**Software Requirements Specification**

**for**

**Let’s Go Team!**

**Prepared by**

**Team Name: Letícia´s team**

Alexandre Lazaretti Zanatta <alexandre.zanatta@acad.pucrs.br>

Letícia Machado <leticia.machado.001@acad.pucrs.br>

**University: PUCRS - PONTIFICAL CATHOLIC UNIVERSITY OF RIO GRANDE DO SUL - FACULTY OF INFORMATICS**

**Country: Brazil**

**October 20, 2014**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author(s)** |
| October 08, 2014 | V1.0 | Document created | Graziela Basilio Pereira |
| October 20, 2014 | V1.0 | Document update | Caroline Queiroz Santos |
| October 20, 2014 | V1.0 | Document update | Alexandre Lazaretti Zanatta |
| October 20, 2014 | V1.0 | Document update | Letícia Santos Machado |
| October 20, 2014 | V1.0 | Document update | Graziela Basilio Pereira |
| November 05, 2014 | V2.0 | Document update  *(just non-functional requirements)* | Alexandre Lazaretti Zanatta |
|  |  |  |  |

# **1 Introduction**

This document describes and specifies the needs of the customer, being the main source of information for design development of a product to support distributed software development teams.

## 1.1 Purpose

This requirements specification fully describes the external behavior of the application. It also describes nonfunctional requirements, design constraints, and other factors necessary to provide a complete and comprehensive description of the requirements for the software.

