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

Character Count:

Semester 1

Date: 17/11/2020

[**Abstract**](#_x3f4uty9jm1a) **3**

[**1. Introduction**](#_ahio13kwhbjj) **3**

[**2. Analysis**](#_vjp4szodhhrs) **4**

[2.1. Functional requirements](#_fkjvt7ql67ab) 4

[2.2. Non-functional Requirements](#_1j0cu1levs9w) 6

[2.3. Use Case Diagram](#_3dwqvymh1wdp) 6

[2.4. Use Case Descriptions](#_aimo219u2dh) 7

[2.5. Relation between requirements and use cases](#_pt92c3n7y0nt) 11

[2.6. Activity Diagrams](#_p21sjtimzqdu) 12

[2.7. Domain Model](#_1c4bqmabqmja) 19

[**3. Design**](#_gn7oqtaq3w63) **19**

[3.1. Class diagram](#_n80d5mf3zpa2) 21

[3.2. Sequence diagram](#_ak01jd4ifo9c) 22

[**4. Implementation**](#_bmo31npxns7h) **22**

[**5. Test**](#_29jnxxy8qmtg) **22**

[5.1. Test specifications](#_e4v891a0feef) 23

[**6. Results and discussion**](#_vga27mo3davb) **23**

[**7. Conclusions**](#_yoncmaim4sbd) **23**

[**8. Project structure**](#_4aibrhsdky77) **23**

[**9. Sources of information**](#_srmhysag6lzt) **23**

[**10. Appendices**](#_3e49kp6ihlei) **24**

[10.1. Appendix A - project description](#_1e65ls8idfwq) 25

# 

# Abstract

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

1. What are the aim and objectives of the project

2. What are the main technical choices

3. What are the results

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

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

# 1. Introduction

The purpose of the introduction is to provide background information and set the scene for your project. Within which business or organization are you doing the project? Who are the stakeholders and who is the customer?

The background information is adapted from your project description where you have already described the problem domain. Describe the current situation and existing context. Your statements must be supported by references to reliable and relevant sources.

This should lead to why this project is relevant and outline your aim and objectives. Which technical problems and challenges will be presented in this report, again taken from your project description. System illustrations and rich pictures are welcome here.

State delimitations relevant for your project in the introduction. Delimitations include what the project will not cover in relation to your project description, i.e. what could have been expected in your project. Remember that you can only make delimitations to aspects mentioned in the project description and you must argue well for your delimitations.

The last sentences of the introduction should be an overview of the sections to follow. This will be a good transition to the next sections.

Remember: You must ensure a clear connection between sections in the project report, from Project Description, Analysis, Design, Implementation to Test. This means that everything that is implemented can be found in design, everything that is designed is based on the analysis, and anything that is found in analysis has a clear link to requirements, etc.

# 2. Analysis

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

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

Use the UML standard for all diagrams where relevant.

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

## 2.1. Functional requirements

**Critical priority:**

1. As a project creator, I want to be able to create projects that contain requirements and tasks, such that we can track the progress of the project.
2. As a project creator, I want to be able to assign team members and their roles, such that they can work on projects.
3. As a product owner, I want to be able to add, remove and prioritize requirements, so that projects can be developed using an iterative approach.
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 team member, I want to be able to access the system without having to login, because the system can only be used by one team member at a time.

**High priority:**

1. 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.
2. 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.
3. 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.
4. 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 and by who) such that I can easily get an overview of all relevant information for a requirement.
5. 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.
6. 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.

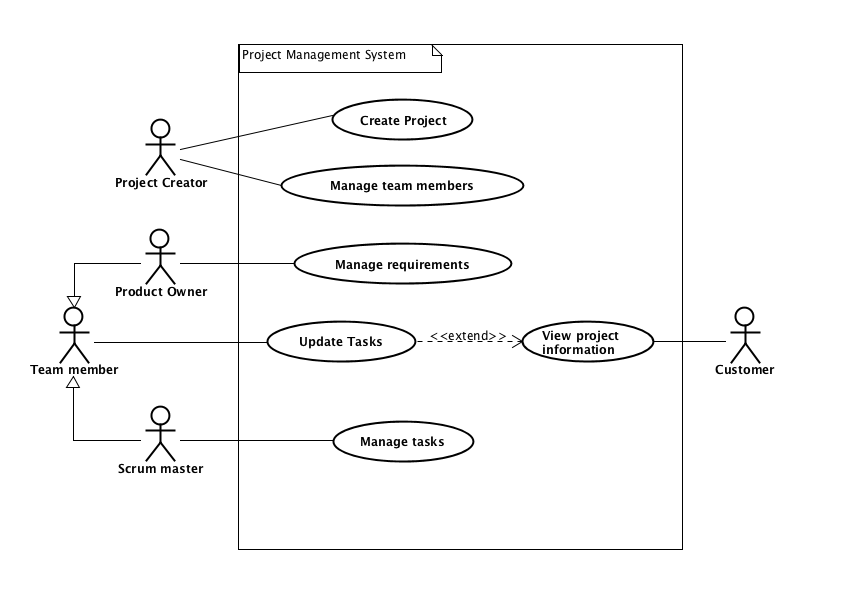
**Low priority:**

1. As a product owner, I want to be able to search information regarding the projects by ID, responsible team members, deadlines, such that I can have an overview of the progress on a current project.
2. 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.
3. As a customer, I want the project’s description, requirements and their status, displayed on a website, such that I can track its progress.
4. As a project creator, I want to be able to add/remove new team members in case the team is not able to cover all the tasks.
5. As a customer, I want my project’s information to be updated daily, such that I am well-informed on its progress.

## 2.2. Non-functional Requirements

1. As a product owner, I want the GUI implemented with Java/JavaFx, such that it would be easy to modify.
2. As a customer, I would like to be able to access project information 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). So that I will be able to find information from multiple devices.

## 2.3. Use Case Diagram



## 2.4. Use Case Descriptions

| **Use case** | **Create Project** |
| --- | --- |
| **Summary** | Creating a project in which the customer can see information |
| **Actor** | Project Creator |
| **Precondition** | Analysis should be accepted by the customer and all requirements and tasks should be specified |
| **Postcondition** | A project is created, therefore, requirements, tasks and other work can be completed |
| **Base sequence** | 1. Open system 2. Create a project 3. Choose the name of the project 4. In case requirements and tasks are to be added, go to step 5, otherwise go to step 9 5. Add requirement 6. Add task for the requirement 7. If necessary, repeat step 6, otherwise go to step 8 8. If multiple requirements should be added, go to step 5 9. Save project |
| **Exception sequence** |  |
| **Note** | This use case covers requirements 1, 2 |

| **Use case** | **Manage Team Members** |
| --- | --- |
| **Summary** | Adding/removing Team Members or assigning (new) roles to those |
| **Actor** | Project Creator |
| **Precondition** | A project has to be created |
| **Postcondition** | The project is better organized so that there are no disruptions to completing the requirements for it |
| **Base sequence** | 1. Open system 2. Search for a project 3. Add Team Members 4. In case Team Members are to be removed, go to step 10 5. Assign roles to each Team Member 6. In case of changing roles of a Team Member, go to step 14 7. In case of unpredicted delays or reasons that interfere with getting the tasks done, redo either step 3 or 4 8. Save project 9. Start working on the project 10. Remove Team Member 11. If necessary, repeat step 10, otherwise go to step 12 12. Save changes 13. Go to step 5 14. Choose Team Member 15. Change role 16. If required, repeat step 14, otherwise go to step 17 17. Save changes 18. Go to step 7 |
| **Exception sequence** |  |
| **Note** | This use case covers requirements 4, 5, 16, 21  If there is no project created, then there can not be any management of the Team Members done. |

