Software Requirements Specification

# Project Tracker

### Version 1.0

### Rafel Salem

### 7/1/2020

**Table of Contents**

[​ Project Tracker 1](#__RefHeading___Toc1172_2695903106)

[​ Version 1.0 1](#__RefHeading___Toc1174_2695903106)

[​ Rafel Salem 1](#__RefHeading___Toc1176_2695903106)

[​ 7/1/2020 1](#__RefHeading___Toc1178_2695903106)

[​ 1. Introduction 3](#__RefHeading___Toc1180_2695903106)

[​ 1.1 Purpose 3](#__RefHeading___Toc1182_2695903106)

[​ 1.2 Intended Audience and Reading Suggestions 3](#__RefHeading___Toc1184_2695903106)

[​ 1.3 Document Conventions 3](#__RefHeading___Toc1186_2695903106)

[​ 1.4 Project Scope 3](#__RefHeading___Toc1188_2695903106)

[​ 1.5 References 3](#__RefHeading___Toc1190_2695903106)

[​ 2. Overall Description 3](#__RefHeading___Toc1192_2695903106)

[​ 2.1 Product Perspective 3](#__RefHeading___Toc1194_2695903106)

[​ 2.2 Product Functions 4](#__RefHeading___Toc1196_2695903106)

[​ 2.3 User Classes and Characteristics 4](#__RefHeading___Toc1198_2695903106)

[​ 2.4 Operating Environment 4](#__RefHeading___Toc1200_2695903106)

[​ 2.5 Design and Implementation Constraints 5](#__RefHeading___Toc1202_2695903106)

[​ 2.5 Assumptions and Dependencies 5](#__RefHeading___Toc1204_2695903106)

[​ 3. System Features (Specific Requirements) 5](#__RefHeading___Toc1206_2695903106)

[​ 3.1 Functional Requirements 5](#__RefHeading___Toc1208_2695903106)

[​ Manager Mode (Use Cases) 5](#__RefHeading___Toc1210_2695903106)

[​ Developer Mode (Use Cases) 8](#__RefHeading___Toc1212_2695903106)

[​ Client Mode (Use Cases) 9](#__RefHeading___Toc1214_2695903106)

[​ 3.2 System Requirements 10](#__RefHeading___Toc1216_2695903106)

[​ 3.2.1 External Interfaces 10](#__RefHeading___Toc1218_2695903106)

[​ Manager Mode 10](#__RefHeading___Toc1220_2695903106)

[​ Developer Mode 14](#__RefHeading___Toc1222_2695903106)

[​ Client Mode 16](#__RefHeading___Toc1224_2695903106)

[​ 3.2.2 System Models 17](#__RefHeading___Toc1226_2695903106)

[​ Manager Mode 17](#__RefHeading___Toc1228_2695903106)

[​ Developer Mode 26](#__RefHeading___Toc1230_2695903106)

[​ Client Mode 30](#__RefHeading___Toc1232_2695903106)

[​ 3.3 Domain Analysis 32](#__RefHeading___Toc1234_2695903106)

[​ User 33](#__RefHeading___Toc1236_2695903106)

[​ Project 33](#__RefHeading___Toc1238_2695903106)

[​ Feature 33](#__RefHeading___Toc1240_2695903106)

[​ Note 33](#__RefHeading___Toc1242_2695903106)

[​ Team 33](#__RefHeading___Toc1244_2695903106)

[​ Chat 33](#__RefHeading___Toc1246_2695903106)

[​ Message 33](#__RefHeading___Toc1248_2695903106)

[​ 4. Nonfunctional Requirements 33](#__RefHeading___Toc1250_2695903106)

[​ 4.1 Performance Requirements 33](#__RefHeading___Toc1252_2695903106)

[​ Reliability 33](#__RefHeading___Toc1254_2695903106)

[​ Availability 34](#__RefHeading___Toc1256_2695903106)

[​ Security 34](#__RefHeading___Toc1258_2695903106)

[​ Portability 34](#__RefHeading___Toc1260_2695903106)

# 1. Introduction

## 1.1 Purpose

Creating large scale applications can be a daunting task. Without road map to follow, a project’s development can become slower, inefficient, and often times messy. This application aims to help make project creation an easier task for the multiple roles involved (managers, developers, and clients). This application aims to make the manager’s job easier by providing them control over multiple aspects of project creation and team management. Gives developers information to help them focus on development. And clients the ability to track progress and make requests to help them feel connected in the creation process.

## 1.2 Intended Audience and Reading Suggestions

There are multiple roles of people that can use this application. It is aimed towards project managers, developers, and clients.

