# CVWO Mid-Assignment Write-up

As someone from the C++ world, this is the first time I am formally learning web development.

I have dedicated the entire December to learning web development using Ruby on Rails with the help of the Rails Guide. I finished all the readings on 27 December and will catch up on other areas of web development not covered in the Rails Guide (e.g. styling, AJAX) in January. Hopefully, I can finish them and incorporate them into my final application.

## Basic use cases

### *Task components*

A task consists of a title, a short description, and a deadline. Each task can also be associated with any number of tags.

### *Listing Tasks*

This will be the homepage of the application. Tasks will be listed on this page and can be ordered by title, time created, or deadline, as chosen by the user. Only uncompleted tasks will be visible by default; archived tasks can be viewed by clicking on a button. Task descriptions will be truncated if they are too long. When a task is near due, a prominent alert message will be displayed to remind the user.

### *Adding Tasks*

New tasks can be created as long as they are provided with a title. All the other fields are optional. This function will be available as a button on the homepage.

### *Viewing Tasks*

All information about a task, including its full description, will be available on this page. Additional buttons, including editing and destroying the task, will also be available. This function will be available as a button on the homepage.

### *Editing Tasks*

Tasks can be edited after they have been created. All attributes can be modified, including adding or removing tags. New tags can be created and added at this step. This function is available as a button both on the homepage and when viewing a specific task.

### *Deleting Tasks*

Tasks can be deleted if the user wishes to do so; otherwise, tasks marked complete will be archived. This function is available when viewing a specific task.

### *Searching for Tasks*

Tasks can be searched for with their names or tags. This function will be available on the homepage.

# Execution plan

### *Overview*

MVC architecture and RESTful design pattern will be strictly followed in this application.

Both the backend and frontend will be done with Rails.

The end product will be hosted on Heroku.

### *More specifically...*

Tasks and tags will be treated as RESTful resources, with a many-to-many relationship. As Heroku requires the application to use PostgreSQL, it will be the choice of database instead of the default SQLite.

Styling will be done with Bootstrap, though I still have some catch-up to do.

A key will be stored in a Cookie to uniquely identify each client so that each client will have its own list of tasks.

As I learn more about AJAX, I would also incorporate it into the app to enhance the user experience.

# Suggestions

For this assignment, we are required to learn Rails by reading the Rails Guide. Though very comprehensive and informational, I find it extremely dry and disorganised, which might make it a good fit for a reference manual (in fact I realised that some of its content are copy-pasted word for word from the Rails API documentation, violating its own principle of DRY), but definitely not for beginners to learn the framework and web development. New concepts seem to emerge at the convenience of the author, without any explanation or regard to the reader's ease of understanding.

As an alternative, I personally find the *Ruby on Rails Tutorial (Learn Web Development with Rails)* by Michael Hartl (free online version at https://www.railstutorial.org) much more effective at explaining how different components of the Rails framework fit together and making the reader truly appreciate 'the Rails way'. It also comes with practices for each section. More importantly, instead of focusing on Rails only, it integrates other areas of web development (styling, AJAX, etc.) into the book, saving the effort to learn them from other resources.

If possible, I would suggest that CVWO require future applicants to read the Ruby on Rails Tutorial instead of the Rails Guide.