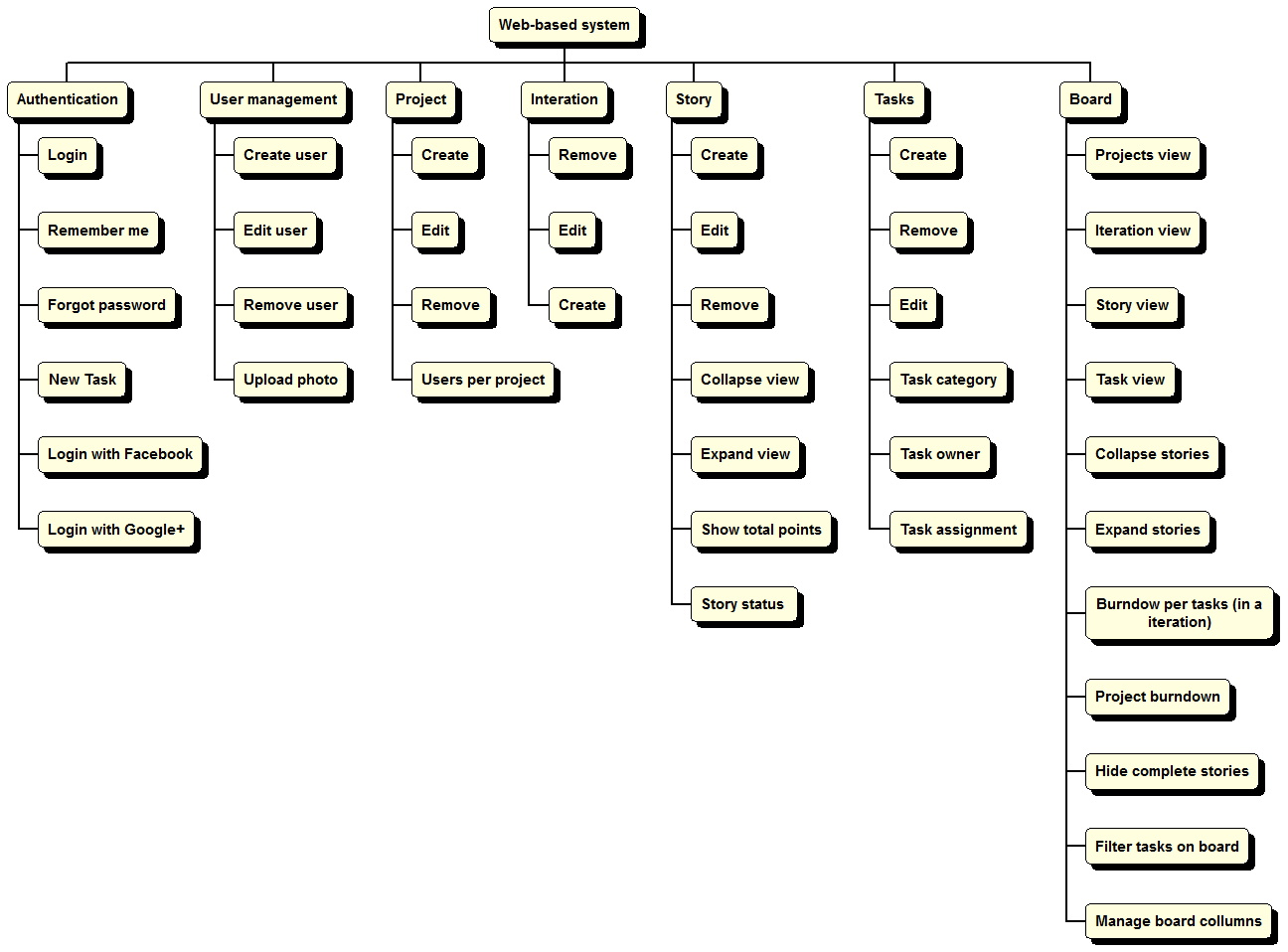
## 1.2 Scope

The scope of this project is to web-based project management system to support distributed software development teams.

● Project develops through successive iterations

● Team members, managers and other stakeholders need to record and track progress

The figure below shows the product structure and its scope.



## 1.3 Definitions, Acronyms and Abbreviations

UC: use case

Kanban: on going board.

## 1.4 References

|  |  |  |
| --- | --- | --- |
| Date | Document | Resources |
| August | [Project overview](http://moodle.pucrs.br/mod/resource/view.php?id=841735) | Profª Sabrina |
| September, 12 | Assignment #2, Part 1: To write clarification requestt | Clients  Time of analysts |

## 1.5 Overview

The SRS contains the description of the functional and non functional requirements about the systems inthe section 3.1 and 3.2. In the section 3.3 are presented the use case diagram and the user cases’s description. Finally, to improve the comprehension and supporting information the documentation brings the main screen system’s in the section 4.

# **2 Overall Description**

## 2.1 Product perspective

## 2.2 Product functions

**Basic Functions**

* Backlog: a collection of requirements that need to / cloud be be implemented
* Sprint log: a log that is used as part of a development sprint
* User management: users need to be able to register with the system; users need to be able to assign users to projects; basic user management (delete account, change personal details). Support for different roles (for example developers and ‘master’ -- similar to SCRUM master--). Functionalities are available depending on user roles.
* User authentication
* Users can be assigned to requirements; maybe requirements need to be broken down

into smaller tasks

* Requirements can be assigned with ‘points’ according to their complexity. One can also assign point to tasks. Users who implement the requirements/tasks get the ‘points’ of the implemented requirements.
* Developers chart: all developers are ranked according to the ‘points’ received from the implemented tasks/requirements.

**Advanced functionalities**

* A dashboard that summarizes ongoing events of (a) project(s)
* Statistics that show how the project performances, compared to the estimates
* Team communication tools (e.g. a discussion board or group chat)
* Integration with issue trackers (e.g. the tracker from the where the source code is
* hosted. Github, Bitbucket, etc) or other platforms

## 2.3 User characteristics

The system users will be software developers teams.

## 2.4 Constraints

The system does not provide:

* cost management
* people management
* mobile interface

## 2.5 Assumptions and dependencies

* A project can be created by any user.
* Work management will be done throughout the project team.

