## Constructor University Bremen

# Project Management Software Requirements

March 5, 2025

**Team:**

Bricked Up

**Author:**

Nikolay Lachezarov Tsonev

# Contents

# 1 Introduction

The goal of this Software Engineer (SWE) project is to create a final product that is a software management application. The application is intended to be web based compatible with any device aspect ratio. The web application is inspired by software such as but not limited to:Github, Microsoft Teams, Slack. The software should help other SWE teams to create their own organisations, full of members sub-teams and sub-projects, in hopes of improving productivity and shareholder satisfaction.

This document states the function and non-functional requirement of the software, the exact working criteria and measurement. Please note that the frontend and backend teams have a more in depth requirement that also include: the software tools used, software architecture, software API and internal component documentation, and justification of used software.

# 2 Functional Requirements

The final product is a general project management software that will help organisations initiate, plan, budget their projects. The following is the current list of **functional requirements**. Note that these may be subject to change.

1. Users should be able to login or register to the platform.
2. Users should be able to log out and delete their account.
3. Each account must be unique and identified by a valid email address.
4. Users should be able to change their passwords.
5. Users should be able to create: organisations, teams and projects.
6. Clear distinction in user privilege as follows from highest to lowest {Admin, Project Lead, Team member}
7. Any team member should be able to create a project and invite other org. members to said project.
8. Project manager should be able to promote other to PM and be able to remove members from the team.
9. If the PM leaves with no other PMs in a given project, the org. admin shall become the PM.

10. be able to schedule tasks
11. Each task should have the following: {budget, person(s) working on it, scope, priority}
12. Each task can be prioritised from: {High, middle, low}
13. Software should allow a team member to have more than one task at a time even if overlapping.
14. Users should be able to customise the software in respect to color, themes and font from a given selection.
15. Software is web-based and should be able to run on most browsers: {Firefox, Safari, Google Chrome, Microsoft edge}.
16. Software should remind team members who have high priority tasks via Email or other forms of communication.
17. Project encapsulation should exist where org. members that are not in the project do not have read/write access unless directly granted by the admin/PM.
18. Software will have a light and dark mode.
19. Software will allow the users too look and customise their accounts.
20. Software will allow for users to have a non-unique display name, different from the unique email address.
21. Software will allow users to find other users via search of display Name.
22. Software will prohibit access of restricted pages.
23. Software will allow the deletion of a user account and delete all user data from the Database (DB).

# 3 Non-functional Requirements

These requirements are more towards the backend and features that the users may not directly experience but are crucial for the design and development of secure, scalable, maintainable and economic software.

1. Data privacy and not allowing unwanted or malicious users to access sensitive/unauthorised data.

2. The software must be dependable and must be resilient and minimise failure and have graceful failure if failure cannot be avoided.

3. The software should be efficient, scalable and have low overhead.

4. The frontend should be designed and implemented in a structured modular manner for easier expansion and re-factoring.

5. Software minimises excessive back-end request of redundant data and instead will cache the data unless said data is updated/modified.

# 4 Productivity Measurement

The main way that productivity of the team will be measure is via Github issues. Each team has their own repository with respective issues that are automatically closed via a pull request. The benefit of this method is that instead of measuring lines of code, we measure how much functionality each team member has contributed. This method also encourages for the developers to write efficient and DRY code as there is no punishment for having fewer lines of code. The whole Github organisation can be found by clicking this link.

Another measurement is meeting participation. Each week the whole team has a stand-up meeting with the fronted and backend having individual weekly meetings. If a team member misses 3 meetings unexcused, or is late by more than 20 minutes, the team member will lose 5% of their grade as punishment. This is done to incentive each member to participate in meetings and keep communication as optimal as possible.