15-Oct-22

Abstract

Submitted in partial fulfillment of the requirements of the software engineering course

Software Requirements Specification

For Schneider Productivity Manager

Contents

[Introduction 2](#_Toc116717974)

[Purpose 2](#_Toc116717975)

[Scope 2](#_Toc116717976)

[Glossary 2](#_Toc116717977)

[Document Overview 2](#_Toc116717978)

[Overall description 3](#_Toc116717979)

[System Environment 3](#_Toc116717980)

[System Context Diagram 3](#_Toc116717981)

[Use Case Diagram 3](#_Toc116717982)

[Functional Requirements Specification 4](#_Toc116717983)

[Update Project 4](#_Toc116717984)

[Update Project Task 5](#_Toc116717985)

[Assign Project Task to Employee 6](#_Toc116717986)

[Mark Project Task as done 7](#_Toc116717987)

[Non-Functional Requirements 8](#_Toc116717988)

# Introduction

## Purpose

The purpose of this document is to present a detailed description of the Productivity Manager. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli. This document is intended for both the stakeholders and the developers of the system.

## Scope

This system is a productivity enhancing tool that keeps track of ongoing projects, employees’ work utilization. This system is a web-based application that provides interfaces for various stake holders (project leaders, managers, employees).

This system’s admins can add new recruits to the employee pool where they’d be assigned to projects according to the respective project leader’s request. The admins can monitor the current work utilization of employees and view over-utilized and cross-utilized employees to better divide their workloads if possible and/or necessary. This system’s admins can also view the status and punctuality of projects. This system’s aggregates relevant and useful statistics to support data driven decision making regarding how workload is managed within the department.

## Glossary

|  |  |
| --- | --- |
| Term | Definition |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

## Document Overview

The Overall Description section of this document gives an overview of the functionality of the product. The system environment chapter describes the informal requirements and is used to establish a context for the technical requirements specification in the next chapter.

The Requirements Specification section of this document is written primarily for the developers and describes in technical terms the details of the functionality of the product.

Both sections of the document describe the same software product in its entirety but are intended for different audiences and thus use different language.

# Overall description

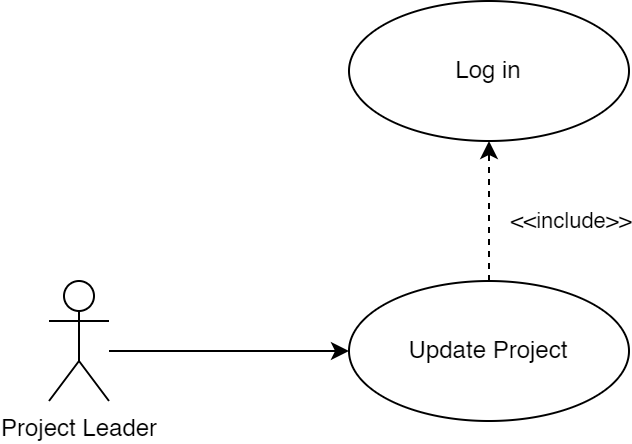
## System Environment

### System Context Diagram

### Use Case Diagram

## Functional Requirements Specification

### Update Project

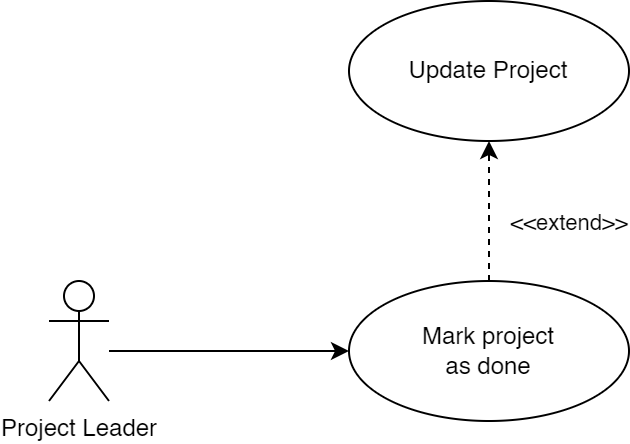


|  |  |
| --- | --- |
| Property | Value |
| ID, Name | Update project |
| Goal | Inputted changes should apply to the project table. |
| Initiator | Project Leader, Admin |
| Precondition | * A user with project leader or higher privileges should be logged in. * The project leader must be the team leader of this project. |
| Postcondition | * The task’s properties will be updated. * The new task data will be returned in response. |