# **3 Specific Requirements**

## 3.1 Functional Requirements

|  |  |
| --- | --- |
| **Requirement ID** | [FR1] |
| **Title** | Authentication |
| **Description** | Authentication system for user login  Authorization services let users provide your application with access to read, write, update and delete the data they have stored in project management system. |
| **Priority** | Priority 1 - mandatory. |
| **Risk** | Critical (C) - It will break the main functionality of the system. The system cannot be used if this requirement is not implemented. |

|  |  |
| --- | --- |
| **Requirement ID** | [FR2] |
| **Title** | Task information |
| **Description** | The system shall provide tasks information like description, duration, resources and comments. |
| **Priority** | Priority 1 - mandatory. |
| **Risk** | Critical (C) - It will break the main functionality of the system. The system cannot be used if this requirement is not implemented. |

|  |  |
| --- | --- |
| **Requirement ID** | [FR3] |
| **Title** | Collaboration |
| **Description** | The system shall provide collaboration mechanisms like chat, comments |
| **Priority** | Priority 2 - feature nice. |
| **Risk** | High (H) - It will impact the main functionality of the system. Some function of the system could be inaccessible, but the system can be generally used. |

|  |  |
| --- | --- |
| **Requirement ID** | [FR4] |
| **Title** | Schedule |
| **Description** | The system shall provide schedule mechanisms like Gantt chart, calendar and time tracking. |
| **Priority** | Priority 2 - feature nice. |
| **Risk** | High (H) - It will impact the main functionality of the system. Some function of the system could be inaccessible, but the system can be generally used. |

|  |  |
| --- | --- |
| **Requirement ID** | [FR5] |
| **Title** | Resources agiles |
| **Description** | The system shall provide support agile methods like Burndown (project, sprint, release), Sprint backlog, Project backlog. |
| **Priority** | Priority 1 - mandatory. |
| **Risk** | Critical (C) - It will break the main functionality of the system. The system cannot be used if this requirement is not implemented. |

