Project Report

Project Management System

Members of Group 8:

Valeriu Rosca	304191
Maxim Zavidei	304321
Jaime Elena	305950
Shaoqing Dai	305559

Supervisors:

Mona Wendel Andersen

Steffen Vissing Andersen

Software Technology Engineering

Autumn Semester

2020

Table of content

List of figures and tables

Optional

Abstract

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

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

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

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

Cf. (Dawson 2009, p.195).

1 Introduction

Color IT is an information technology company of small scale that offers its services to wide range of customers, mostly of them begin private. Their services consist of developing IT systems oriented to solve specific problems of their customers.

One of the biggest issues Color-IT currently faces is the lack of a Project management system to allow them to distribute information in an efficient way. Such poses a future threat for them that could potentially hinder the global performance of the company.

For instance, in their current state they are not allowed keep a reliable count of the hours invested on each of the tasks of the project, this could obstruct the delivery of a specific work since they would not be able to state a deadline for each of the labors. Moreover, it could also cause conflicts between the workers, since the lack of a management system produces miscommunication and it might turn into an unnecessary argument on a topic that was already agreed on.

The company is asking 1st semester students to develop a project management system, for internal use and an organized progress tracking website that would improve the efficiency of their workflow in the company. The system will not contain a log-in system, since the information will be accessed by both their costumers and the workers of ColorIT. Although the storing system was not specified by the costumer, he mentioned that it does not have to store the information in a data base. The system will also be only in English and no other languages will be supported.

Therefore, they are looking for a well-built system to improve their productivity at the company.

In order to start the begin the analysis the following delimitation will have to be set:

- There is no need of a log-in system.
- The information does not have to be specifically stored in databases.
- A multilingual platform is not required.

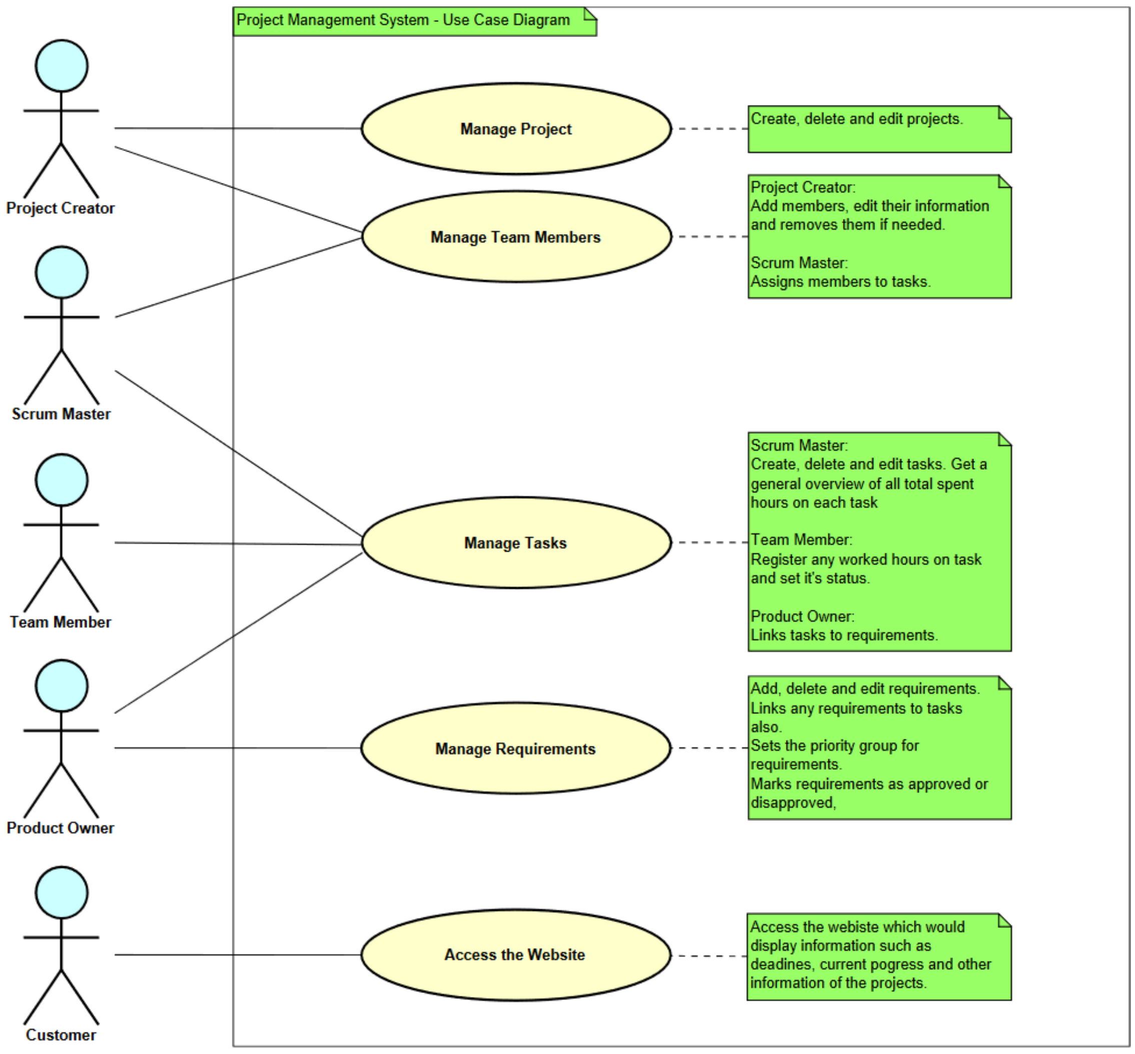
In order to get a clear overview of an efficient system for the customer that would meet their needs and solve their problem a comprehensive analysis of the requirements will be described with the following paragraph.