## 1.3 Document Conventions

The document is written in English. It is divided using different heading sizes. The font style is consistent throughout the document (Times New Roman). The document is divided into sections and subsections using different styling (Title, Header 1, Header 2, Header 3, and body paragraphs).

## 1.4 Project Scope

The following applications features will be included in the first iteration of the project that aim to boost a developer’s productivity:

* A main window containing project details (team, features, project chat button) and history.
* A popup window containing features and their details (sub tasks, deadlines, etc…).
* A popup window containing notes left by other team members.

A database will be created to store projects, chats, notes, and users.

## 1.5 References

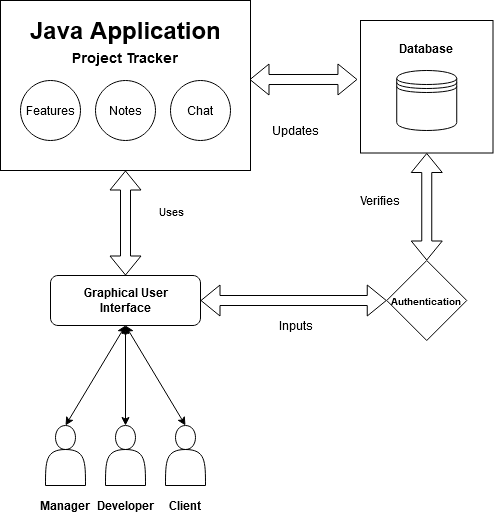
Included in the package are also an Analysis and Design Document and a Testing Plan Document.

* Oracle. (n.d.-a). *Download Free Java Software*. Java Runtime Environment. Retrieved July 13, 2020, from https://www.java.com/en/download/

# 2. Overall Description

## 2.1 Product Perspective

The program will run using the Java Virtual Machine. Popular progress trackers usually include a way to assign what a person does for a given day, allow the person to track their progress on the given feature, and allow communication about the project between team members.



## 2.2 Product Functions

The application will have the following features:

1. Project – displays project information and history.
2. Features – area that is viewed via button click.
3. Chat – chat area that is viewed via button click.
4. Manager mode – allows the manager to assign teams and features.
5. Developer mode – allows the developer to view project details, view features, leave notes on features, and declare a feature completed (pending the manager’s approval).
6. Client mode – allows the clients to view project progress and information.

## 2.3 User Classes and Characteristics

There will be three user classes on the first iteration of the program. These will be the manager, the developer, and the client.

Manager – interacts with most of the application features. Oversees the development of the project. Responsibilities range from assigning features to team members to tracking their progress.

Developer – interacts with some of the application features. Works towards finishing features. Responsibilities include viewing project details, updating feature status, and communicating through chat.

Client – interacts with the least amount of features. Views status of project and posts requests to backlog.

## 2.4 Operating Environment

The program will work on any OS as Java is OS independent. The program will have an SQL database to house relevant information.

## 2.5 Design and Implementation Constraints

The application needs to fit the needs of multiple types of users which can cause it to be bloated. Handling the different types of users will require different GUI for each type of user. Another challenge is whether the information will be available on an online or offline server. If it is on an offline server it will be more secure. If it on an online server it will provide more accessibility.

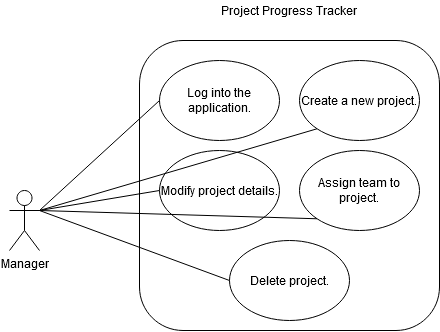
## 2.5 Assumptions and Dependencies

The application will contain different features based on the user type. A manager will have access to project details, features, and chat rooms in addition to other tools that allow them to facilitate the creation and completion of a project. A developer will have access to project details, features, and chat rooms. A client will have access to project details. As the application will be designed using Java, the user is expected to have installed the Java Runtime Environment. Each user type is expected to have experience with the tool sets provided to them.

# 3. System Features (Specific Requirements)

## 3.1 Functional Requirements

#### Manager Mode (Use Cases)

The application allows a manager control over project creation and project modification. As a new project is created, the manager opens the application and proceeds to enter their credentials. After the credentials are verified, the manager is granted an elevated privilege GUI that allows them to create a project. The manager enters the details of the project and clicks on create. The project is shown as an active project. The listing has an edit button, a delete button, and an assign members button. The edit button allows the manager to change the details of the project. The delete button removes the project listing AFTER confirmation. The assign members button allows the manager to assign members to complete the project.

