there were days when nothing was done, and there were also days when most of the work was done. Communication section more or less met my expectations, however, since half of the group left home very early in the month of December, it was sometimes difficult to reach these members and get help with the project. In the meeting section we agreed to always be on time, notify in case of not showing up and try not to skip the meetings in general. If in the beginning of the project the team did not have any problems to fulfill these criteria, then towards the end it was a total disaster. Some of the members were frequently late, had no valid excuses, and on a few occasions not showing up at all without even

saying something. Good thing was that we did not have any major conflicts, which could be attributed to our personalities, where people would instead avoid the conflict altogether. The last section of the contract was about meeting the deadlines, where we agreed to do everything on time and not to postpone the work until the last minute. Given that we were sometimes slacking and not doing a proper job, as well as, not always receiving input from all the members, we encountered this problem. The project report was done in a slightly rushed manner, therefore there might be some imprecisions and sloppiness to it.

It would be a good idea to mention the action plan with the steps to do if something is not followed by the members in the contract. For example, what happens when somebody skips more than 3 meetings, or what to do when someone leaves the project earlier than expected (going for holidays before actual holidays), because it puts extra pressure on the team and the overall motivation in such a situation might suffer. In our group contract we only scratched the surface about the action plan - it was talked about in the section of conflicts and we decided to introduce a voting system in case of disagreements (*see Appendix J - Group Contract*).

It is always a challenge to work in the multicultural environment, since the difference in the behavior and the way of thinking. What works best for me is that I always try to find some similarities and hang on to them, it helps to establish a better relationship between the two members of different cultures. The biggest benefit of working in such an environment is to think about the problems from way different perspectives and sets of values. Because of the fact that I previously already had an experience working in a multicultural environment, I felt like I am ready for it. However, it is always a new and different experience, because you never know what kind of people you are going to meet. Sometimes I felt very demotivated when something did not go my way, next time I would try to look over that feeling and try to involve myself more, as well as others, to resolve this issue.

One of the most important advantages of working in groups and problem based learning is the fact that it prepares you for the work environment. I think it is a great way to learn, because you get not only the theory of the subjects, but you also get to practice them. Later on, when searching for work in real life, it is going to be easier to jump in and quickly adapt in any workplace, because you already have some experience to a certain degree.

The disadvantage could be that by working in the group things don't always go as you have planned, therefore you have to be ready for lots of compromises. While I think the problem based learning is a good strategy to learn, it also has some negative side to it. For example, it tries to model the real work situation, but it does not take into account that at the real workplace there would be no people who are performing very bad, since they would most likely be fired right away or not taken to

that workplace in the first place. While this is not always the case in the schools and universities, since this fact does not translate to learning institutions.

Good tools when starting to work on the projects are problem formulation and project description, it helps to gain a deeper understanding of what needs to be solved and the way it is going to happen. It gives a solid base for the project and acts as a guideline for the project's completion. I think it is a great way to start any project, since it allows you to be more structured in your work. However, sometimes it can also act as a barrier, because in the case of using these tools, you put yourself in a certain position and have predetermined expectations, which sometimes can be hard to achieve.

## Henrik

I am very happy with the outcome of the project and it was refreshing to work with people who were as committed to producing good work as myself.

Our group was motivated to meet regularly and this reflected positively on our project because it ensured that everyone was up to date on what needed doing and what everyone was working on. However, motivation for completing the actual work at hand was somewhat sporadic, with some work only being completed as much as the person was actually interested in it. This is understandable in a way because some tasks are much more exciting to work on than others. Had we worked on these more tedious sections together, maybe motivation would have improved.

I feel like I could have contributed more to coding the program. Had we split implementation into parts and assigned each other specific tasks, every member would have had an equal workload. Despite this, I firmly believe that everyone did a sufficient amount of work. In my opinion, the group contract was fair and flexible so I would not change anything about it.

I enjoyed working with a multicultural group because you learn how despite cultural differences, people can still collaborate effectively. In my experience, working with people of the same nationality as yourself can create too much comfort and over-reliance on each other. As such, people tend to be more relaxed and work is not prioritized as much as it should be.

Problem-based learning (PBL) provided a structured way of thinking and trained us to assess a problem from multiple different perspectives. My only issue with PBL is that it takes time to get used to. We would have benefited from some real world examples of PBL and perhaps a few exercises before starting the project. If this was done, the project description would not have taken so long to complete and we would have had more time to focus on the work itself.

I have always prospered in group projects where everyone is willing to contribute. However, when this is not the case, group projects can become a hassle because it takes your attention away from the actual work and you find yourself spending more time solving group conflicts instead. This is especially true for group projects where a member establishes themselves as a leader because it can create a divide between everyone involved. Thankfully, this was not the case in our project, as everyone took initiative when they needed to.

The order in which the project was developed was more than satisfactory. Starting with a project description helped to establish the context for the product as well as assess the problem in depth. However, if we had the opportunity to do this project again, I would have started documentation much earlier. Had we done this, we would have had more time for polishing the final product itself. Furthermore, it would better depict our original thought process.

Overall, I learned a great deal from this project, namely the importance of regular communication and effective time management.

## Kim

I started out in a different group, but my reflections are going to be based on the project period starting when I joined the final group.