| **Use case** | **Manage requirements** |
| --- | --- |
| **Summary** | Add or remove requirements, such that the project is developed using an iterative approach. |
| **Actor** | Product Owner |
| **Precondition** | A project should already be created in the system. |
| **Postcondition** | The requirements are now updated. |
| **Base sequence** | 1. Open System. 2. Open Project. 3. Open List of Requirements. 4. Add or Remove Requirement/s. 5. Give Requirement/s an ID, estimated time, user stories, deadline, who is responsible for working on them, status and total time spent. 6. If all tasks for requirements are complete, the requirement is marked as ‘ended’. 7. Change Priority of Requirements if need be. 8. Approve or reject requirements if need be. 9. Save List of Requirements. 10. Save Project. |
| **Exception sequence** |  |
| **Note** | This use case covers requirements 3, 6, 7, 8 |

| **Use case** | **Update tasks** |
| --- | --- |
| **Summary** | Update tasks that a team member has worked on, and find project information. |
| **Actor** | Team member |
| **Precondition** | A project has to be created in the system. |
| **Postcondition** | Tasks have been updated, and team members have found required information. |
| **Base sequence** | 1. Open system 2. Search for a project 3. If project not found go back to step 2 4. Open the project 5. For updating tasks go to 5 6. Finding information about a project go to 10 7. Open tasks 8. Open task you need to update 9. Enter amount of hours spent and what date 10. Save task 11. If more tasks need to be updated go to step 2 12. Read information about project 13. To find information about requirements go to step 11 14. To find information about tasks go to step 13 15. Open requirements 16. Open specific requirement you are interested in 17. To find information about a different project go to step 2 18. Open tasks 19. Open specific task you are interested in 20. To find information about a different project go to step 2 |
| **Exception sequence** |  |
| **Note** | If there are no projects in the system, team members will not be able to do anything in the system.  This use case covers requirements 5, 10 |

| **Use case** | **View project information** |
| --- | --- |
| **Summary** | Customer finds information regarding the ordered project. |
| **Actor** | Customer |
| **Precondition** | Work has started on the customers ordered project. |
| **Postcondition** | Customer knows about progress on the ordered project. |
| **Base sequence** | 1. Enter provided web address 2. Search for specific project 3. Wrong search term, repeat step 2. 4. Open project 5. Read project information 6. For information about a requirement go to step 5 7. Open requirements 8. Open specific requirement 9. To find more information about the project go to step 3 10. To find information about another project go to step 2 |
| **Exception sequence** |  |
| **Note** | Website has to be regularly updated by the development team in order for the customer to see the latest information regarding their projects.  This use case covers requirements 12, 14, 16 |

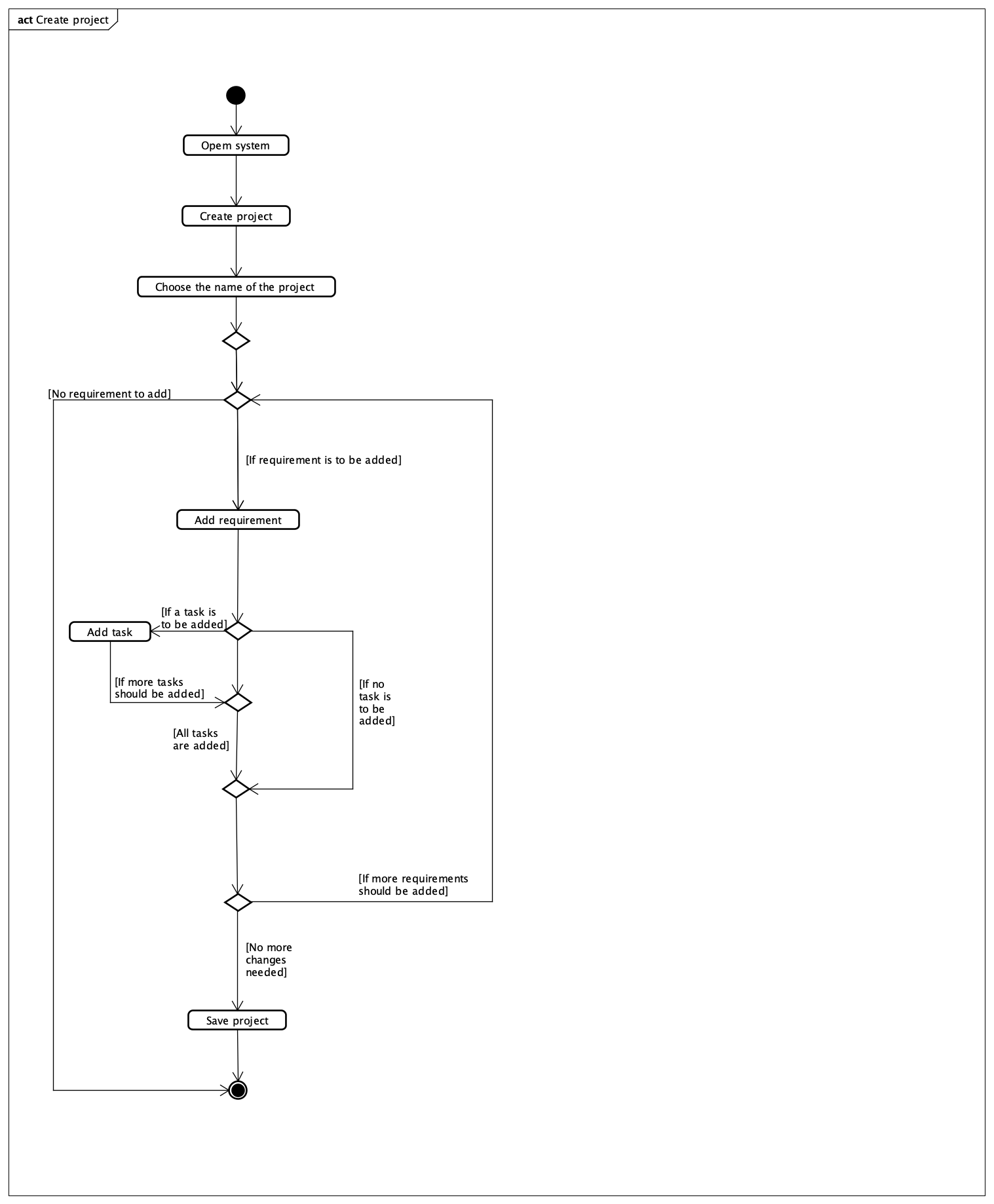
| **Use case** | **Manage Tasks** |
| --- | --- |
| **Summary** | Adding tasks to the project and documenting the status of each task in form of - “Started”, “Not started”, “Finished”. Also, reporting the status of the tasks further to the Product owner. |
| **Actor** | Scrum master |
| **Precondition** | Project requirements have to be identified and project with assigned team members created. |
| **Postcondition** | A task has been added and later on updated from “Not started” status to "Started" status or “Finished” status. A sent documentation to the Product owner about the status information for further approval/verification. |
| **Base sequence** | 1. Start the system 2. Search for the requirements by their ID to display necessary information like description, time estimation, deadlines, responsible team members. 3. In case of a new project - Add task/-s, if not, then skip this step and go to step 4. 4. Follow-up and check the status of each specific task and requirement by using a search option like in step 2. 5. Update the status of the project by marking each task:  * If the work process on the task has begun, then mark it as “Started”. * If the work process on the task has not begun yet, then mark it as “Not started”. * If the work process on the task has been finished, then mark it as “Finished”  1. Save changes. |
| **Exception sequence** |  |
| **Note** | This use case covers requirements 4, 9, 11. |

