# CVWO AY2020/21 Mid-Assignment Submission

Benjamin Tan

December 30, 2020

## 1 Introduction

Hi, I'm Benjamin Tan, a Year 1 Computer Science student. I have had prior programming experience, particularly with JavaScript and React, but have not had experience in creating a full-stack project from scratch. I hope that this assignment during the AY2020/21 winter break would allow me increase my experience in creating complex projects, and bring me up to speed on the various tools used to develop websites commonly used in the CVWO technology stack.

My goals for this assignment would be to learn and gain familiarity with the various technologies which I have not used before (Ruby, Rails, SQL / databases / PostgreSQL) and create a project with good code structure and techniques that is easily maintained.

### 1.1 Primary goals

- clear and thorough documentation, including well-written Git commit messages
- automated deploys of the frontend and backend of the code into a website
- linting of the codebase based on best practices
- if possible, writing unit tests for the backend

## 2 Product

**TodoMeister** will be a simple and easy-to-use todo/task-list manager, with filtering/sorting capabilities.

Users will be able to sign up for a new account, then login to the website to start managing their tasks.

Tasks will contain a brief title and an optional extended description. If desired, users can also set a due date/deadline for their tasks. Tasks can also contain subtasks, to allow users to divide their large tasks into smaller, more attainable tasks. Tasks can also be tagged with various categories for easier management of tasks.

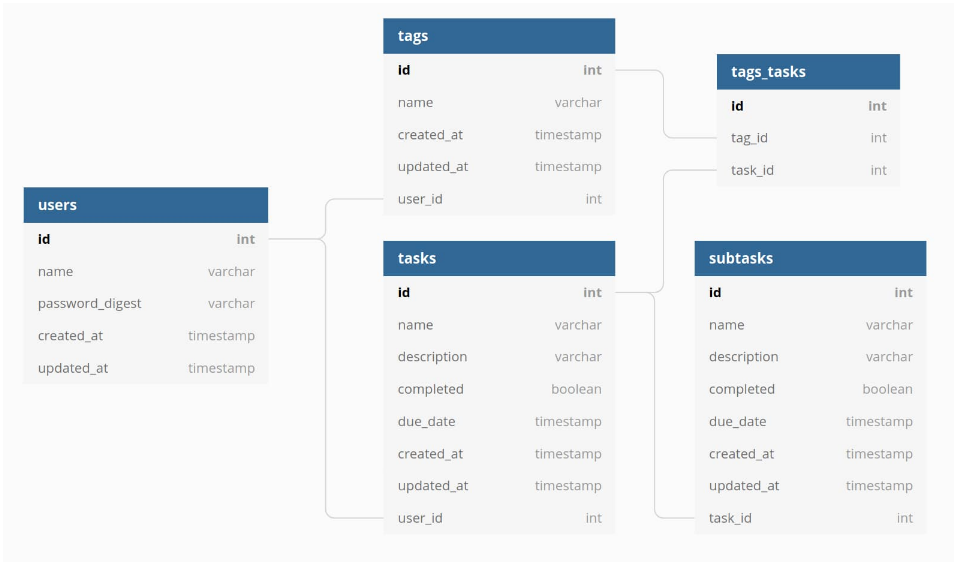
**TodoMeister** will also include filtering and sorting capabilities. All tasks will be shown by default, but they can be filtered based on their completion status, tags and due date. Tasks can also be sorted based on their creation date, last edit date and due date.

## 3 App structure

- The designed website/app will have separate frontend and backend codebases.
- The frontend will be a React application, written in TypeScript, which makes use of React Router to handle routing, and react-query to handle interfacing with the backend.
- The backend will be a Rails application, which provides a RESTful JSON API.
- The frontend and backend codebases will be linted according to best practices using ESLint and Rubocop respectively.
- The backend will be backed by a PostgreSQL database. PostgreSQL was chosen since it appears to be one of the more widely used relational databases, and is offered by Heroku, which could lead to a simpler deploy process.
- Authentication between the frontend and backend will likely be performed using JSON Web Tokens, but will remain a lower priority until after the main tasks functionality has been implemented.
- Deploys will likely be performed with the frontend being hosted on Netlify and the backend being hosted on Heroku.

- Styling of the app will either be performed using styled-components, or using a CSS library like Tailwind or Bootstrap. *Note:* Styling of the app will remain a low priority: my primary focus will be making the rest of the application functional, and styling will probably be the last task once the main features are completed.

## 4 Database schema



## 5 Tasks

| Task | Priority | Status |
|---|---|---|
| CRUD of tasks and subtasks | High | In progress |
| Filtering of tasks by completion status and tags | High | Completed |
| Sorting by creation date, last edit date and due date | High | Completed |
| Users and authentication | Medium | Researching |
| Automated deploys of the frontend and backend | Medium | Completed |
| Editing of tags | Low | Not started |
| Styling | Low | Not started |