2 Analysis

In order to better understand what kind of system our customers it is of great interest to firstly analyze the way the operate in the company. This will represent valuable information that can be used to develop the system in a way such that it will closely match the workflow within the company, therefore increasing the chances of the final product matching the vision of the customer.

The company starts any project of their customers with an interview where they try to grasp the client's idea, followed by a list of requirements that the system needs to fulfil the specific task given by the costumer. The list created describes what features are needed, why are they relevant and who wants them and is later approved by the customer in a second interview. Within the company they might use the waterfall method or often time an iterative approach of developing the projects. Each of their project is separated in 3 different roles: Scrum master, product owner and team members. Each of the tasks has a team member as responsible, the other members of the team must report to this member the state of the specific task.

The following use case diagram showcases the relation between the actors and the functionality of the system.


Project Management System - Use Case Diagram
Project Creator
Scrum Master
Team Member
Product Owner
Customer
Manage Project
Manage Team Members
Manage Tasks
Manage Requirements
Access the Website
Create, delete and edit projects.
Project Creator: Add members, edit their information and removes them if needed.
Scrum Master: Assigns members to tasks.
Scrum Master: Create, delete and edit tasks. Get a general overview of all total spent hours on each task
Team Member: Register any worked hours on task and set it's status.
Product Owner: Links tasks to requirements.
Add, delete and edit requirements. Links any requirements to tasks also. Sets the priority group for requirements. Marks requirements as approved or disapproved,
Access the webiste which would display information such as deadines, current pogress and other information of the projects.

The company needs to have a project management system which will facilitate the workflow of the members on their tasks, and a website for the customer to follow progress of his project.

That is why the system has 5 main functionalities:

1) The Project Creator is responsible for managing the projects.
2) Both the Project Creator and Scrum Master can manage the team members.
3) The Product Owner, Scrum Master, and Team Member are responsible for managing the tasks
4) The Product Owner is managing and approving the requirements.
5) The Customer has access to the website.

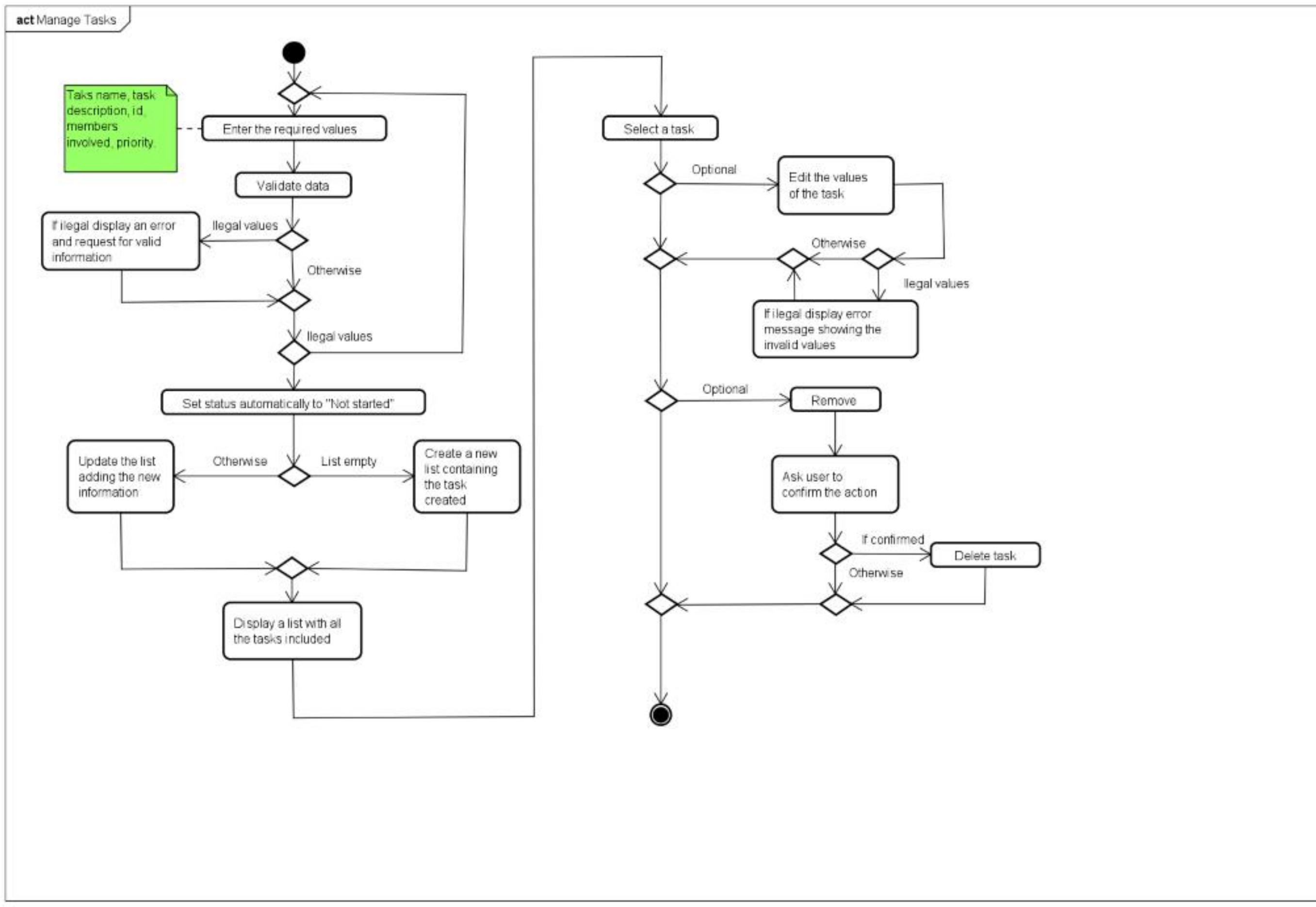
The requirements of the system were covered in the use cases presented in the diagrams above. One of the most crucial use cases is described below:

Use Cases:	Manage Tasks
Summary:	Add and show tasks information, edit task description and status and remove task.
Actor:	Scrum Master, Team Member
Precondition:	
Postcondition:	A new task has been added to the system and linked with a determined id to a specific requirement.
Base sequence:	**<u>ADD:</u>** 1) If adding a new task enter the following values: a) Task name b) Task description c) Estimated work hours d) Deadline 2) If the title is longer than 14 characters ask to input another title. 3) If the deadline is before today or after project's deadline ask to input another deadline. 4) If valid a new task is added to the list of tasks of a specific project. End the use case for **<u>ADD</u>** **<u>SHOW:</u>** 5) Show a list with all the tasks with the name, description, id, priority and members assigned. 6) When a task is selected show the description, priority, members and id specific to the new selected task. 7) If remove step 11. End the use case for **<u>SHOW</u>** **<u>EDIT:</u>**

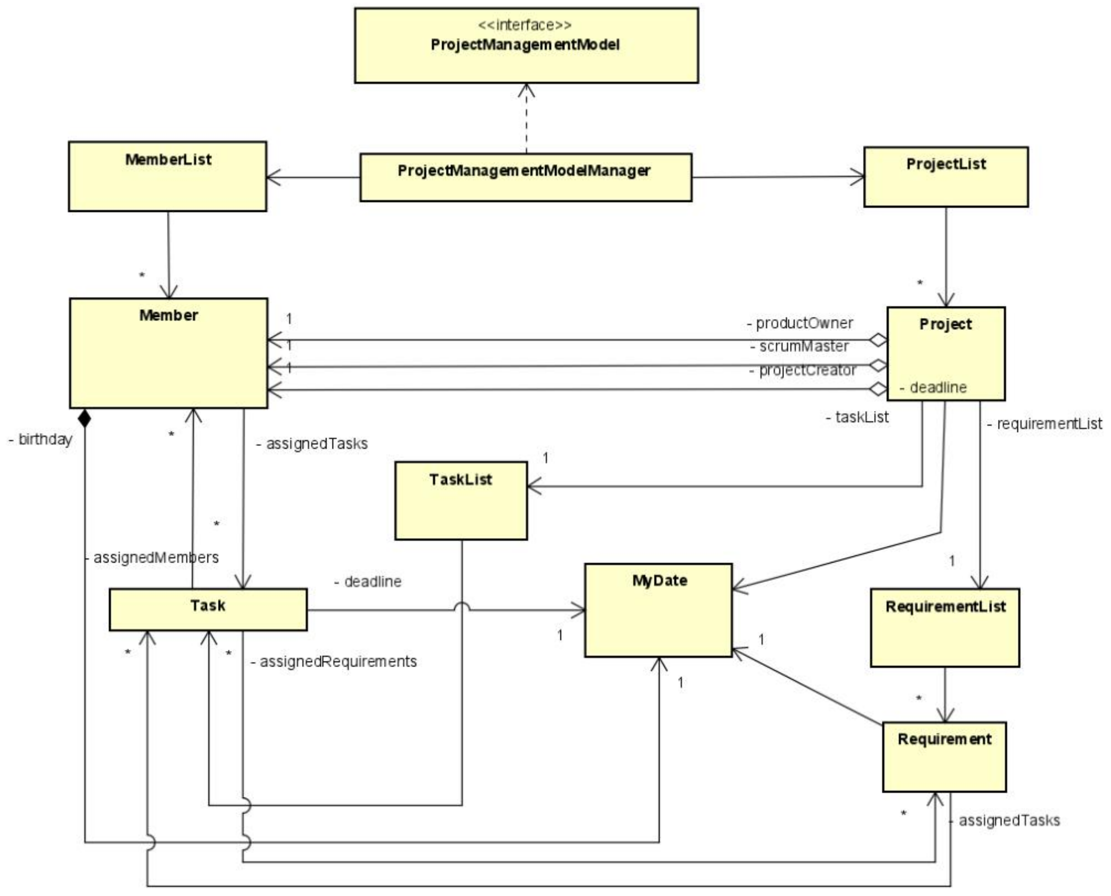
	8) Edit one or more of the values shown in step 1. 9) System verifies input, if wrong show the user what values need to be changed and go back to step 8. 10) System updates the new values given to a task in the task list. End the use case for **<u>EDIT</u>** **<u>REMOVE:</u>** 11) Select task 12) If the task has any linked requirements or is has any assigned members show an error message. 13) If not remove that task. End the use case for **<u>REMOVE</u>** **<u>LINKREQUIREMENT</u>** 14) Select a requirement. 15) Select a task to link to. 16) Check if requirement in not already linked to that task. 17) Requirement is linked to that task. End the use case for **<u>LINKREQUIREMENT</u>**
Exception sequence:	
Note:	This use case covers requirements 3, 4, 5, 7, 8, 10, 11, 12, 13, 15, 17.

The other use cases can be found at Appendix A.

For a better understanding of how the system works, a series of sequence diagrams were created.


act Manage Tasks
Taks name, task description, id, members involved, priority
Enter the required values
Validate data
If ilegal display an error and request for valid information
Ilegal values
Otherwise
Ilegal values
Set status automatically to "Not started"
Update the list adding the new information
Otherwise
List empty
Create a new list containing the task created
Display a list with all the tasks included
Select a task
Optional
Edit the values of the task
Otherwise
Ilegal values
If ilegal display error message showing the invalid values
Optional
Remove
Ask user to confirm the action
If confirmed
Delete task
Otherwise

For other sequence diagrams check Appendix A.
In the sequence above it can be seen what steps need to be taken in order to manage a task.
After starting all the use cases the following domain model shows the relationship between classes.