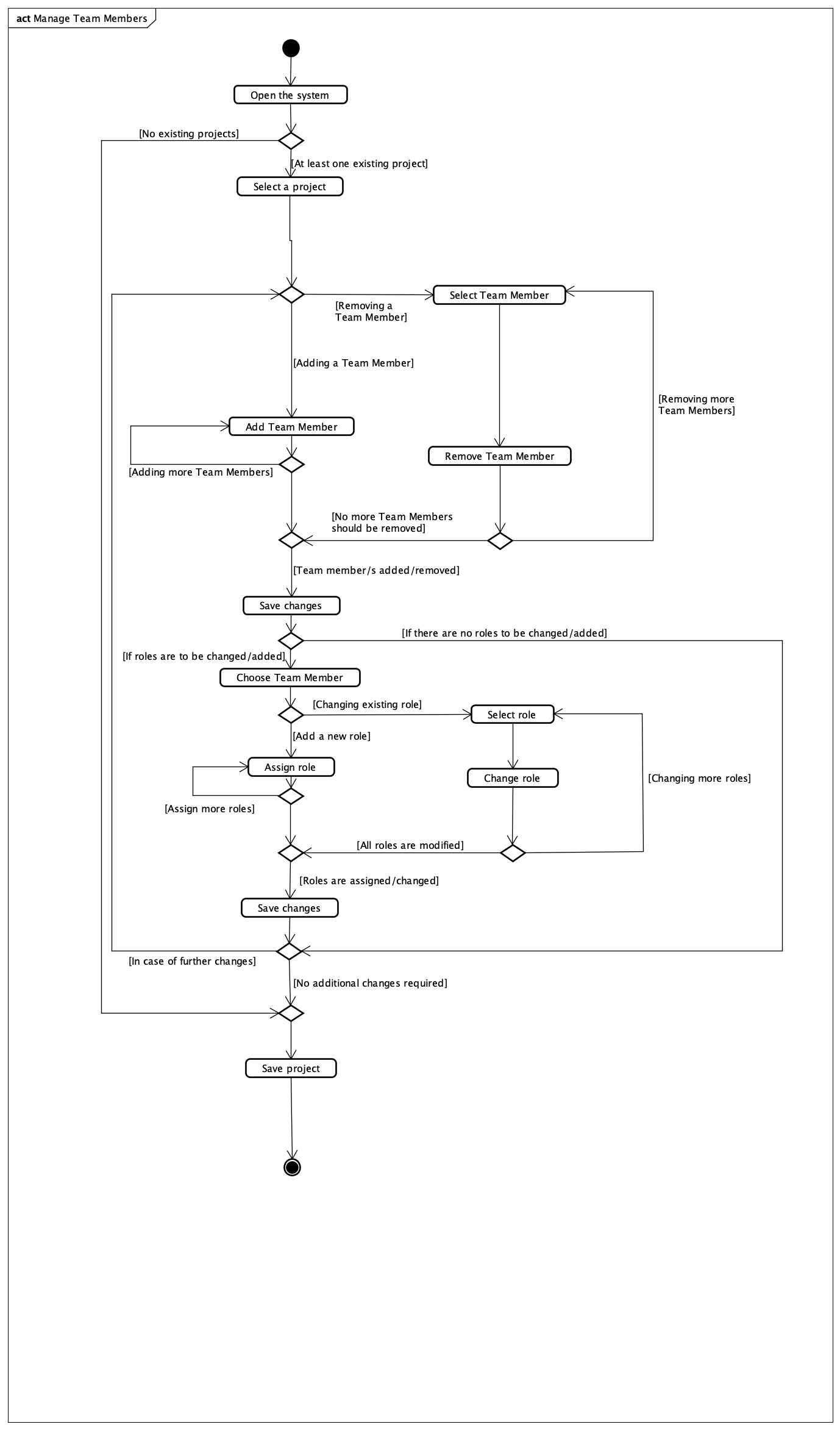
## 2.5. Relation between requirements and use cases

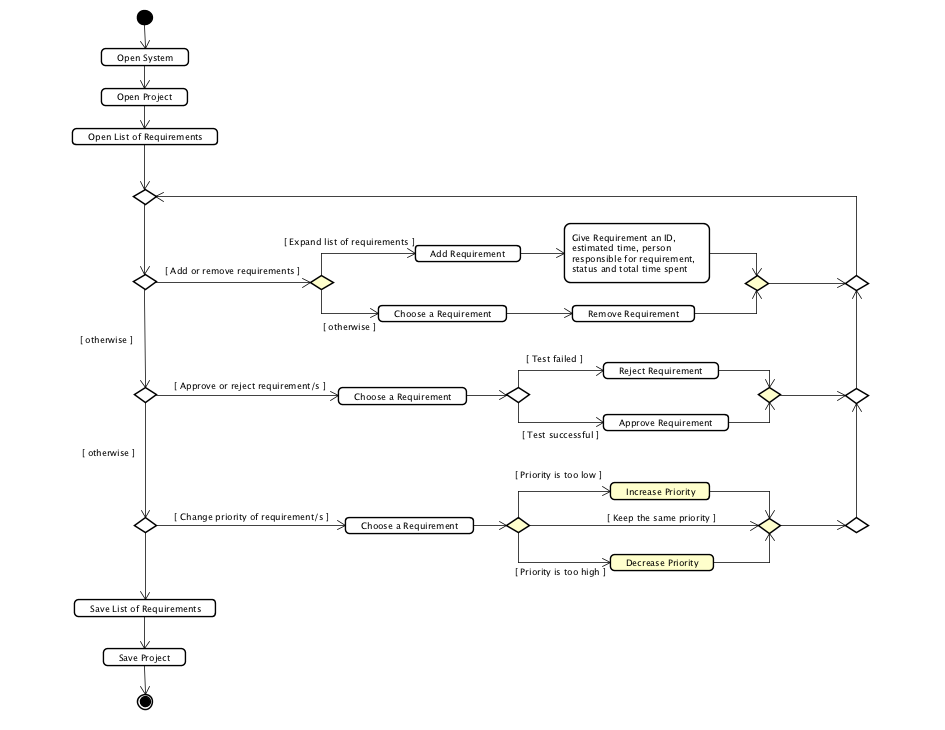
| **Use case** | **Covered requirements** |
| --- | --- |
| *Create project* | 1 |
| *Manage team members* | 2, 13, 15 |
| *Manage requirements* | 3, 6, 7, 8 |
| *Update tasks* | 5, 10 |
| *View project information* | 12, 14, 16 |
| *Manage tasks* | 4, 9, 11. |