**Manage Project**

Create projects, create features for projects, assign members to complete feature, edit project details, and delete a project. The create features button will popup a form the manager can fill to add a feature to the project. The form will contain information fields such as “name” of the feature, “description” of the feature, and an optional “deadline” field. Once the feature is created, the manager can click an icon next to the feature that allows them to assign the feature to a developer or team. If the client changes their mind on the project or it becomes retired (not maintainable), then the manager can choose to delete the project. There will be a verification popup to verify the deletion action.

Create project

1. The manager logs in by filling out their username and password and clicking login.
2. After verification, the manager is greeted with the projects page.
3. Click create project to open project creation form.
4. Fill out details and press submit button.

Create features

1. Click on project.
2. Click on create new feature for project.
3. Fill out details and press submit button.

Assign features

1. Click assign button.
2. Write in the text box who (developer or team) to assign to the feature.
3. Click on submit to save changes.

Edit project details

1. Click edit next to created project to edit details.
2. Click delete project.

**Manage Team**

Create teams and rearrange developers within teams or remove them. The create a team button will popup a window that allows the manager to fill in a team name and description. Once the team is created, the manager can assign members to the team from a roster list. If the team members need to be edited, the manager can click on edit in the team tab to change the current team configuration. They can add more team members or remove them by clicking the respective buttons. The manager can click submit to save the changes.

Create team

1. Click on “teams” tab.
2. Click create button.
3. Fill out form and list team members.
4. Click on submit.

Add member

1. Click on edit team.
2. Click on add member.
3. Type name of developer.
4. Click on submit.

Remove member

1. Click on delete.
2. Click yes on confirmation popup.

**Manage Chat**

View the chat, edit (delete certain messages), and clean chat. The manager can click on the teams tab and on the chat button. Once the chat window pops up, the manager is shown the chat with a different interface than the other modes that allows them to manage the chat. The manager can click on the delete button on a message to remove it. Or they can click on the clear button to clear all chat.

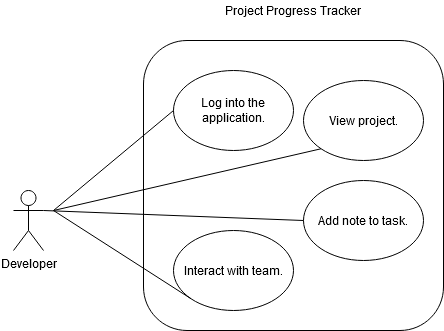
Remove message

1. Click on project or team.
2. Click chat.
3. Click delete button on single messages (only visible to managers).
4. Click yes on confirmation.

Clear chat

1. Click clear chat.
2. Confirm by typing “confirm” and clicking submit.

#### Developer Mode (Use Cases)

The application gives the developer tools to make their development more efficient and organized. When the developer logs into the application, they will be greeted with project details. They then view the project details and features they are in charge of. The features they’re assigned will also be available for them to check. As they work on the features they can leave notes for their teammates and management to check. When they complete their feature, they change the feature’s status to completed.

**View Project**

The developers can view project details. On log in, the developer will be greeted with a home screen. There will be a projects tab that a developer can click on once they log in. In the projects tab, it will have the current project in the main display. The window will have a list of desired features (with priority) and other project details such as client information, project goal / vision, technologies, and other details.

View project

1. Log into the application.
2. The main window will show the project and its details.

**Modify Features**

The developers can view features, write notes on features, and declare a feature is finished. By clicking on one of the features, they are shown a features detail screen in which they can view the feature details and leave notes. Clicking on the leave note button gives the developer a text box to type out their note and once they are finished writing the note they can click on submit to add it to the feature’s notes list. They can also change the feature’s status by click on the status button. The developer can declare a feature finished which will cause the status to change to pending. The manager will check the commit and can accept the declaration. Once the declaration is accepted, the status of the feature changes to completed.

Add note

1. Click on a specific feature in the feature field in the project details.
2. In the popup click on add note.
3. In the note field give it a title and description then click submit.

Change feature status

1. Click on status in the feature popup to declare a status change (will change to pending).

**Chat**

View members of the team and chat inside the chat room or send a team member a private message. The user can click on the team tab to gain access to a list of team members and the team chat. They can type in the chat text box and click submit to send their message to the team chat. The developer can also click on a chat icon next to a team member to send a private message to them. Once click, a popup with a text box and a submit button will show for them to write their message in and a send button to send it.