2.1 Requirements

From the above summary of their workflow, it could be inferred that the any activity related to projects is performed by members with special roles specifically Project Creator, Product Owner, Scrum Master, and Team Member.
The Project Creator is responsible for creating, editing and deleting projects and also for adding, editing and removing any members.

The Product Owner is responsible for creating, editing and deleting new requirements. Mark them as approved or rejected and also assigning requirements to tasks.
The Scrum Master creates, edits and deletes new tasks. Sets role to members in the project. And links tasks to requirements.
As a result, 4 essential objects may be identified a member, a project, a requirement or a task. In order to create a logical linking and a rational understating of how they interact together to following requirements were established:

2.2 Functional Requirements

1. *As a Project Creator I want to be able to* add new projects, edit their information and delete any of them if needed.
2. *As a Project Owner I want to be able to* add new requirements within any project, edit their information and delete any of them if needed.
3. *As a Scrum Master I want to be able to* add new tasks within any project, edit their information and delete any of them if needed.
4. *As a Project Owner I want to be able to* link or unlink any requirement to one or more tasks within the same project.
5. *As a Project Owner I want to be able to* link or unlink any task to one or more requirements within the same project.
6. *As a Project Creator I want to be able to* add new members, edit their information and delete any of them if needed.
7. *As a Scrum Master I want to be able to* assign any member to any number of tasks of any project.
8. *As a Scrum Master I want to be able to* assign special roles (Project Creator, Product Owner, Scrum Master) to any member within a project.
9. *As a Product Owner I want* each requirement to have a priority group (“Critical”, “High” or “Low”) *so that we can keep track of the more important requirements that need to be developed.*
10. *As a Team Member I want to be able to* edit the status of each one of the tasks, so that each task is set to either “Started” or “Finished”.
11. *As a Scrum Master I want* each task to store the total amount of time any member has worked on it.

12. *As a Team Member I want to be able to* register my time worked on a specific task.

13. *As a Scrum Master I want* each task to input and store an estimated number work hours when creating it, *so that we have basic perspective of the amount of time needed to develop it.*

14. *As a Project Owner I want to be able to* state of a requirement and mark it as approved or rejected.

15. *As a Scrum Master I want to be able to* search for all the projects that a specific member is working on.

16. *As a Customer I want to have access to* the information related to the project, so that I can keep track of the general status of the project.

17. *As a Scrum Master I want each task* to have descriptions, in order to have a general idea of what each task is about.

18. *As a Costumer I want to be able to* access the home page at any time so I do not get confused when navigating.

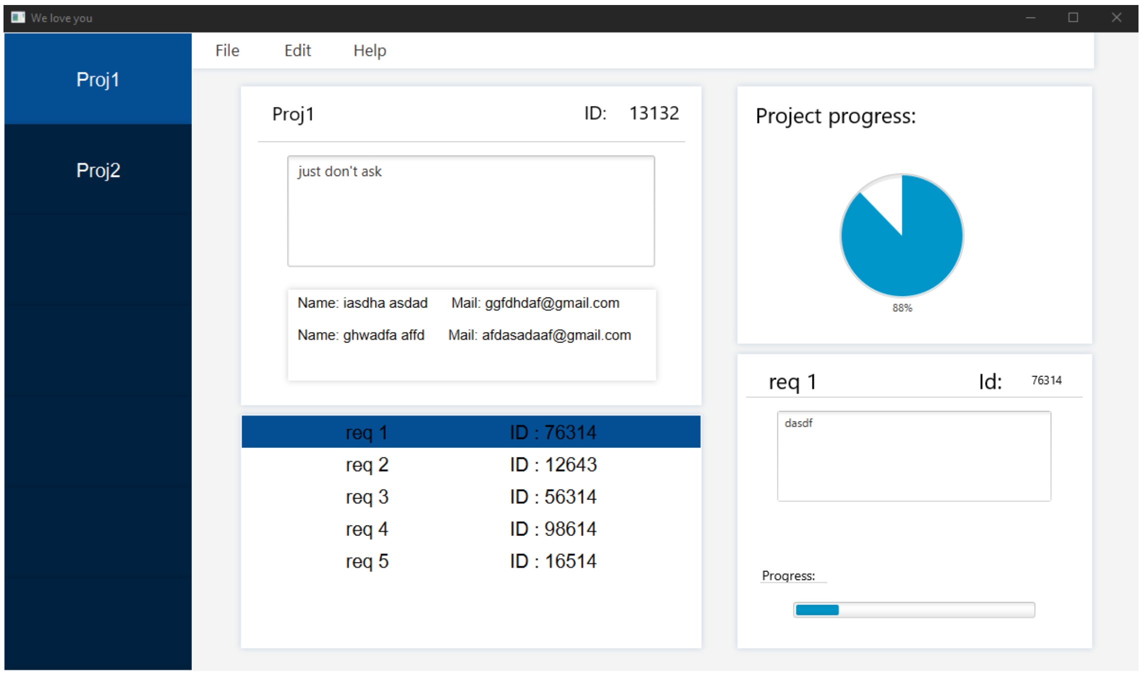
2.3 Non-Functional Requirements

19. *As a Customer I want* a manual explaining how the system works so that I won't have any issue understanding how the system functions.

20. *As a Costumer I want* the website to have a search feature.