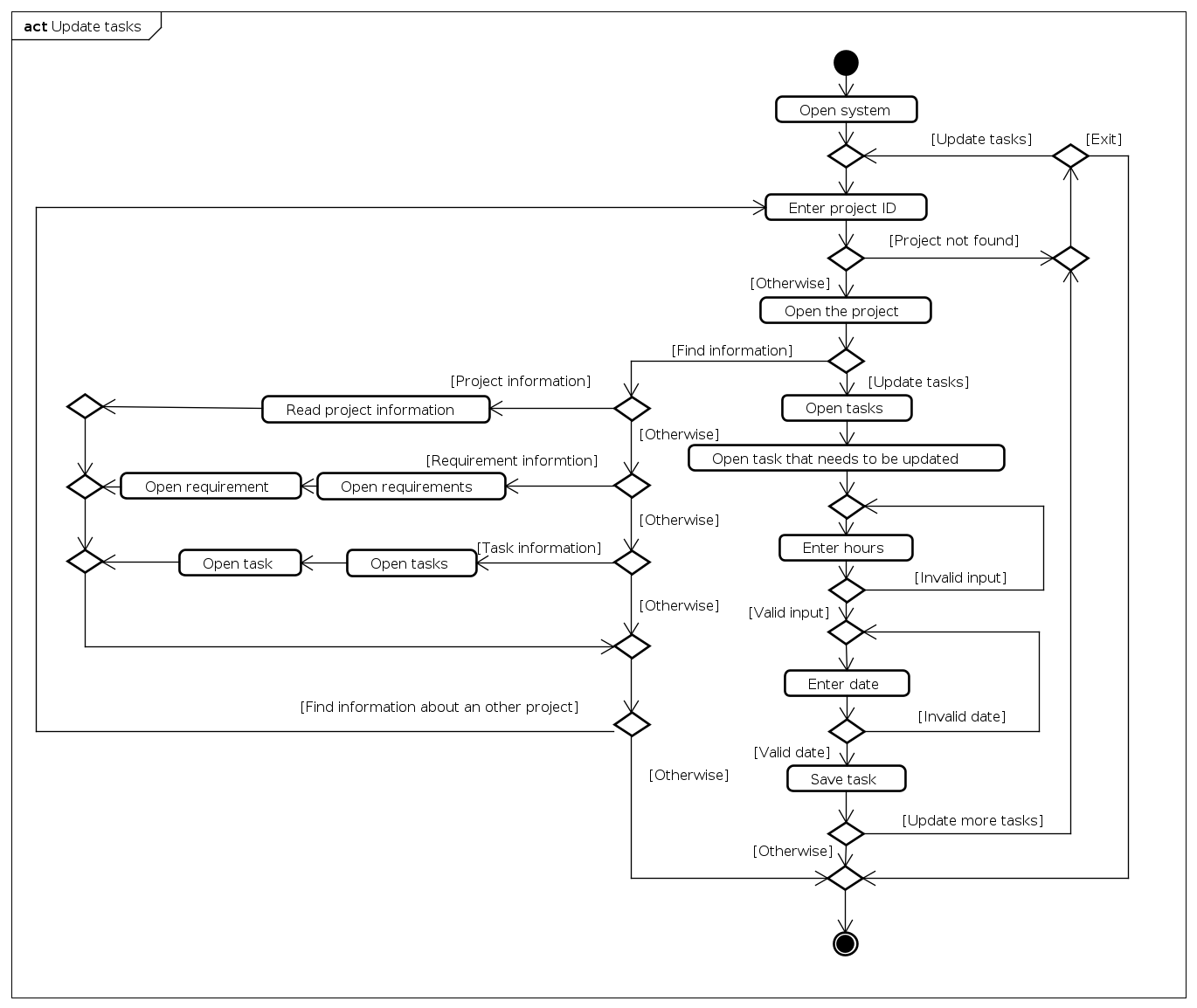
| **Requirement** | **Related use case** |
| --- | --- |
| **1** | Create project (step 2) |
| **2** | Create project (step 3) |
| **3** | Manage requirements (step 4) |
| **4** | Manage tasks (step 3) |
| **5** | Update tasks (step 1) |
| **6** | Manage requirements (step 5) |
| **7** | Manage requirements (step 4a) |
| **8** | Manage requirements (step 7) |
| **9** | Manage tasks (step 2 and 4) |
| **10** | Update tasks (step 7) |
| **11** | Manage tasks (step 5) |
| **12** | View project information (step 2) |
| **13** | Manage team members (step 5) |
| **14** | View project information (step 3) |
| **15** | Manage team members (step 3, 10) |
| **16** | View project information (step 3) |