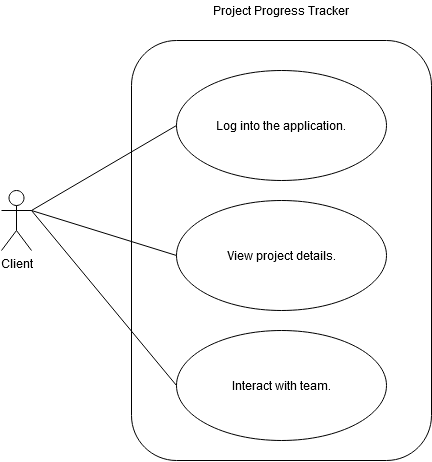
Chat with team

1. Click on the teams tab.
2. Click on the chat button.
3. Type a message in the text box under the chat messages and click submit.

Message member

1. Alternatively, click on the message icon next to a member of the team.
2. Type a message in the text box under the chat messages and click submit.

#### Client Mode (Use Cases)

Wondering how the project is coming along, the client logs into the application. They are then greeted with project details such as features in progress, features completed, and ETA for the project. As the client remembers a feature they forgot to mention click a button that allows them to access an area to add values or project feedback to help the management and team prioritize features.

**View project**

Allows the client to view project and details such as progress, features completed, and ETA of project. The client will be greeted with the project’s details once their credentials are verified.

View project

1. Log into the application.
2. After verification, a page with project details will appear.

**Request feature**

The client can submit features / components to the backlog for the team / management to prioritize and work on. The client can click on a request feature button which is only visible to this type of user in the project details page. Upon clicking the button, a popup will give them the ability to enter a feature name and a description. Once they finish filling the form, they can press the submit button to add the feature to the backlog.

Request feature

1. Click “request feature” button.
2. Fill the title and description of the form.
3. Click submit request.

## 3.2 System Requirements

### 3.2.1 External Interfaces

#### Manager Mode

**Manage Project**

The following table outlines the “Create” use case details:

|  |  |
| --- | --- |
| Use Case ID: 1 | Create Project |
| Description | Provides the manager the ability to start a new project. They will need to fill the project details in order to create a project. |
| Input | Button click and form submission. |
| Output | Project added to projects. |
| Preconditions | User must be logged in as manager. |
| Post Conditions | Projects will be updated. |
| Action | The application will add the project details to a data base table and display information from the database. |
| Frequency of Use | Once every four to nine months. |

The follow table outlines the “Create Features” use case details:

|  |  |
| --- | --- |
| Use Case ID: 2 | Create Features |
| Description | Allows the manager to add a feature to the project. |
| Input | Button click and form submission. |
| Output | Feature added to features list. |
| Preconditions | User must be logged in as manager. A project must exist to add a feature to. |
| Post Conditions | Features list will be updated. |
| Action | The application will add the feature and its details to a database table and display the information from the database in the feature list. |
| Frequency of Use | A few times a week or less. |

The following table outlines the “Assign Features” use case details:

|  |  |
| --- | --- |
| Use Case ID: 3 | Assign Features |
| Description | Provides the ability for the manager to assign developers to a feature. |
| Input | Form to collect developer name. |
| Output | Feature assigned to developer. |
| Preconditions | User must be logged in as manager. Member must exist. |
| Post Conditions | Task assigned to updated. |
| Action | The application will add the developer’s name to the tasks table and display it under task details. |
| Frequency of Use | Multiple times every four to nine months. |

The following table outlines the “Edit Project Details” use case details:

|  |  |
| --- | --- |
| Use Case ID: 4 | Edit Project Details |
| Description | Allows the manager to change the project’s details after the project has already been submitted. |
| Input | Button click and form. |
| Output | New details for project. |
| Preconditions | User must be logged in as manager. Project must already be created. |
| Post Conditions | The project’s details will be updated. |
| Action | The application will send the new project detail fields and to the database to change the old entries. |
| Frequency of Use | Once every four to nine months. |

**Manage Team**

The following table outlines the “Create Team” use case details:

|  |  |
| --- | --- |
| Use Case ID: 5 | Create Team |
| Description | The manager can create a team to work on a project. |
| Input | Button click and form. |
| Output | Team added to team’s list. |
| Preconditions | User must be logged in as manager. Project must already be created. |
| Post Conditions | Team’s list updated to represent the changes. |
| Action | The application will send the information to the team’s table and display the information on the application. |
| Frequency of Use | Once every four to nine months. |

The following table outlines the “Add Member” use case details:

|  |  |
| --- | --- |
| Use Case ID: 6 | Add Member |
| Description | Allows the manager to add a member to the project’s team. |
| Input | Button click. |
| Output | Adds developer to members list. |
| Preconditions | User must be logged in as manager. Project must already be created. |
| Post Conditions | The members list will be updated. |
| Action | The application will add the developer to the members table in the database. |
| Frequency of Use | One a week to once every nine months. |