3 Design

The projects will be displayed on the left



Proj1 ID: 13132

just don't ask

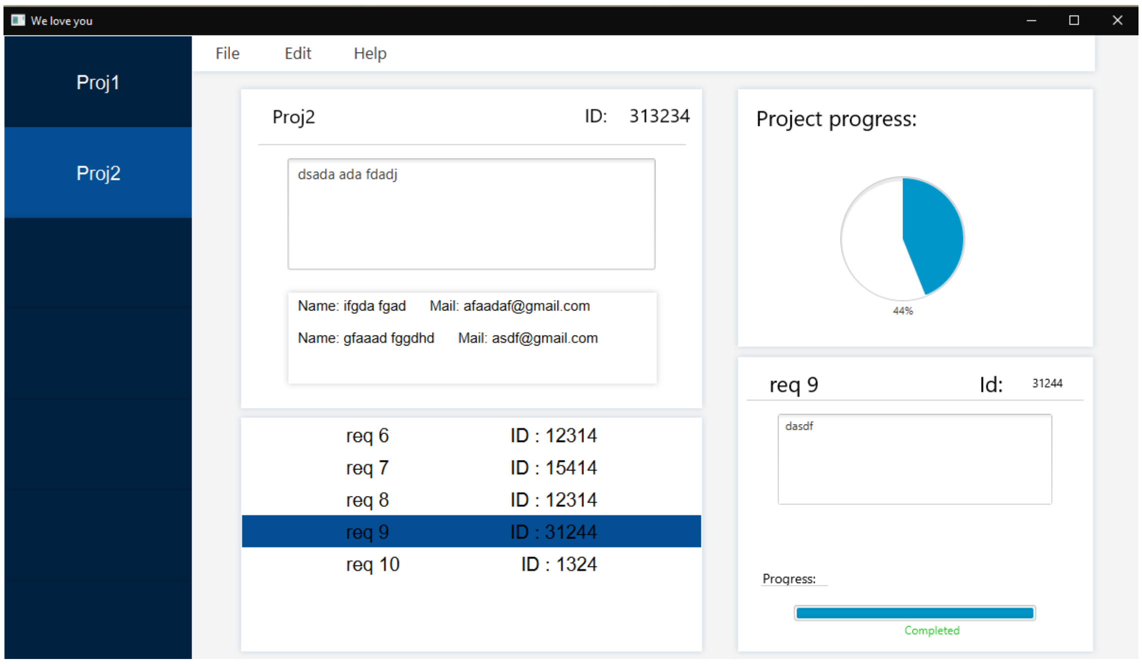
Name: iasdha asdad Mail: ggfdhdaf@gmail.com
Name: ghwadfa affd Mail: afdasadaaf@gmail.com

```
req 1    ID : 76314
req 2    ID : 12643
req 3    ID : 56314
req 4    ID : 98614
req 5    ID : 16514
```

req 1 Id: 76314

dasdf

Progress:



Proj2 ID: 313234

dsada ada fdadj

Name: ifgda fgad Mail: afaadaf@gmail.com
Name: gfaaad fggdhd Mail: asdf@gmail.com

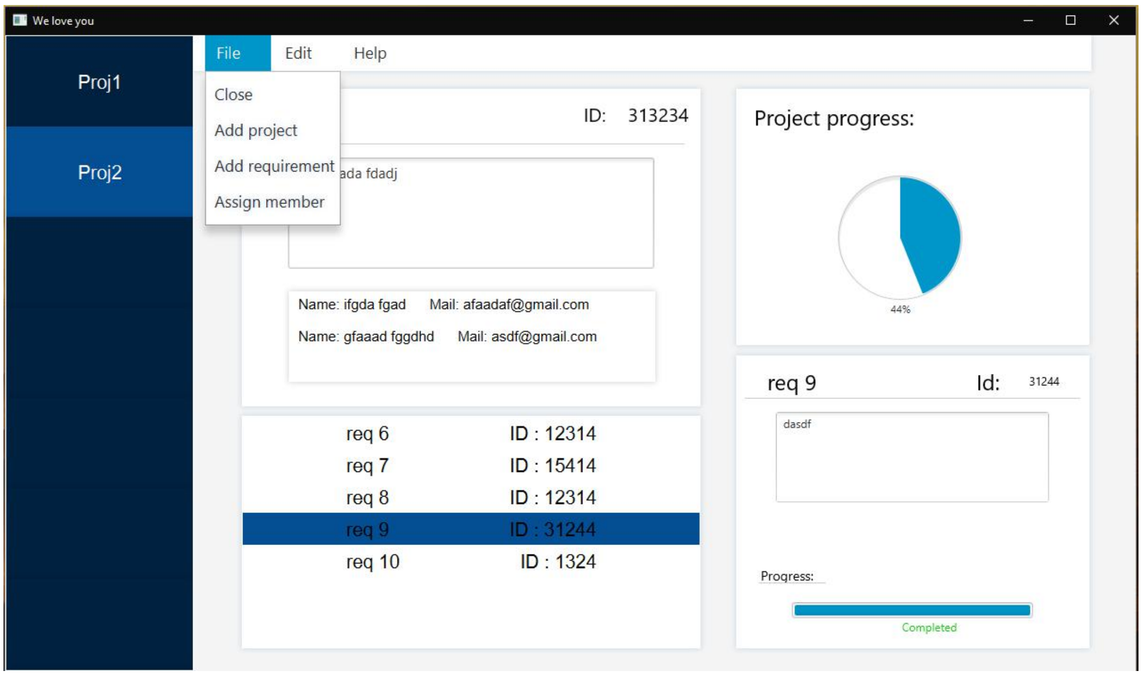
```
req 6    ID : 12314
req 7    ID : 15414
req 8    ID : 12314
req 9    ID : 31244
req 10   ID : 1324
```