## 

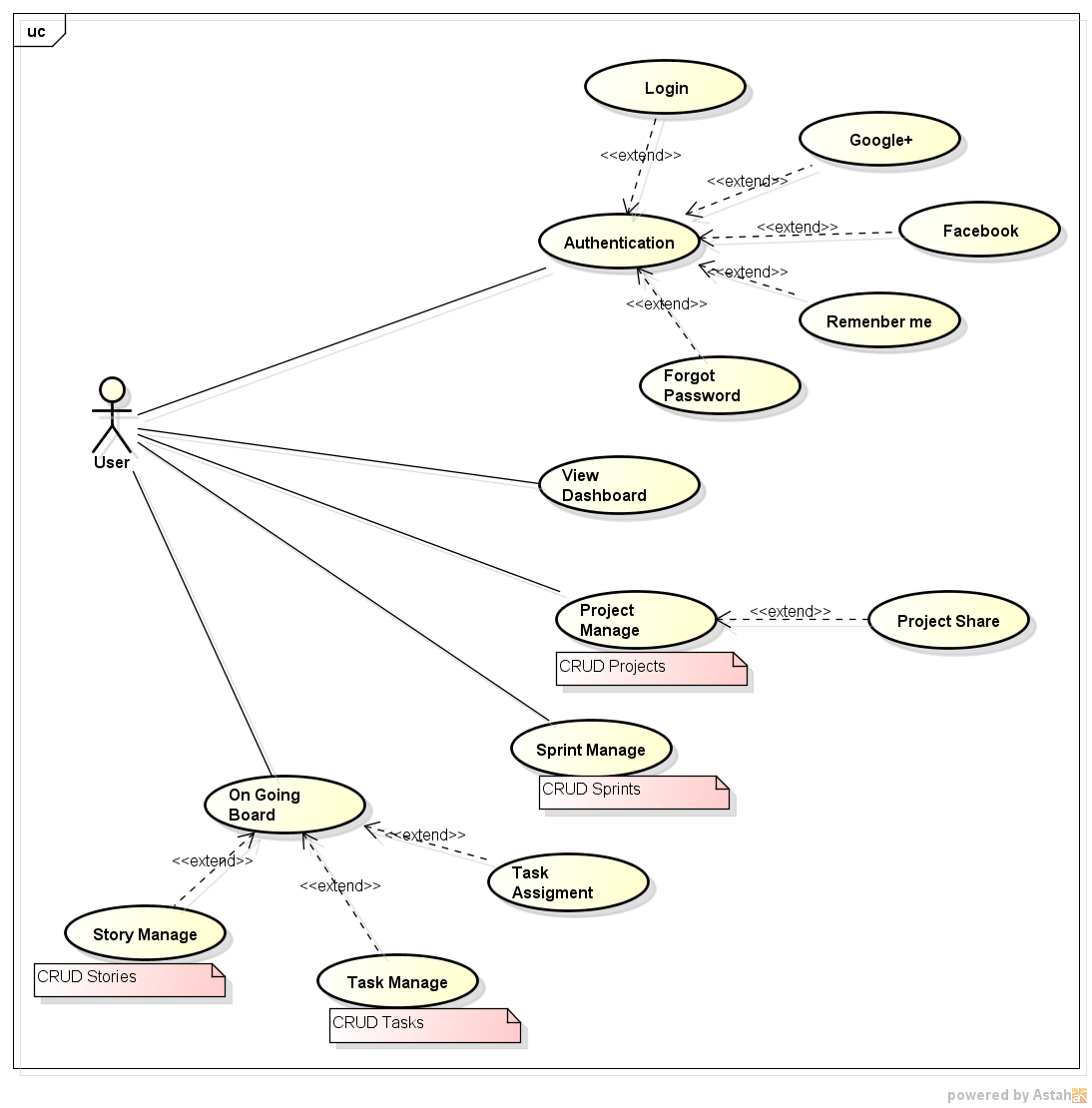
## 3.2 Non-Functional Requirements

|  |  |
| --- | --- |
| **Requirement ID** | NFR1 |
| **Title** | Performance |
| **Description** | 1. Perform of the system to read and write data. 2. Response times 3. Processing times 4. Query and Reporting times  * **0.3 second** is about the limit for having the user feel that the system is **reacting instantaneously**, meaning that no special feedback is necessary except to display the result. * **1.0 second** is about the limit for the **user's flow of thought** to stay uninterrupted, even though the user will notice the delay. No special feedback is necessary during delays of more than 0.1 but less than 1.0 second, but the user does lose the feeling of operating directly on the data. * **5 seconds** is about the limit for **keeping the user's attention.**   *( adapted by Jakob Nielsen – Response time: The 3 important limits)* |
| **Priority** | Priority 2 - feature nice. |
| **Risk** | Low (L) The system can be used without limitation, but with some workarounds. |

|  |  |
| --- | --- |
| **Requirement ID** | NFR2 |
| **Title** | Availability |
| **Description** | 1. Hours of operation - **24 hours** a day, **7 days** a week 2. Locations of operation:  web browser and internet connection |
| **Priority** | Priority 1 - mandatory. |
| **Risk** | Critical (C) - It will break the main functionality of the system. The system cannot be used if this requirement is not implemented. |

|  |  |
| --- | --- |
| **Requirement ID** | NFR3 |
| **Title** | Security |
| **Description** | 1. ~~Login requirements (authentication) - access levels~~   All users and groups of users have the same role and access (permissions), this system doesn’t need to control the login requirements. **This item must be removed.**  Password requirements  Length: Require a minimum password length for eight characters. There are no maximum password lengths.  Special characters: Require the use of both upper- and lower-case letters (case sensitivity) - inclusion of one or more numerical digits - prohibition inclusion of special characters, e.g. @, #, $ and so on.  expiry date: Require users to change passwords periodically, every 180 days.   1. Inactivity timeouts – durations and actions: the web session timeout will expire in five minutes. |
| **Priority** | Priority 1 - mandatory. |
| **Risk** | Critical (C) - It will break the main functionality of the system. The system cannot be used if this requirement is not implemented. |