The following table outlines the “Remove Member” use case details:

|  |  |
| --- | --- |
| Use Case ID: 7 | Remove Member |
| Description | The manager can remove a member from the team if circumstances change. |
| Input | Button click and a form. |
| Output | Member removed from members list. |
| Preconditions | User must be logged in as manager. Project must already be created. |
| Post Conditions | The members list will be updated. |
| Action | The application will update the database to remove the member from the members table. |
| Frequency of Use | Once a day to multiple times in nine months. |

**Manage Chat**

The following table outlines the “Remove Message” use case details:

|  |  |
| --- | --- |
| Use Case ID: 8 | Remove Message |
| Description | Allows the manager to remove a message from the team chat. |
| Input | Button click. |
| Output | Remove message from chat. |
| Preconditions | User must be logged in as manager. Project must already be created. Group must already be assigned to project. Message must be created. |
| Post Conditions | The message will be removed from the team chat. |
| Action | The application will remove the message from the chat table in the database. |
| Frequency of Use | Once or multiple times in nine months. |

The following table outlines the “Clear Chat” use case details:

|  |  |
| --- | --- |
| Use Case ID: 9 | Clear Chat |
| Description | Provides the manager with the ability to clear the team chat messages. |
| Input | Button click. |
| Output | Remove all team chat messages. |
| Preconditions | User must be logged in as manager. Project must already be created. Group must already be assigned to project. More than one message. |
| Post Conditions | All messages removed from team chat. |
| Action | The application will |
| Frequency of Use | Once or multiple times in nine months. |

#### Developer Mode

**View Project**

The following table outlines the “View Project” use case details:

|  |  |
| --- | --- |
| Use Case ID: 10 | View Project |
| Description | Allows the developer to view the current project. |
| Input | Button click. |
| Output | Displays the project details. |
| Preconditions | User must be logged in as developer. Project must already be created. Must be assigned to the project. |
| Post Conditions | Project details will be displayed. |
| Action | The application will process the button click and display the project details page. |
| Frequency of Use | Once or multiple times per day. |

**Add Note**

The following table outlines the “Add Note” use case details:

|  |  |
| --- | --- |
| Use Case ID: 11 | Add Note |
| Description | The developer can add a note to the feature they are viewing. |
| Input | Button click and form submit. |
| Output | Add note to note list. |
| Preconditions | User must be logged in as developer. Project must already be created. Feature must already be created. Must be assigned to the project. |
| Post Conditions | Note will be displayed in the note list. |
| Action | The application will store information from the note into the note table. |
| Frequency of Use | Once to multiple times a day / week. |

The following table outlines the “Change Feature Status” use case details:

|  |  |
| --- | --- |
| Use Case ID: 12 | Change Feature Status |
| Description | Provides the developer the ability to declare a task as completed. |
| Input | Button click. |
| Output | Store declaration and gray out button. |
| Preconditions | User must be logged in as developer. Project must already be created. Feature must already be created. Must be assigned to the project. |
| Post Conditions | The feature will be declared completed. |
| Action | The application will change a Boolean in the feature table in the database. |
| Frequency of Use | Once a day to once or more a week / month. |

**Chat**

The following table outlines the “Team Chat” use case details:

|  |  |
| --- | --- |
| Use Case ID: 13 | Team Chat |
| Description | A tab where a certain project’s team chat will be visible to the team members to discuss the project. |
| Input | Button click. |
| Output | Display team chat tab. |
| Preconditions | User must be logged in as developer. Project must already be created. Must be assigned to the project. Must be assigned to the team. |
| Post Conditions | The chat for the team will be displayed. |
| Action | The application will process the click on the team chat button and switch tabs to the team chat. |
| Frequency of Use | Once or multiple times per day. |

The following table outlines the “Message Member” use case details:

|  |  |
| --- | --- |
| Use Case ID: 14 | Message Member |
| Description | Allows the developer to send messages to members of the team individually. |
| Input | Button click to open send message form and fill out details of message and click submit. |
| Output | Send message to team member. |
| Preconditions | User must be logged in as developer. Project must already be created. Must be assigned to the project. Must be assigned to the team. |
| Post Conditions | Message will be send to the team member. |
| Action | The application will open a form for the developer to write out message and submit message. This will be stored in the messages table in the database. |
| Frequency of Use | None, once, or multiple times a day. |