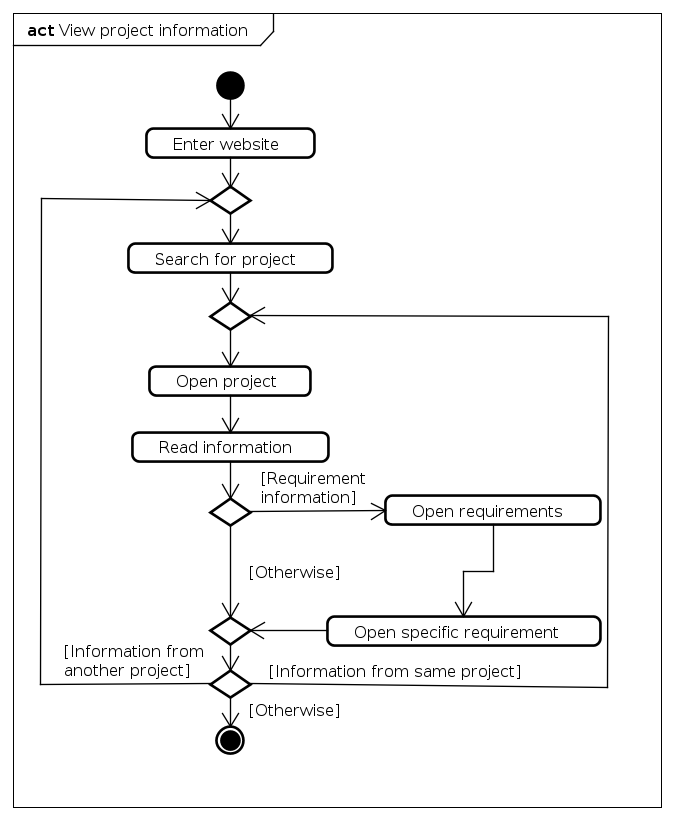
## 2.6. Activity Diagrams

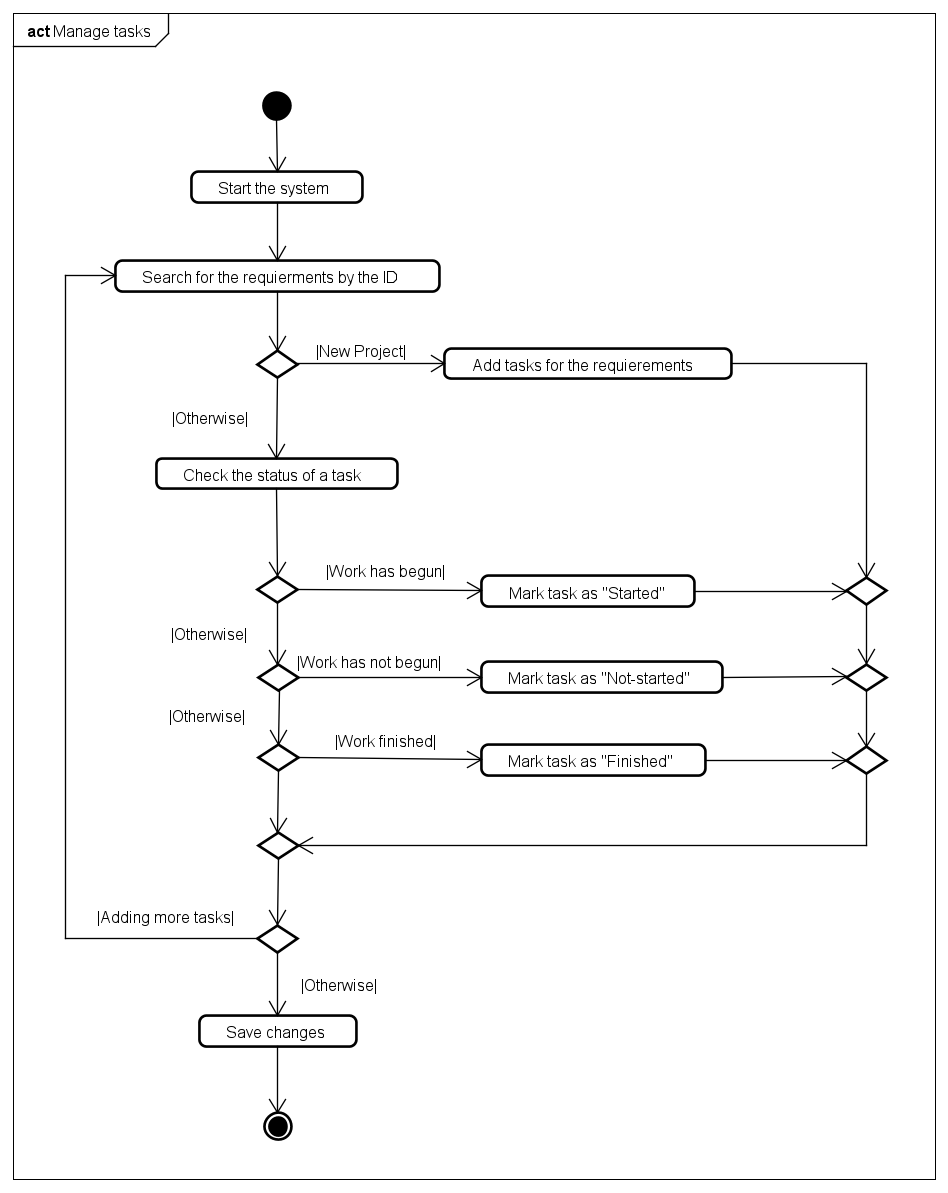




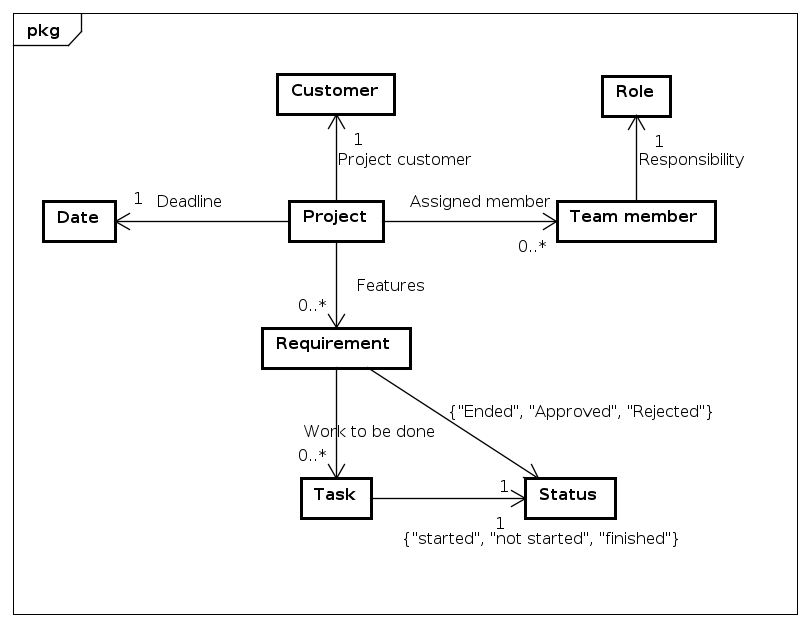








## 2.7. Domain Model



# 3. Design

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

Elements that may be relevant in this section:

\* Architecture: Find architecture patterns here (Leszek Maciaszek 2004, chap.9).

\* Technologies: Describe technologies used, also alternative technologies. Argue for choice of technology according to the project aim.

\* Design Patterns: Describe which design patterns (GoF (Gamma et al. 2002) etc.) you are using and why.

\* Class Diagrams

\* Interaction Diagrams

\* UI design choices

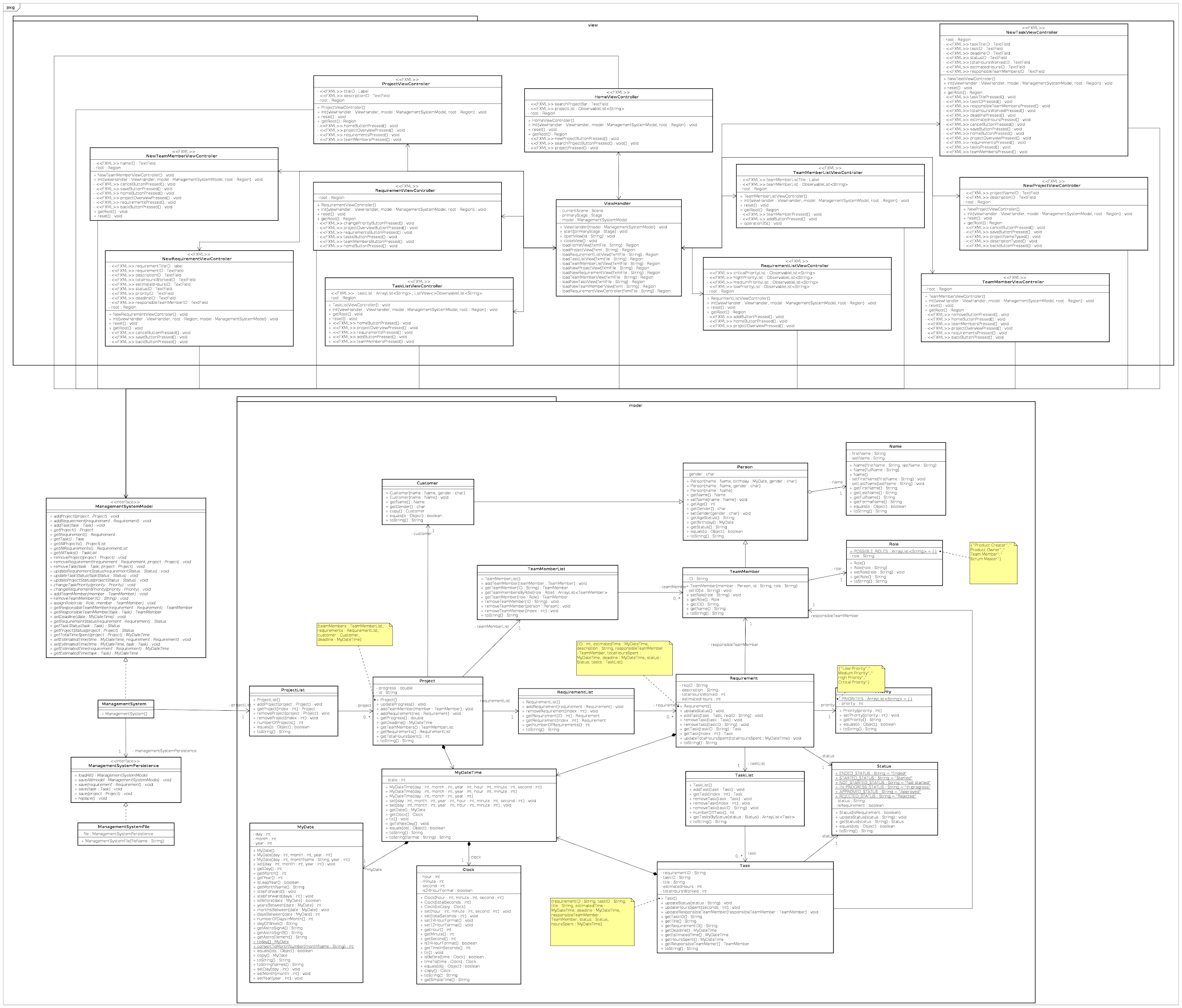
\* Data models, persistence, etc.