## 3.3 Use Cases



|  |  |
| --- | --- |
| Use Case ID | UC 1.0 |
| Use Case Name | Authentication |
| Description | This use case **Authentication (login access)**  details the login steps of a user; it is necessary to gain access to other functionality of the system. |
| Actors | User |
| Preconditions | None |
| Normal flow | 1. The system presents fields for login, password and a list of language. 2. The user inserts her/ his e-mail (username). 3. The user inserts her/ his password. 4. The user selects a language. 5. The user clicks on Login button. 6. The use case ends successfully. |
| Alternative flows | None |
| Exceptions | 1. Invalid User or Password    1. If username or password cannot be located, the system presents a message “Incorrect username or password combination. T[ry again](https://www.seenowdo.com/application/index.xhtml)!”.    2. The system presents two options: “Forgot your password?” and “Not Registered?” |
| Posconditions | The system locates the user and verifies the access rights. |
| Notes and issues | 1. Logging or Username is usually used to enter a specific software (web page) or computer system. A password is a secret word or string of characters that is used for authentication, to prove identity or gain access to a systems. 2. The system default language is English. |

|  |  |
| --- | --- |
| Use Case ID | UC 2.0 |
| Use Case Name | Project Manage |
| Description | This UC allows to perform the actions to: create, read (retrieve), delete and update the projetcts |
| Actors | User |
| Preconditions | User can be logged |
| Normal flow | 1. Create:    1. User chooses create new project    2. User fills the project’s description    3. User defines the date to start and completion date    4. User indicates the current projetc’s status: open, close    5. User submit the project    6. The system shows a confirmation message “Projetc created successfully!” 2. Retrieve:    1. User chooses the option to view a list of projects    2. The system presents a list of projects    3. User select a project    4. The system open the project selected 3. Update:    1. User chooses to edit the project    2. User submit the project edited    3. The system shows a confirmation message “Projetc edited successfully!” 4. Delete (Remove Project):    1. User chooses to delete the project    2. The system presents a message to confirm the action    3. User confirms the action (remove project)    4. The system presents a confirmation message “Projetc removed successfully!” |
| Alternative flows | The project already exists |
| Exceptions | 1. The user aborts the use case. 2. The system doesn’t conclude the action. 3. The use case ends. |
| Posconditions | The system locates the project. |
| Notes and issues | None |

|  |  |
| --- | --- |
| Use Case ID | UC 2.1 |
| Use Case Name | Project Share |
| Description | This UC allows users to share the projetcs with other stakeholders or key people of the company. |
| Actors | User |
| Preconditions | 1. User can be logged 2. Just will be able to invite registered emails |
| Normal flow | Invite People   1. User indicates the emails of the people that will recive the project 2. User writes some message about the project or other information 3. User submit the invitation 4. User submit the message 5. System shows the dialogue box “Message sent with success!”   Clear Invitation   1. User can decide to clear the fields: invite people or message 2. User press the “clear option” 3. System delete the description’s field   Edit Invitation   1. User edits the fields: invite people and message 2. System shows the dialogue box |
| Alternative flows | 1. User invites members that not have access on the platform 2. The platform notifices that the user don’t have a login (or valid login) |
| Exceptions | 1. The user aborts the use case. 2. The system doesn’t conclude the action. 3. The use case ends. |
| Posconditions | The system to share the projetcs with other stakeholders. |
| Notes and issues | Any project can be sharing with others (or just current projects?) |

|  |  |
| --- | --- |
| Use Case ID | UC 3.0 |
| Use Case Name | Sprint Manage |
| Description | This UC allows to perform the actions to: create, read (retrieve), delete and update sprints |
| Actors | User |
| Preconditions | User can be logged |
| Normal flow | 1. Create:    1. User chooses create new sprint    2. User fills the sprint’s name    3. User defines the date to start and completion date    4. User indicates the current sprint’s status: future, closed, planned, in progress    5. User submit the sprint    6. The system presents a confirmation message “Sprint created successfully!” 2. Retrieve:    1. User chooses the option to view a list of sprints    2. The system presents a list of sprints    3. User select a sprint    4. The system open the sprint selected 3. Update:    1. User chooses to edit the sprint    2. User changes data(s)    3. User submit the sprint edited    4. The system presents a confirmation message “Sprint edited successfully!” 4. Delete (Remove Sprint):    1. User chooses to delete the sprint    2. The system presents a message to confirm the action    3. User confirms the action (remove sprint)    4. The system presents a confirmation message “Projetc removed successfully!” |
| Alternative flows | None |
| Exceptions | 1. The user aborts the use case. 2. The system doesn’t conclude the action. 3. The use case ends. |
| Posconditions | None |
| Notes and issues | A sprint belongs to a project. |

