**Your Name** <your-ju-id@student.ju.se>,
**Your Name** <your-ju-id@student.ju.se>

A Project Work in *Client-Server Communication*

Jönköping University 2022

Your platform title here

*In this template, all italic text should be removed and replaced with your own text (which should not be italic); the italic text is just a placeholder letting you know what to write there.*

*On the cover page, change to your own platform title, your own names and your own JU email address.*

*You have a lot of freedom when it comes to writing this report. You do not have to use any part of this template, but the report you write should in the end somehow (in a good way) provide the same information as indicated in this template. Most students trying to do it in their own way usually fail, so if you try that, be sure to know what you are doing!*

*To learn common mistakes students often do (so you can avoid doing them), read:*

[*https://peppel-g.github.io/course-material/lectures/report-writing/*](https://peppel-g.github.io/course-material/lectures/report-writing/)

*This page should be removed.*

Table of Contents

[Introduction 3](#_Toc112505020)

[Graphical User Interface 4](#_Toc112505021)

[Architecture 5](#_Toc112505022)

[Database 6](#_Toc112505023)

[REST API 7](#_Toc112505024)

[Backend Application 8](#_Toc112505025)

[Frontend Application 9](#_Toc112505026)

# Introduction

*Introduce your platform. Write text that* ***indirectly*** *answers questions like:*

* *Why does the platform exist/what is the problem it solves?*
* *How does the platform solve the problem?*
* *Why is your platform solution better than existing solutions (if any exists)?*
* *How will end users use the platform?*
* *...*

*After having read this chapter, those that have never heard of the platform before should a have a good understanding of what It is, and what users are able to do on the platform. If they would like to learn how it has been implemented, they just need to continue reading the rest of the report.*

*If possible, use pictures/figures (preferably a UML use-case diagram).*

*This chapter can be seen as a pitch text: imagine the reader is an investor, and you should convince the reader that your platform solves an existing problem in an excellent way and that it is worth investing money in it.*

# Graphical User Interface

*Show pictures of what you intend the graphical user interface to look like in the end, and how the user can use it to achieve her goals on the platform. Details (such as colors, specific font size, beautiful background, etc.) are not that important now (wireframes are good enough), but be sure to make the reader understand how the user can navigate between the different pages in the SPA, and how to achieve her goals (such as how to click to first sign in and then create a new blogpost (in case you implement a blog hosting platform)).*

*When you have implemented the platform, you can take screenshots of the GUI and show in this chapter too.*

# Architecture

*Give an overview of the platform. Which components does it consist of (backend application, frontend applications, frontend devices, databases, etc.)? Visualize this using a figure and show how the different components make use of/communicate with each other.*

*For each component, provide a sub-chapter with a small amount information about what the component is responsible/used for. Unless there is very little to say about the component, also have a main chapter about the component, and explain in greater detail how that component work.*

*After having read this chapter, the reader should have a broad (but shallow) understanding of the entire platform.*

# Database

*Describe the structure of your database, and your resources in detail. What attributes do they consist of? Showing an ER diagram might be a good way to visualize the resources? Try to use a figure of some kind.*

*After having read this chapter, the reader should know how the data stored on the platform is structured and stored. If the reader is a new programmer that should start working on the platform, she should now know what she needs to know if she wants to change the resources or add more type of resources (e.g. know how to add a new table to the database with a relation to an existing table in the database).*

# REST API

*This chapter should contain the specification for the REST API on your platform. It should contain enough information so a new programmer can start using the REST API without reading through the code on the backend application. Be sure to mention all the details one need to know to use the REST API.*

# Backend Application

*This chapter should in detail describe how your backend application has been implemented. Which programming language have you used? Which framework have you used? How has the source code been structured? In multiple files? Which libraries/packages are you using? Etc.*

# Frontend Application

*This chapter should describe how your frontend application has been implemented in detail. Mention the structure of your code, such as which Vue components you have in the "views" folder and which Vue components you have in the "components" folder, how you structured the code communicating with the backend, which your global variables are, which libraries you use, etc.*