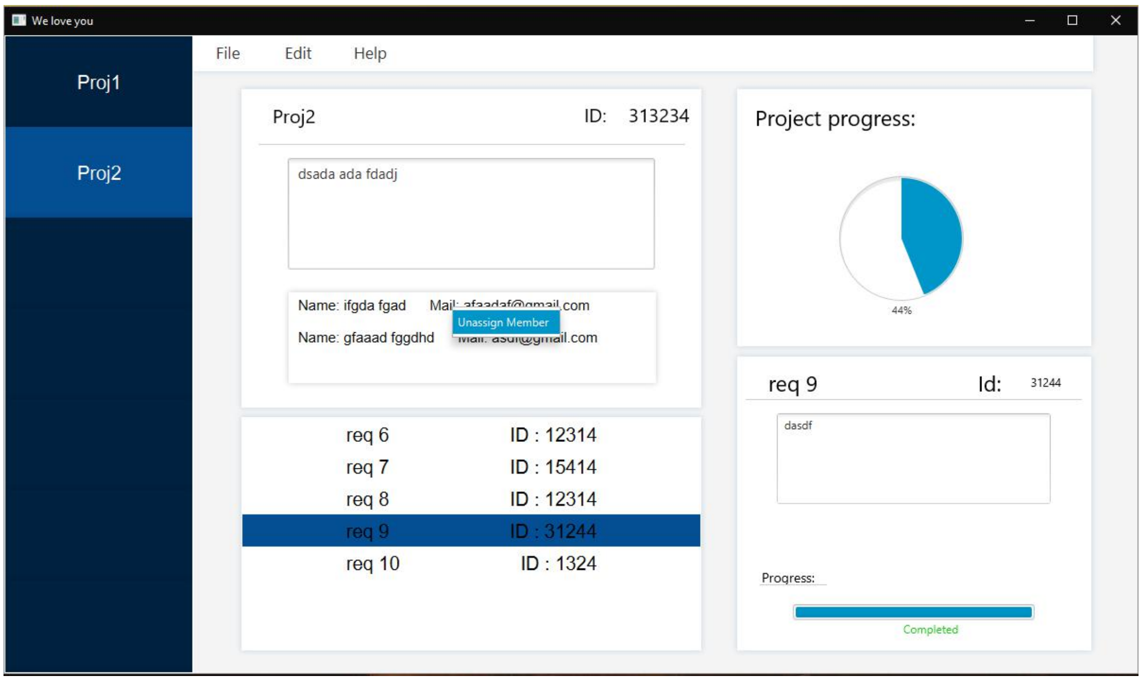
req 9 Id: 31244

dasdf

Progress:

Completed

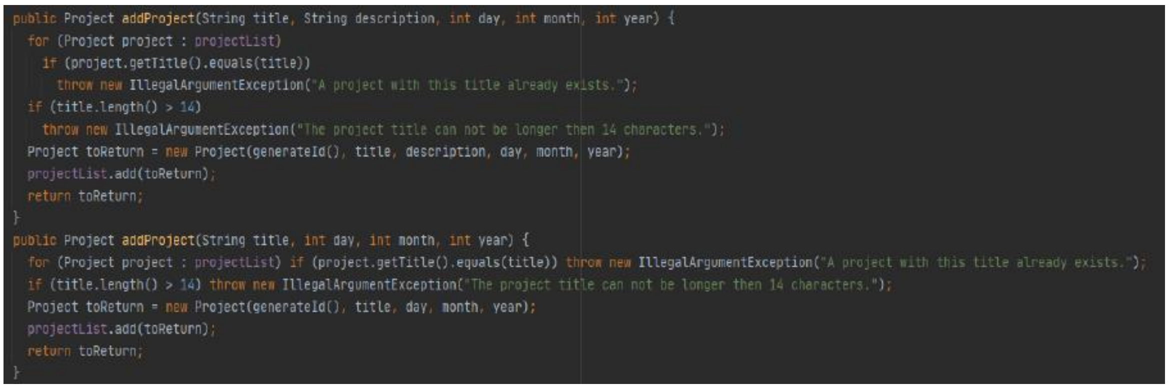

We love you
Proj1
Proj2
File
Edit
Help
Close
Add project
Add requirement
Assign member
ID: 313234
ada fdadj
Name: ifgda fgad Mail: afaadaf@gmail.com
Name: gfaaad fggdhd Mail: asdf@gmail.com
req 6 ID : 12314
req 7 ID : 15414
req 8 ID : 12314
req 9 ID : 31244
req 10 ID : 1324
Project progress:
44%
req 9 Id: 31244
dasdf
Progress:
Completed


We love you
Proj1
Proj2
File
Edit
Help
Proj2
ID: 313234
dsada ada fdadj
Name: ifgda fgad Mail: afaadaf@gmail.com
Name: gfaaad fggdhd Mail: asdf@gmail.com
Unassign Member
req 6 ID : 12314
req 7 ID : 15414
req 8 ID : 12314
req 9 ID : 31244
req 10 ID : 1324
Project progress:
44%
req 9 Id: 31244
dasdf
Progress:
Completed

4 Implementation

The project creator set as a requirement that the projects should be able to be add, edit their information and delete any of them if needed.