|  |  |
| --- | --- |
| Use Case ID | UC 4.0 |
| Use Case Name | On Going Board (story board) |
| Description | This UC represents the system to control the software development process. The users will be able to add new stories, management stories, and include new tasks |
| Actors | Users |
| Preconditions | User can be logged |
| Normal flow | 1. User chooses create new story board 2. User fills the story board description 3. User manages stories (UC 4.1), and manages tasks (UC4.2) 4. The system presents a confirmation message “Story Board created successfully” 5. User moves the tasks to other process steps. (not started, in progress, completed and blocked) |
| Alternative flows | None |
| Exceptions | 1. The user aborts the use case. 2. The system doesn’t conclude the action. 3. The use case ends. |
| Posconditions | None |
| Notes and issues | None |

|  |  |
| --- | --- |
| Use Case ID | UC 4.1 |
| Use Case Name | Story Manage |
| Description | This UC allows to perform the actions to: create, read (retrieve), delete and update stories. |
| Actors | User |
| Preconditions | User can be logged |
| Normal flow | 1. Create:    1. User chooses create new story    2. User fills story’s title (mandatory)    3. User fills story’s description (optional)    4. User fills story´s points (mandatory)    5. User fills notes (optional)    6. User submit the story    7. The system presents a confirmation message “Story created successfully!” 2. Retrieve:    1. User chooses the option to view a list of stories    2. The system presents a list of stories    3. User select a story    4. The system open the story selected 3. Update:    1. User chooses to edit the story    2. User changes data(s)    3. User submit the story edited    4. The system presents a confirmation message “Story edited successfully!” 4. Delete (Remove Story):    1. User chooses to delete the story    2. The system presents a message to confirm the action    3. User confirms the action (remove story)    4. The system presents a confirmation message “Story removed successfully!” |
| Alternative flows | None |
| Exceptions | 1. The user aborts the use case. 2. The system doesn’t conclude the action. 3. The use case ends. |
| Posconditions | None |
| Notes and issues | A story belongs to a sprint. |

|  |  |
| --- | --- |
| Use Case ID | UC 4.2 |
| Use Case Name | Task Manage |
| Description | This UC allows to perform the actions to: create, read (retrieve), delete and update tasks. |
| Actors | User |
| Preconditions | User can be logged  User is a member alocated in the project/sprint |
| Normal flow | 1. Create:    1. User chooses create new task    2. User fills task’s description and task’s number    3. User defines **assigned tasks**    4. User submit the task    5. The system presents a confirmation message “Task created successfully!” 2. Retrieve:    1. User chooses the option to view a list of tasks    2. The system presents a list of tasks    3. User select a task    4. The system open the task selected 3. Update:    1. User chooses to edit the task    2. User changes data(s)    3. User submit the task edited    4. The system presents a confirmation message “Task edited successfully!” 4. Delete (Remove Task):    1. User chooses to delete the task    2. The system presents a message to confirm the action    3. User confirms the action (remove task)    4. The system shows a confirmation message “Task removed successfully!” |
| Alternative flows | None |
| Exceptions | 1. The user aborts the use case. 2. The system doesn’t conclude the action. 3. The use case ends. |
| Posconditions | None |
| Notes and issues | 1. A task belongs to a story. 2. Anyone can take to assigned the task. 3. Tasks don’t need to have a user assigned when the task is added to the user story |

# **4 Supporting Information**

User-interface prototypes.