I felt responsible for trying my best, based on my motivational orientation of being knowledge motivated, and I like to challenge myself when it comes to programming. I took a leading role, keeping track of deadlines and what we had to do in the different group based hand ins during the semester. But before the project weeks I had lost almost all motivation, caused by some group members' decision of going home before we would start on implementation, and lack of interest in the project. Started out by planning group meetings in calendar, but since the time of some meetings was postponed, and yet some members still did not show up on time, I lost all interest in trying to schedule any meetings. This was showing lack of interest in the project.

Our group contract states that we want to "strive for the best results" (*Appendix J. Group contract*), and spend many hours working on the project each week, while attending all lectures, and try to split project work equally. In the end we did not strive for the best result, but the best we could do with the limited time caused by lack or poor time management. Time investment was really low compared to what was stated in the contract. Attendance of all lectures was not followed by all group members. This was especially evident when it came to implementation of the code, where in the beginning we would split up work on the code. But after a few days of slow or no progress at all, and hearing some group members mentioning they had to go back and learn how to do parts of the coding, I started to do all the coding myself.

This is not a good approach when it comes to group work, but since we did not do any pre planning of the two project weeks, and many days had already passed. I evaluated that it would be the only way of getting a finished working system. Furthermore, i had set up a git repository hosted on github, which would make collaboration on the code much easier, but since we did not have much experience using it, it also took some time to learn it first, and this was another barrier for some group members(*History of code development can be found in appendix files gitlogshort.txt and gitlogstats.txt*). So my strengths were being used by me focusing on coding, which led to the point that I did not have much time to focus on report writing. Even though the project report should have a higher priority. This was strengthened further by my poor report writing skills.

Our group had a large age difference which could explain some of the differentiating motivation. As for myself, being an introvert, I tend to avoid conflicts. If not it could have helped the group if we talked about our current motivation, and how others actions affected each other's motivation. Maybe start by talking about why it was hard to show up on time for group meetings. A more aggressive approach could be to point towards the group contract, and talk about what we had agreed upon, but this should only be done if the more neutral approaches would not work. When it comes to coding I am very stubborn when it comes to ways of doing the code, I am sometimes having a hard time accepting other people's point of view on the code, but since I am aware of this it is not that big of a problem most of the time.

Some advantages of problem based learning are that it is possible to help each other learn more, split work between group members so it's possible to be more efficient, and possible to reach bigger goals when working together. But some disadvantages means that you can spend unlimited time learning new ways of solving a problem, and some group members might take a more passive role, and contribute less, since others might do more work.

## 7. Supervision

During the SEP project, the team received support from the supervisors, which were also our teachers during the semester. To be more specific, the team had a possibility to ask questions in case of any uncertainties, as well as receive feedback on some parts of the project. Mentioned parts were Analysis and Design, for example, the students received feedback when creating the list of requirements, making Use Case diagram and descriptions, Activity diagrams, and lastly - the Class and Sequence diagrams. It was a good guiding tool, since we had a possibility to improve on our already existing material that we used for the project.

In addition, there was a possibility to reserve a time slot with the supervisors to talk about any problems we encountered related to the project. It was a great chance to find the answers to the questions that arose during the project writing. Being all caught up while making the system, the team unfortunately forgot about such an option, but got then reminded by one of the supervisors. This specific moment stayed in our memories, because we realized that we could get all the help we needed, which we also utilized in the end. One more specific thing that we remember was that we got support even outside working hours of the supervisors, the emails were answered almost right away regardless what time of the day it was.

Unfortunately, the current project was conducted during the “Corona times”. That of course put some restrictions on how the process of supervision was managed. If in the beginning stage of the project the supervision took place mostly in person and partly online through the emails, then later it changed. Especially closer to the end of the project, the supervision was happening only online due to the worsening of the pandemic situation. For our team, we felt that the meetings with the supervisors were more successful when we could meet in person, because of the face to face interaction. Nevertheless, we did receive the answers for all the questions we had, regardless of the change of the situation.

## 8. Conclusions

It was the group consensus that a more detailed time schedule would have ensured more equal workload, especially in the implementation stage. Our original time schedule only showed the number of days we would spend on a given stage in relation to hand-in dates. A better solution would have been to outline how long we should spend on specific tasks within a particular stage. For instance, within analysis we should have determined how long we would spend on the list of requirements, use case diagrams, use case descriptions, activity diagrams and the domain model. We also should have established in what order these tasks were carried out and who was working on them at any given time.

Our project would have benefited from starting documentation earlier. We should have updated the project report whenever we completed a certain task. For instance, when we finished our activity diagrams, we should have written a summary of what we did, how we did it and why. This way, as the project progressed, so would the project report. Furthermore, whenever changes were made, one could simply do the same in the project report and write a note somewhere that a particular feature was changed and why. If we did this from the very start of the project, by the time we go to the implementation stage, we would have already completed most of the required documentation. Additionally, it would show a more natural thought process to our work.

## 9. Sources of information

1. Hofstede Insights, 2020. *Country comparison*. [online] Available at: <<https://www.hofstede-insights.com/country-comparison/denmark,latvia,portugal,romania/>>[Accessed 17 December 2020].
2. MindTools, 2020. *Forming, storming, norming, performing*. [online] Available at: <[https://www.mindtools.com/pages/article/newLDR\\_86.htm](https://www.mindtools.com/pages/article/newLDR_86.htm)>[Accessed 17 December 2020].
3. E-stimate Personal Profile, 2019. *E-stimate Personal Profile*. [pdf] Available at: <<https://via.itslearning.com/LearningToolElement/ViewLearningToolElement.aspx?LearningToolElementId=1345193>>[Accessed 17 December 2020].