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

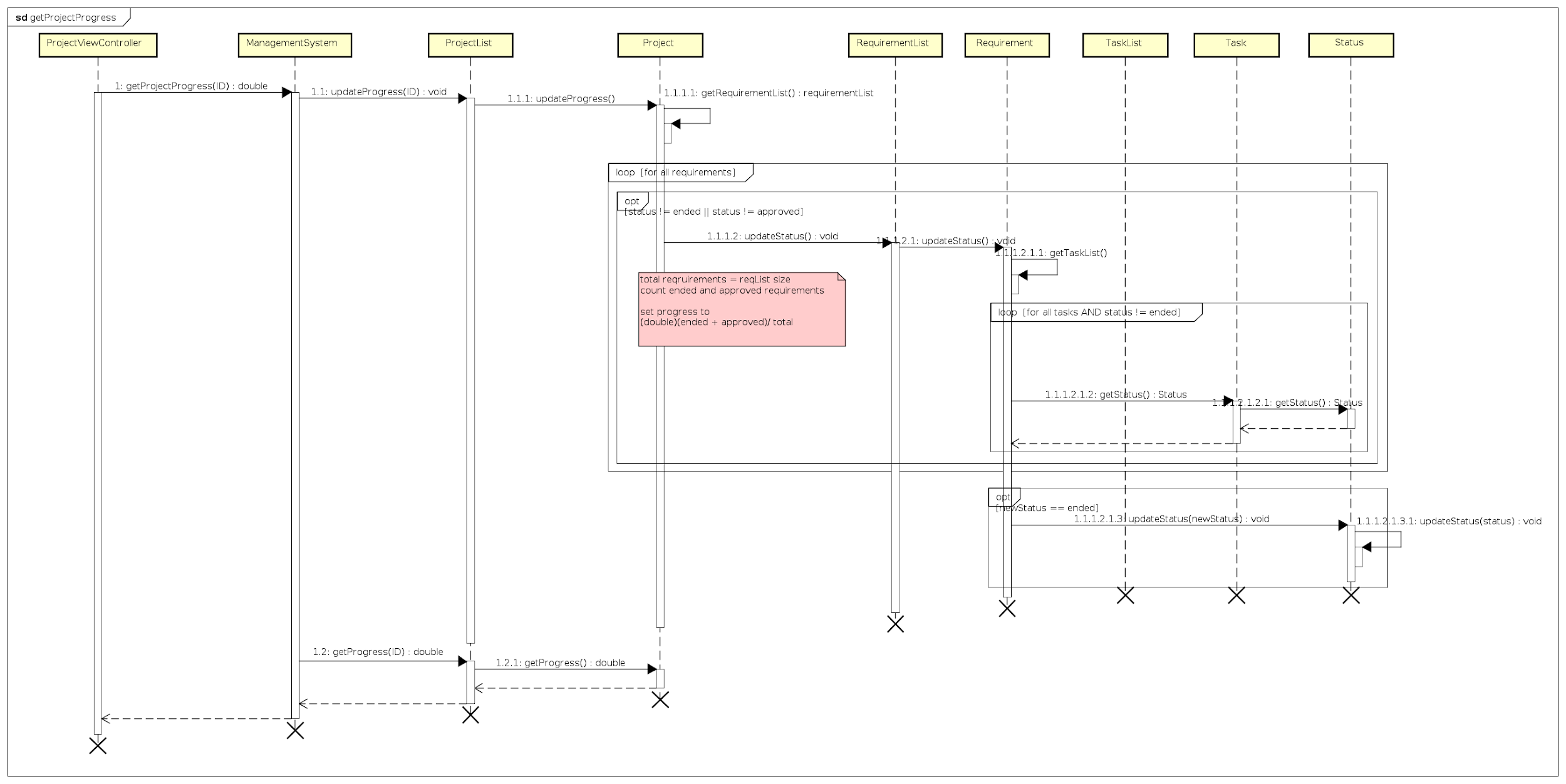
Hint: One way to figure out which objects/classes are needed in the design is to apply the General Responsibility Assignment Software Patterns/principles (GRASP) (Larman 2004, chap.17).

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

## 3.1. Class diagram

****

## 3.2. Sequence diagram



# 4. Implementation

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

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

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

Hint: Implement your code in a testable manner.

# 5. Test

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

## 5.1. Test specifications

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

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

# 6. Results and discussion

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

# 7. Conclusions

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

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

# 8. Project structure

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

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

# 9. Sources of information

# 10. Appendices

Examples of appendices:

\* Project Description

\* User Guide

\* Source code – source documentation

\* Diagrams

\* Data sheets

\* Etc.

## 10.1. Appendix A - project description

Software Project Managing System

Aleksandrs Bistrovs 304542

Henrik Koster 305916

Kim Tranberg 172394

Laurentiu Mihai 304456

Astrid Hanghøj

Steffen Vissing Andersen

Software Engineering

First semester

7.10.2020

Table of content

[Background description](https://docs.google.com/document/d/1KsEYc92SzDZtvDoBeUxBCb7VmEeKl2X-SbLPFO_eerE/edit#heading=h.2et92p0) [1](https://docs.google.com/document/d/1KsEYc92SzDZtvDoBeUxBCb7VmEeKl2X-SbLPFO_eerE/edit#heading=h.2et92p0)

[Problem Statement](#_1xcxwr5vcmb) 3

[Definition of purpose](https://docs.google.com/document/d/1KsEYc92SzDZtvDoBeUxBCb7VmEeKl2X-SbLPFO_eerE/edit#heading=h.lqbip097q5oq) [4](https://docs.google.com/document/d/1KsEYc92SzDZtvDoBeUxBCb7VmEeKl2X-SbLPFO_eerE/edit#heading=h.lqbip097q5oq)