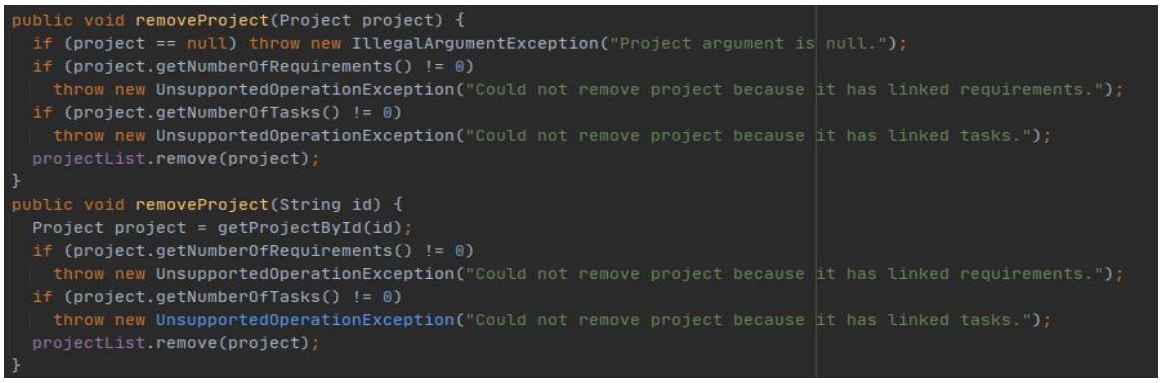
Given this fact, we have two part of the code in class projectList are the addProject(String title, String description, int day, int month, int year) and its other version addProject(String title, int day, int month, int year) with minimal number of defined values. They cover part of requirement 1, which is project creator can add projects.



```
public Project addProject(String title, String description, int day, int month, int year) {
  for (Project project : projectList)
    if (project.getTitle().equals(title))
      throw new IllegalArgumentException("A project with this title already exists.");
  if (title.length() > 14)
    throw new IllegalArgumentException("The project title can not be longer then 14 characters.");
  Project toReturn = new Project(generateId(), title, description, day, month, year);
  projectList.add(toReturn);
  return toReturn;
}
public Project addProject(String title, int day, int month, int year) {
  for (Project project : projectList) if (project.getTitle().equals(title)) throw new IllegalArgumentException("A project with this title already exists.");
  if (title.length() > 14) throw new IllegalArgumentException("The project title can not be longer then 14 characters.");
  Project toReturn = new Project(generateId(), title, day, month, year);
  projectList.add(toReturn);
  return toReturn;
}
```

- The first method takes a title, a description, a day, a month and a year as arguments. These will become information about new project to be added. This method will add a project with the information in system and return it. And it will throw exception when the title of new project same as someone already exists or its length over upper limit 14.
- The second method can add project with minimal number of defined values. It is mostly same as the first one but do not need to add description.

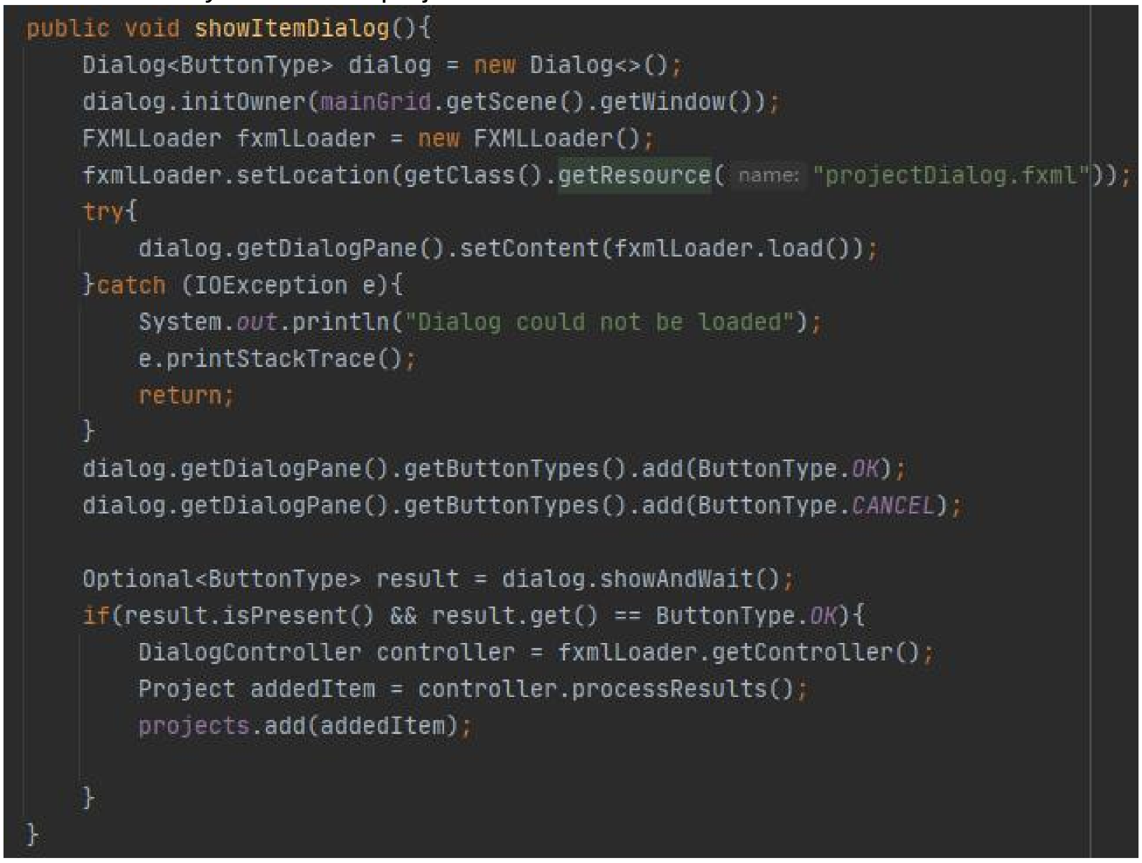
And we have two part of code also in the class projectList are the removeProject(Project project) and removeProject(String id). They cover the other part of requirement 1, which makes projects can be delete.



```
public void removeProject(Project project) {
  if (project == null) throw new IllegalArgumentException("Project argument is null.");
  if (project.getNumberOfRequirements() != 0)
    throw new UnsupportedOperationException("Could not remove project because it has linked requirements.");
  if (project.getNumberOfTasks() != 0)
    throw new UnsupportedOperationException("Could not remove project because it has linked tasks.");
  projectList.remove(project);
}
public void removeProject(String id) {
  Project project = getProjectById(id);
  if (project.getNumberOfRequirements() != 0)
    throw new UnsupportedOperationException("Could not remove project because it has linked requirements.");
  if (project.getNumberOfTasks() != 0)
    throw new UnsupportedOperationException("Could not remove project because it has linked tasks.");
  projectList.remove(project);
}
```

- The first method has an argument of a project. This method selects the project corresponding to the argument and delete it. But if that project not exists or link to any requirements or tasks, it would not be deleted and system will throw out exception.
- The second method is mostly same as the first one but it selects by id of project.