#### Client Mode

**View Project**

The following table outlines the “View Project” use case details:

|  |  |
| --- | --- |
| Use Case ID: 15 | View Project |
| Description | The client can view the project and its details to see how it is developing. |
| Input | Log in. |
| Output | A project page containing its details. |
| Preconditions | User must be logged in as client. They must have a project in the database. |
| Post Conditions | The project and its details will in display. |
| Action | The application will verify user type and show the client their view type. |
| Frequency of Use | Once or multiple times a day / week / month. |

**Request Feature**

The following table outlines the “Request Feature” use case details:

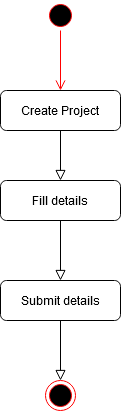
|  |  |
| --- | --- |
| Use Case ID: 16 | Request Feature |
| Description | Allows the client user to request a feature to be added to the project. |
| Input | Button click and form submit. |
| Output | Add request to backlog. |
| Preconditions | User must be logged in as client. They must have a project in the database. |
| Post Conditions | The backlog will contain the requested feature. |
| Action | The application will add the task to the backlog in the project table. |
| Frequency of Use | Once or multiple times a day / week / month. |

### 3.2.2 System Models

#### Manager Mode

**Create Projects**

When a client contacts the company and requests a new application, the manager can create a new project. The manager has to sign in with their credentials into the application. After they sign in, they will see the project page. On the project page there will be a create new project button. When they click the button a form will pop up for them to fill out the project information. Once they finish filling out the information, they can click the submit button to create the project. The project will be added to the database with all its relevant information.



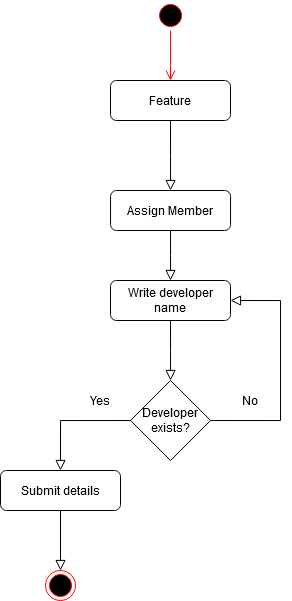
**Create Features**

Once the manager creates the project and fills out its details, they can add features to the project. This can be done by going to the project page and clicking the “create feature” button. A popup form will appear which can be filled out to provide details for the feature. Once it’s filled, the information can be saved and the feature created by clicking the submit button.



**Assign Features**

When a project has features, they can be assigned to group members. Once a project has been created and the features for the project are also created, the manager can assign a feature to one or more developers of the group. This is done by clicking the feature, and clicking assign member. The developer needs to exist, otherwise the application will ask for a valid input. Once the manager inputs a valid developer, they can click submit to assign the feature to the developer(s).



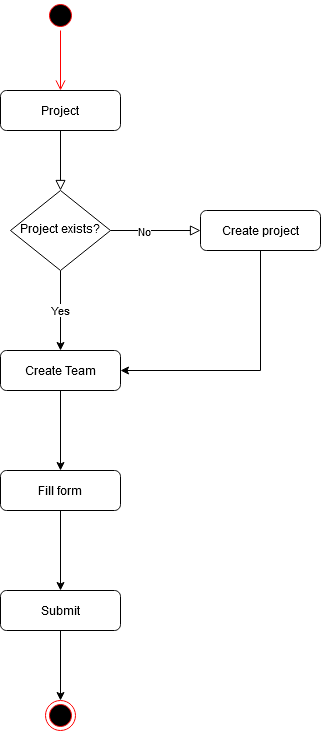
**Edit Project Details**

Once a project is created, a manager can change its details if they choose to. This is done by navigating to project and then clicking on “edit”. This will popup a form for them to fill out. After the form is filled out, the manager can click submit to save changes.



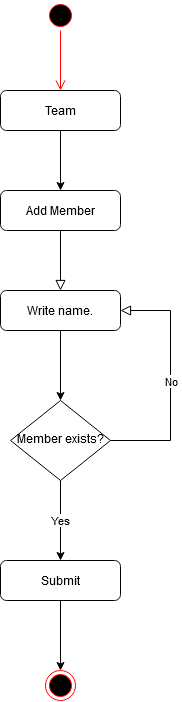
**Create Team**

A team can be created to be assigned to a project. Once the manager creates a project, he can also create a team for it to work on its features. This can be done by clicking the “create team” button on the project page. Then they will have to fill out the developers’ names that are going to be part of the team. The developers have to exist in the database for the manager to be able to add them to the team. After writing out the names, the manager will have to click “submit”.



