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

A Project Work in *Web Development – Advanced Concepts*

Jönköping University 2023

Your 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 name, 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.*

*For more tips on how to write a good report, read:*

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

*This page should of course be removed.*

Table of Contents

[Introduction 3](#_Toc124676069)

[Graphical User Interface 4](#_Toc124676070)

[Architecture 5](#_Toc124676071)

[REST API 6](#_Toc124676072)

[Frontend 7](#_Toc124676073)

[Database 8](#_Toc124676074)

[Backend 9](#_Toc124676075)

[Worklog 10](#_Toc124676076)

[Alice 10](#_Toc124676077)

[Bob 10](#_Toc124676078)

# Introduction

*Introduce your platform here. Write text that* ***indirectly*** *(if you do not know what “indirectly” means, you need to look it up) answer questions like:*

* *Why was the platform created?*
* *What can the platform be used for?*
* *Why would anyone use the platform?*
* *Who will be using the platform?*
* *How will users use the platform?*
* *...*

*Add a UML use-case diagrams that visualizes how end users will use the platform and what they will use it for.*

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

# Graphical User Interface

*Explain what the graphical user interface (GUI) will look like, so the reader will know how to perform the use-cases described in the previous chapter. The best way to describe the GUI is to show what it will look like. In the beginning of the project, this can be simple sketches of the layout and the different pages. When you are done with the project, you can instead use screenshots of the final GUI.*

# Architecture

*Give an overview of the components the platform consists of (frontend, database, web browser, backend, end-users, etc.). Visualize this in a figure and show how the different components make use of each other. Do not show implementation details of the different components (you have other subchapters for that).*

*Are you using Docker? Then it also makes sense to describe which parts of the platform that run in which containers in this chapter. Maybe visualizing this with a figure is a good idea?* ***Hint: Yes, it is****.*

*After having read this chapter, the reader should have a broad (but shallow) understanding of the platform’s internal components and structure.*

# REST API

*Specify how the REST API on your platform works. Which different operations does it support? Which URI should be used? Which headers should be used? Which status codes can be sent back (for each type of request)? What information should the bodies contain? Etc.*

*Anyone who needs to implement the REST API or use the REST API should find enough information about it to do so by reading this chapter.*

*Note: This chapter is just for the specification of the REST API. How the frontend uses the REST API should be described in the Frontend chapter, and how the backend has implemented the REST API should be described in the Backend chapter.*

# Frontend

*Explain to the reader how the frontend application has been implemented, and how it communicates with the backend, etc. Use many figures.*

*After having read this chapter, the reader should have a good understanding of how the frontend application has been implemented.*

# Database

*Describe the resources on the platform in detail. What attributes do they consist of? How are they related? Maybe an ER diagram is suitable to have?* ***Hint: Yes, it is.***

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

# Backend

*Describe implementation details of the web application. Which language have you used? Which framework have you used? Which libraries/packages have you used, and for what purpose?*

*Has all code been written in one file? Or have you somehow structured it in multiple files? Or layers? Are you using some design patterns (e.g. MVC)? Are you using middlewares? Etc... Maybe visualizing these things in figures is a good idea?* ***Hint: Yes, it is.***

*You do not need to show any code to describe the implementation (class diagrams, object diagrams, sequence diagrams, etc. are enough), but if you feel that showing code improves the quality of the report, then you should of course do that.*

*After having read this chapter, the reader should have a very good understanding of how the web application has been implemented and how the code is structured. If the reader is a programmer who should start working on the web application, she should now know where to start when she should implement new features to the web application.*

# Worklog

*Although it doesn't make sense to make the worklog part of this report (which should really only describe the implementation of the project), add it here anyway.*

*If you have used Git properly (descriptive commit messages, no huge commit that implements 3.5 features, etc.), showing the list of commits per member is enough, for example:*

## Alice

commit 89cc24d3c59d4dc5942d7e4198a9235f85575c72

Author: Alice <al@ice.com>

Date: Thu Nov 25 10:40:18 2021 +0100

Setup Docker for the backend with the Express framework.

commit 67f9e90bc3abfbfda57fded1bf30f005406704a3

Author: Alice <al@ice.com>

Date: Tue Nov 23 15:58:05 2021 +0100

Added the about page with explanation of what the platform can be used for.

…

## Bob

commit 9f4434e4a538231be1a27cf573ddabb53e1a645f

Author: Bob <bob@by.com>

Date: Thu Nov 25 09:38:20 2021 +0100

Implemented the layout in the frontend.

…

*If you haven't used Git, or haven't used Git properly, show similar info in a table, for example:*

|  |  |  |
| --- | --- | --- |
| Date | Alice | Bob |
| 2021-01-17 | Formed group | |
| 2021-01-18 | Had meeting, decided to build video games platform | |
| 2021-01-19 | Researched suitable CSS framework to use, designed the layout of the platform | Selected brand colors and created a logo for the platform |
| 2021-01-20 |  |  |
| 2021-01-21 | Had meeting, presented work since last meeting | |
| 2021-01-22 |  |  |
| 2021-01-23 |  |  |
|  |  |  |
| 2021-01-24 | Setup the basic structure for the platform with docker |  |
| 2021-01-25 |  |  |
| 2021-01-26 |  |  |
| 2021-01-27 |  |  |
| … |  |  |