[Delimitation](https://docs.google.com/document/d/1KsEYc92SzDZtvDoBeUxBCb7VmEeKl2X-SbLPFO_eerE/edit#heading=h.1t3h5sf) [5](https://docs.google.com/document/d/1KsEYc92SzDZtvDoBeUxBCb7VmEeKl2X-SbLPFO_eerE/edit#heading=h.1t3h5sf)

[Methodology](https://docs.google.com/document/d/1KsEYc92SzDZtvDoBeUxBCb7VmEeKl2X-SbLPFO_eerE/edit#heading=h.4d34og8) [6](https://docs.google.com/document/d/1KsEYc92SzDZtvDoBeUxBCb7VmEeKl2X-SbLPFO_eerE/edit#heading=h.4d34og8)

[Time schedule](#_2s8eyo1) 7

[Sources of Information](#_3e217rq2dbs5) 9

[Appendices](https://docs.google.com/document/d/1KsEYc92SzDZtvDoBeUxBCb7VmEeKl2X-SbLPFO_eerE/edit#heading=h.jy9olvnsje1h) [10](https://docs.google.com/document/d/1KsEYc92SzDZtvDoBeUxBCb7VmEeKl2X-SbLPFO_eerE/edit#heading=h.jy9olvnsje1h)

Background description

Current demand for IT solutions is high and demand for IT solutions keeps increasing(Mordor Intelligence, 2020). Especially in the months following the current COVID-19 pandemic (Research Nester, 2020), where companies are forced to find new solutions that let their employees work from home (Felix Iblher, 2020).

Managing large IT projects is a very complex task while at the same time also very important and challenging. There are many reasons for this, such as technology required to meet a customer's demands can be very complex, time limited development and having to stay within budget.(F. John Reh, 2019)

Development of IT solutions is a very time consuming and expensive process, and can often become even more expensive than what was estimated before the start of a project. Things like not having a clear objective, requirements changing during the development process, lack of skills required to fulfill stated requirements by the customer and poorly managed time schedules. These reasons often lead to a finished solution that does not meet the requirements of the customer, which causes further development time and expenses (Michael Bloch, Sven Blumberg, Jürgen Laartz, 2012).

Updating customers on progress of their ordered projects is a very important aspect of customer satisfaction, keeping your customers satisfied is really important to all companies that sell products or services. Otherwise “Many times, however, consumers do not complain to the company, but instead take actions such as switching brands or engaging in negative word of mouth (WOM).” (Hawkins & Mothersbaugh, 2010, p. 636)

(Note that it was handed in as a previous group but was written by Kim Yung-un Supreme Leader)

Definition of purpose

The purpose of this project is to provide a company in question with an IT solution for their development projects, that helps the company in monitoring the progress and managing the work.

# Problem Statement

Managing large software projects is a very difficult activity to do, keeping an overview of all the tasks each requirement has, the progress, who is working on it and so on. It will quickly become unmanageable.

The following sub-questions are formulated to get a better understanding of the main problem:

1. How to efficiently manage software development projects?
2. What could be done to avoid a project falling behind schedule?
3. What kind of tools should be available to the user of the system?
4. In which way should customers be able to track progress of their projects?
5. What kind of information should be available to the customer?
6. Who is responsible for the cost, when the finished product does not meet the users requirements?

(Note that it was handed in as a previous group but was written by Kim Yung-un Supreme Leader)

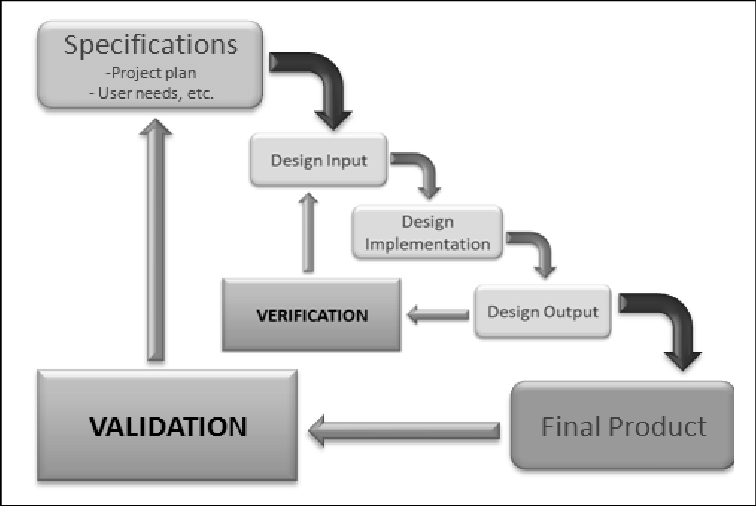
Delimitation

Considering the time restrictions, the team will not be able to carry out the solution for large IT development projects.

The project is not going to include the consideration of the budget for such a system itself, as well as, a solution to the problem about the extra costs that could occur in case if the project does not meet the requirements.

Methodology

In the given project the development team will make use of the modified Waterfall approach. It consists of 5 stages, which follow strict, linear order, where each stage has to be fully completed before moving on to the next one (Winston Royce, 1970). In order for the Waterfall model to work, the requirements have to be well-defined, otherwise it is fairly easy to fail within this approach (Rumor, 2019). Since there is no possibility to navigate between the given stages, and fix the occurred problems on the way, the team has decided to use a modified version of the Waterfall method. It is being done in order to work around the previously mentioned issue by enabling the users of the model to go back and forth between the stages and fix the errors occurred during their work.