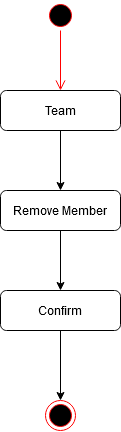
**Add Member**

Once a team is created, it will appear in the teams tab. From there, the manager can click “add member” button. This will ask them to write the developer’s name. The application will check if the developer exists, if it does not, it will not add the name to the database. After writing a valid developer name, the manager can click “submit” to add them to the team.



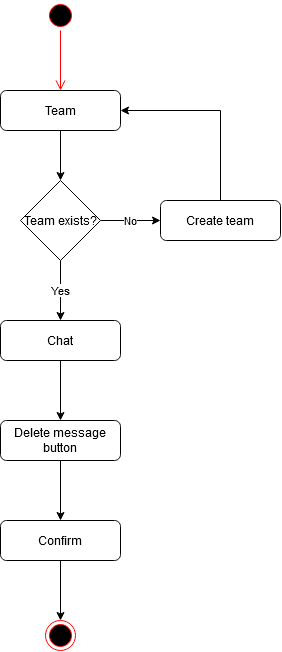
**Remove Member**

When circumstances change, a member needs to move or be removed, the manager can access this feature through the “teams” tab. Once they navigate there, they can click on the remove icon next to the developer’s name. They will have to confirm by reading a dialogue and clicking “accept”.



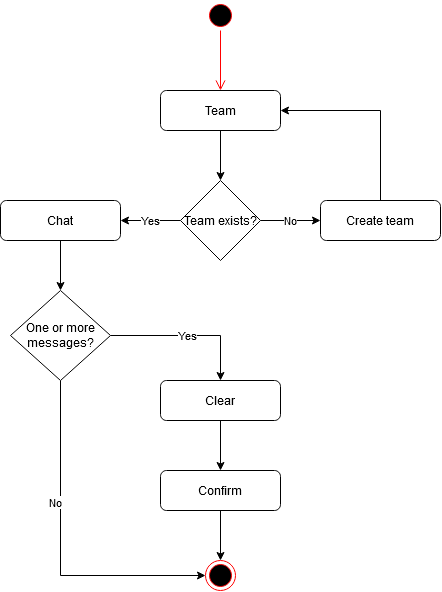
**Remove Message**

If there’s an unproductive message in the team chat, the manager can choose to remove it. First by going to the “teams’ tab. Then navigating to the “remove” button under the comment and clicking it. After this the manager will see a confirmation dialogue and need to click “accept” to remove the message.



**Clear Chat**

Should the need to clear the chat arise, the manager will have the ability to do so by first navigating to the “teams” tab. There has to be a team for the manager to be able to do this. There will also need to be more than one message in the chat. There will be a “clear chat” button in the bottom of the chat. Once the manager clicks the button, they will have to confirm their decision by clicking “accept”.



#### Developer Mode

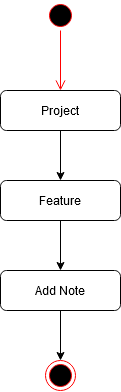
**View Projects**

For the developer to view their project, all they have to do is sign in. Once they sign in, they will be greeted with the project tab and its details.



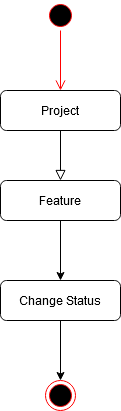
**Add Note**

The developer can add a note to the feature of the project they are working on. This note can be seen by other members or the manager so they can have more information about the task. To submit a note, the developer will have to be on the project page and click on the feature they want to leave a note on. Once the feature is displayed, they can click the ‘add note” button to add a note to the notes list of the feature.



**Change Feature Status**

Once the developer finishes working on a feature, they can choose to declare it as finished. This will let the team and the manger know of the feature status. To do this, the developer has to be on the project page and click on the feature they wish to declare as finished. On the feature page there is a status button that can be clicked to declare the feature finished. This will await the approval of the manager.



**Chat with Team**

A developer can communicate with their team through chat. To do this, they need to navigate to the “team” tab. Once on the team page, they can write their message in the text box on the bottom and click send.



**Message Member**

A developer can communicate directly to a single team member if they so choose. They can do this by going to the “teams” page and clicking the “message” button next to a team member’s name. They will then need to fill out the text box and click send to send them the message.