#### Step-by-step Description

|  |  |
| --- | --- |
| 1. The actor submits an update request to update one of the tasks associated with one of his projects. 2. The system runs validation checks, upon the failure of any, throws an exception.    1. The system checks the actor’s login validity.    2. The system checks that the actor is either the project leader assigned to this project or a user with higher privileges.    3. The system verifies the input format of the request. 3. The system queries the database to update the given project’s properties with the new data. 4. The system returns a response carrying a success code and the updated project data. | Diagram  Description automatically generated |

### Mark Project as done

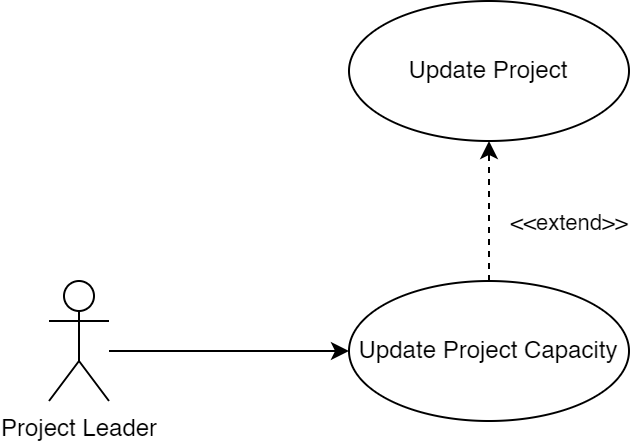


|  |  |
| --- | --- |
| Property | Value |
| ID, Name | Mark project as done |
| Extends | Update project |
| Goal | A project’s status is updated to done. |
| Initiator | Project Leader |
| Precondition | * A user with project leader or higher privileges should be logged in. * A project leader must be the team leader of this project. |
| Postcondition | * The project’s status will be updated. * The new project’s data will be returned in response. |

#### Step-by-step Description

|  |  |
| --- | --- |
| 1. The project leader submits an update request to update the status field of one of the projects that they’re leading. 2. The system runs validation checks, upon the failure of any, throws an exception.    1. The system checks the actor’s login validity.    2. The system checks that the inputted project’s leader is the logged in actor.    3. The system verifies the input format of the request. 3. The system queries the database to update the given project’s status with the done status that indicates that this project does not currently utilize any employees. 4. The system returns a response carrying a success code and the updated project data. |  |

### Update Project Capacity



|  |  |
| --- | --- |
| Property | Value |
| ID, Name | Update project capacity |
| Extends | Update project |
| Goal | A project’s status is updated to done. |
| Initiator | Project Leader |
| Precondition | * A user with project leader or higher privileges should be logged in. * A project leader must be the team leader of this project. |
| Postcondition | * The project’s capacity is updated. * The new project data will be returned in response. |

#### Step-by-step Description

|  |  |
| --- | --- |
| 1. The project leader submits an update request to update the capacity field of one of the projects associated with one of his projects. 2. The system runs validation checks, upon the failure of any, throws an exception.    1. The system checks the actor’s login validity.    2. The system checks that the inputted project’s leader is the logged in actor.    3. The system verifies the input format of the request. 3. The system queries the database to update the given project’s capacity with the inputted capacity that indicates that this project needs a given number of employees on its team. 4. The system returns a response carrying a success code and the updated project data. |  |

## Non-Functional Requirements