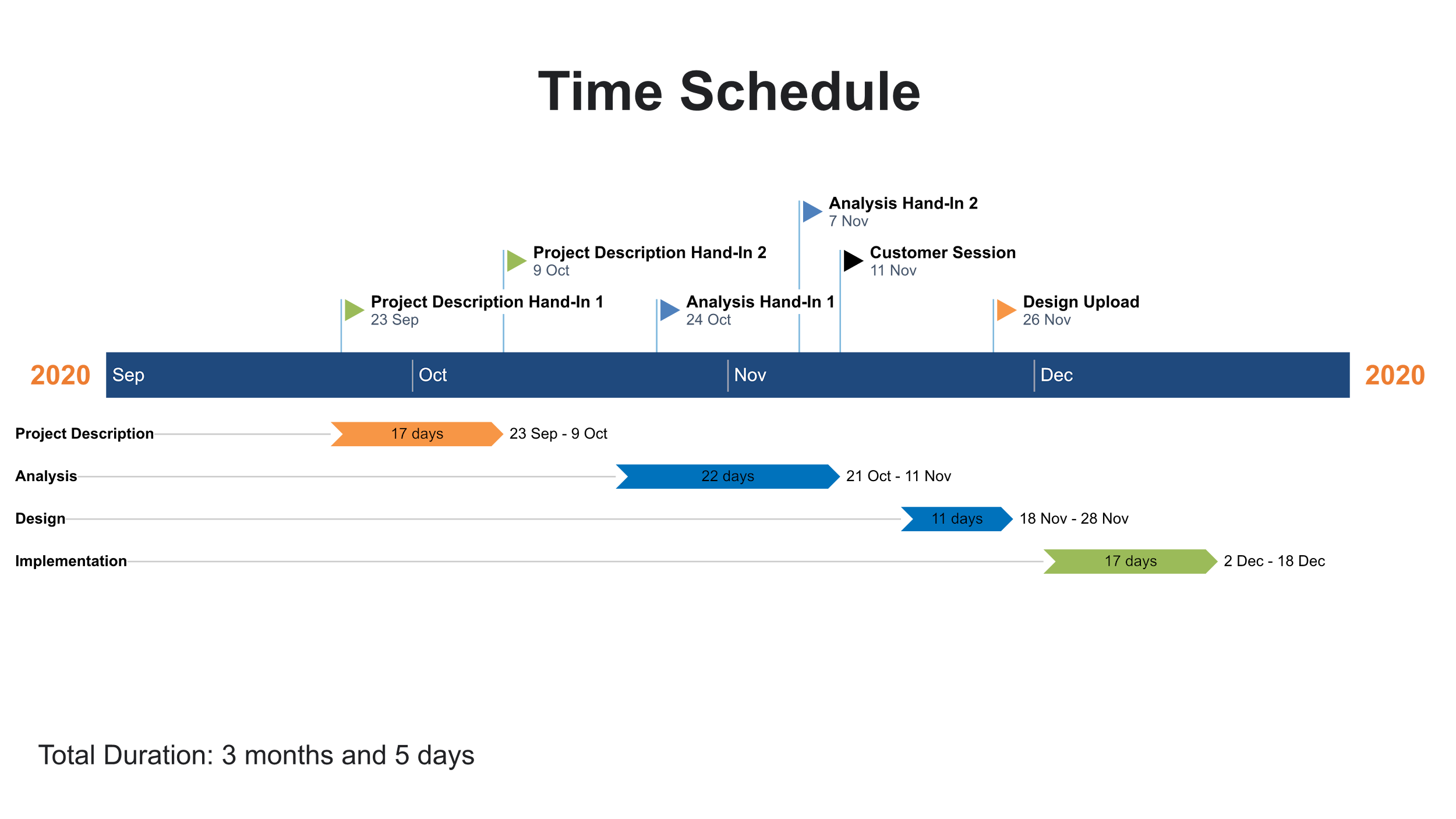


*Fig. 1. Waterfall method design process* (Koivukangas, 2015)

# 

# Time schedule

The expected workload is 27.5 hours per ECTS per student. And SEP1 is worth 5 ECTS so the total expected workload per student is 137.5 hours. The project spans 12 weeks so each member should spend 2.3 hours working on the project each workday of each week.



*Fig. 2. Gantt chart time schedule.*

Risk assessment

| Risk | Description | | Likelihood Scale: 1-5  (5 = high risk) | | Severity Scale: 1-5  (5 = high risk) | | Product of Likelihood and  Severity | | Risk Mitigation | | Responsible | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|
| Risk 1 | Lack of time before  hand-in/assignment | | 2 | | 3 | | 6 | | Assign internal deadlines for specific tasks,  more classwork. | | Everyone | |
|
| Risk 2 | Late hand-in/assignment | | 2 | | 3 | | 6 | | Assign internal deadlines for specific tasks. | | Everyone | |
|
| Risk 3 | Plagiarism | | 1 | | 5 | | 5 | | Reference all third party information/sources. | | Person who plagiarized work | |
|
| Risk 4 | Unequal workload | | 2 | | 3 | | 6 | | Divide tasks evenly between team members. | | Everyone | |
|
| Risk 5 | Crunch time | | 2 | | 3 | | 6 | | Complete most important tasks early on and  leave less significant work for the end. | | Everyone | |
|

# *Fig. 3. Risk assessment table.*

# 

# Sources of Information

Del I. Hawkins, David L. Mothersbaugh. 2010. Consumer Behavior: Building Marketing Strategy. 11th ed. New York: McGraw-Hill/Irwin

F. John Reh. 2019. *How to Successfully Manage Your First Project.* [online] Available at: <<https://www.thebalancecareers.com/how-to-successfully-manage-your-first-project-2276127>> [Accessed 8 Oktober 2020]

Felix Iblher, Andreas Oberlaender, Hendrik Willenbruch, Hugues Havrin. 2020. *It services: Who will win when demand bounces back*? [pdf] available at: <<https://www.oliverwyman.com/content/dam/oliver-wyman/v2/publications/2020/jun/IT_services_Who_Will_Win_When_Demand_Bounces_Back.pdf>> [accessed 8 Oktober 2020].

Michael Bloch, Sven Blumberg, and Jürgen Laartz. 2012. *Delivering large-scale IT projects on time, on budget, and on value.* (online) Available at: <<https://www.mckinsey.com/business-functions/mckinsey-digital/our-insights/delivering-large-scale-it-projects-on-time-on-budget-and-on-value#>> [Accessed 8 Oktober 2020]

Mordor Intelligence. 2020. *IT Services Market - Growth, Trends, and Forecast (2020 - 2025).* [online] Available at: <<https://www.mordorintelligence.com/industry-reports/it-services-market>> [Accessed 8 Oktober 2020]

Research Nester. 2020. *IT Managed Services Market: Global Demand Analysis & Opportunity Outlook 2023.* [online] Available at:

<<https://www.researchnester.com/reports/it-managed-services-market-global-demand-analysis-opportunity-outlook-2023/336>> [Accessed 8 Oktober 2020]

Royce, Winston. (1970). *Managing the Development of Large Software Systems*. [pdf] Available at: <<http://www-scf.usc.edu/~csci201/lectures/Lecture11/royce1970.pdf>> [Accessed 8 Oktober 2020]

Koivukangas Tapani. (2015). *Streamlining Medical Device Design Development Process from Research Laboratory to the Market.* [pdf] Available at: <<https://www.researchgate.net/publication/274570349_Streamlining_Medical_Device_Design_Development_Process_from_Research_Laboratory_to_the_Market>> [Accessed 17 November 2020]

# Appendices

Appendix 1. Group contract

**Group Contract**

***Group 4***

**Members:**

Aleksandrs Bistrovs **304542**

Henrik Koster **305916**

Kim Dahl Tranberg **172394**

Laurentiu Mihai **304456**

The following document has been developed and includes the code of conduct and cooperation between the team members of the group, where each member has to agree for the following terms and conditions as stated below:

**Participation**:

Strive for the best results, squeeze out maximum potential!

Invest the time, 2-3 hours each day - including the weekends.

Attend all the lectures, so everyone is on board.

Respect the team and their work, show interest for the project.

Try to divide the work equally.

**Communication**:

Talk about all the problems that arise during the project, try to find a solution between the members before calling in for help from the outside environment - supervisors.

Ask questions in case of doubt, make suggestions if there are any worthy ideas.

Give feedback to the team and be able to take constructive criticism from the peers.

**Meetings**:

Be serious and respectful towards the group members by showing up on the previously agreed meetings. Always come prepared and on time. In case of not being able to show up, always notify the team. Do not skip the meetings without a valid reason.

**Conflicts**:

When disagreements arise the team has agreed to solve them by a voting system. It means that the ideas which are backed up by the majority of the group are the ones which are taking place.

In case of any major conflicts or try to solve the problems by communicating within the group first. If that does not work - try to find a solution with the help of the supervisors.

**Deadlines**:

Respect the previously agreed time schedule and follow the deadlines. Try not to postpone the work until the last moment, thus making it sloppy and imprecise.

***Signatures: Date: 11.10.2020***