#### Client Mode

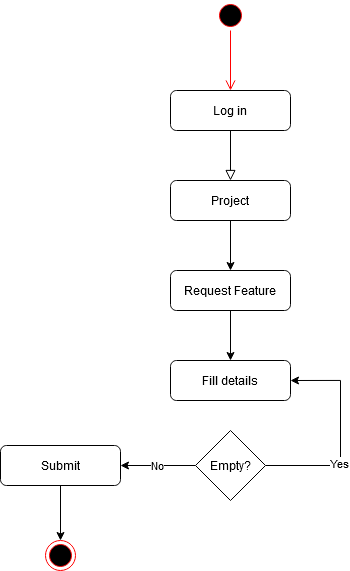
**View Project**

It is very easy for a client to see their project in development. All they need to do is sign in with their credentials and they will be greeted with their project page.

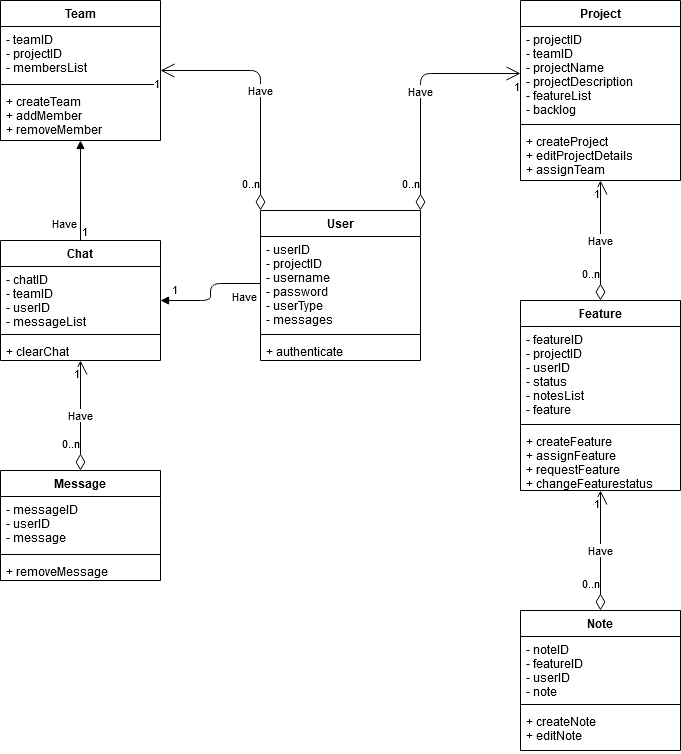


**Request Feature**

Should the client need to request an extra feature, they can do so directly from the project page. They can click the “request feature” button. Then they will have to fill out the request title and description and click “submit”. This will add the requested feature to the backlog.



## 3.3 Domain Analysis



### User

The user entity is the blueprint for the creation of different types of user classes. It helps create users with multiple properties. It also has an authenticate function to allow the user access to the application.

### Project

The project entity contains important properties of a project such as name, description, feature list, and other important properties. It contains a methods that allow for creation of a project, editing the details of an already created project, and assigning a team to the project.

### Feature

The feature entity is responsible for the creation of features of the project. It contains properties about the features such as the feature, its notes, and its status. It also allows the user to create, assign, or request a feature depending on their type.

### Note

The note entity is responsible for creating note objects that have properties such as the note, the user that left the note, and the feature the note belongs to. It also has methods that allow for creation and editing of existing notes.

### Team

The team entity is used to represent a development team that is assigned to a project. It has properties such as “teamID” to uniquely identify each team. “ProjectID” to help identify which project the team is assigned to. And a “membersList” to keep track of which developers are part of the team. It has methods to allow for the creation of a team and the addition / removal of members to and from the team respectively.

### Chat

The chat entity helps with storing the chat of the team. It includes properties that tie it to a team and user such as “userID” and “teamID”. It also contains a “clear chat” method that empties the message list.

### Message

The message entity is responsible for keeping track of what’s said and who said. It has properties such as “messageID” to uniquely identify a message, “userID” to identify who left the message, and “message” for the message content.

# 4. Nonfunctional Requirements

## 4.1 Performance Requirements

### Reliability

The application should be able to run without any errors. It should meet the requirements set for it in terms of functionality. The functionality should also be predictable. As such, the GUI should be intuitive and easy to follow / use.

### Availability

The application is mostly used offline so it will be available all the time. In terms of stability, it will be bug free and responsive. This should provide high up time and availability of the application for its users.

### Security

The content of the application will not be accessible without credentials. This provides protection of the project and its details from the physical security aspect. The application will use Spring Security framework to help with authenticating which will help make the application more secure from the digital security aspect.

### Portability

Since the program will be created using Java, it will have a high degree of portability because Java is a platform independent language. As long as the user has a JRE installed on their system, they can make use of the application.