In the Controller, a method showItemDialog() is responsible for get and send the information to system to add projects.



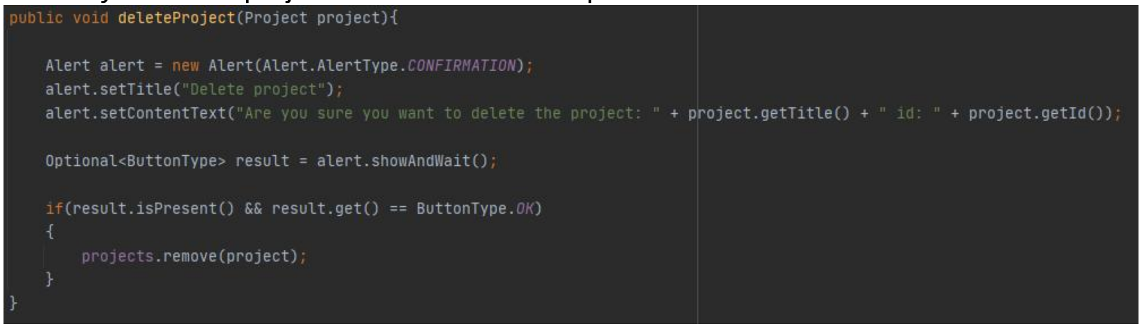
```
public void showItemDialog(){
    Dialog<ButtonType> dialog = new Dialog<>();
    dialog.initOwner(mainGrid.getScene().getWindow());
    FXMLLoader fxmlLoader = new FXMLLoader();
    fxmlLoader.setLocation(getClass().getResource( name: "projectDialog.fxml"));
    try{
        dialog.getDialogPane().setContent(fxmlLoader.load());
    }catch (IOException e){
        System.out.println("Dialog could not be loaded");
        e.printStackTrace();
        return;
    }
    dialog.getDialogPane().getButtonTypes().add(ButtonType.OK);
    dialog.getDialogPane().getButtonTypes().add(ButtonType.CANCEL);

    Optional<ButtonType> result = dialog.showAndWait();
    if(result.isPresent() && result.get() == ButtonType.OK){
        DialogController controller = fxmlLoader.getController();
        Project addedItem = controller.processResults();
        projects.add(addedItem);

    }
}
```

This method opens a dialog, and user can fill in the information needed, and it would create new project and put all the information in.

And another method deleteproject(Project project) in Controller can show an alert when try to delete project and delete it when press OK.



```
public void deleteProject(Project project){

    Alert alert = new Alert(Alert.AlertType.CONFIRMATION);
    alert.setTitle("Delete project");
    alert.setContentText("Are you sure you want to delete the project: " + project.getTitle() + " id: " + project.getId());

    Optional<ButtonType> result = alert.showAndWait();

    if(result.isPresent() && result.get() == ButtonType.OK)
    {
        projects.remove(project);
    }
}
```

5 Results and Discussion

As the part of the test shown, all the requirements functional and the system is completely working. All of the functions are working well, add or change things in system are responsible and efficient.

6 Conclusions

Color It asked for a system in order to manage their project since, up to that point they did not have any way to manage their projects.

Every single project starts with a personal interview with the costumer, where they try to grasp the client's idea about the project, once this part is completed a list of functional requirements is created. These requirements have to state who wants a specific feature and why is that feature relevant to a specific project. In addition to that there's another interview to confirm that that's what the client works and if not make changes.

The projects needs were stated by a series of requirements, these were divided in 2 big sections, functional and non-functional requirements, and within these main groups they were sorted by the priority of a specific requirement, dividing them into critical-priority, high-priority, and low priority. This was a crucial part since the following part of the system was based on this section and therefore any error could hinder the final result.
The analysis section formed the foundation of the entire project and the functionality of the system. The main part of the analysis was the case diagram, where the relation between the users and the cases of the system is formed, giving way to the general structure of the system. All the use cases were represented visually in the use case diagram, moreover the use cases covered all requirements. The way each use case worked was expressed in the use case description and later it was represented in the activity diagram.
In the design section we stated how we should create the system, how should we merge it, what type of interface we should use and what type of files were needed to fulfil the requirements.

The UI was designed with a series of listViews that would update and show the information of the projects on the left side, and when a project is clicked it would show all the data linked to that specific project on the right such as name, id, description, status list of members involved and list of requirements needed to complete the project.
In order to save the data and make use of it in the system it was decided to use both binary an xml files since they would facilitate its usage also on the website and the system itself.
The test didn’t follow a linear path but instead, due to lack of time it was roughly tested entering generic parameters for each one of the fields needed, but in the end it proved to be working accordingly to what we thought in the beginning.
Although not all the features were included in the system, it went through all the necessary steps to make a working system meeting the functionality that was ask for the project to have.

7 Project future

For further development of the project a search feature could be added in order to make items easier to find, overall in case there are many projects stored in a future. Moreover, the current IO system is thought for a single user when, realistically a company would rarely ask for this type of systems to be develop. Therefore the implementation of databases would make this feature possible and be more efficient than the current system that stores all the data in either binary or xml files.

Related to the UI the system could be more efficiently distributed and, instead of showing all the information in one tab a new tab could have more tabs with the information of a selected requirement or task.

Related to the methods and the current division of array lists could be improved, reducing the number if arrays in the system and improving the connection between arrays and classes. Also some of the methods could be implemented in a different way in order to make them more efficient therefore having a faster system.