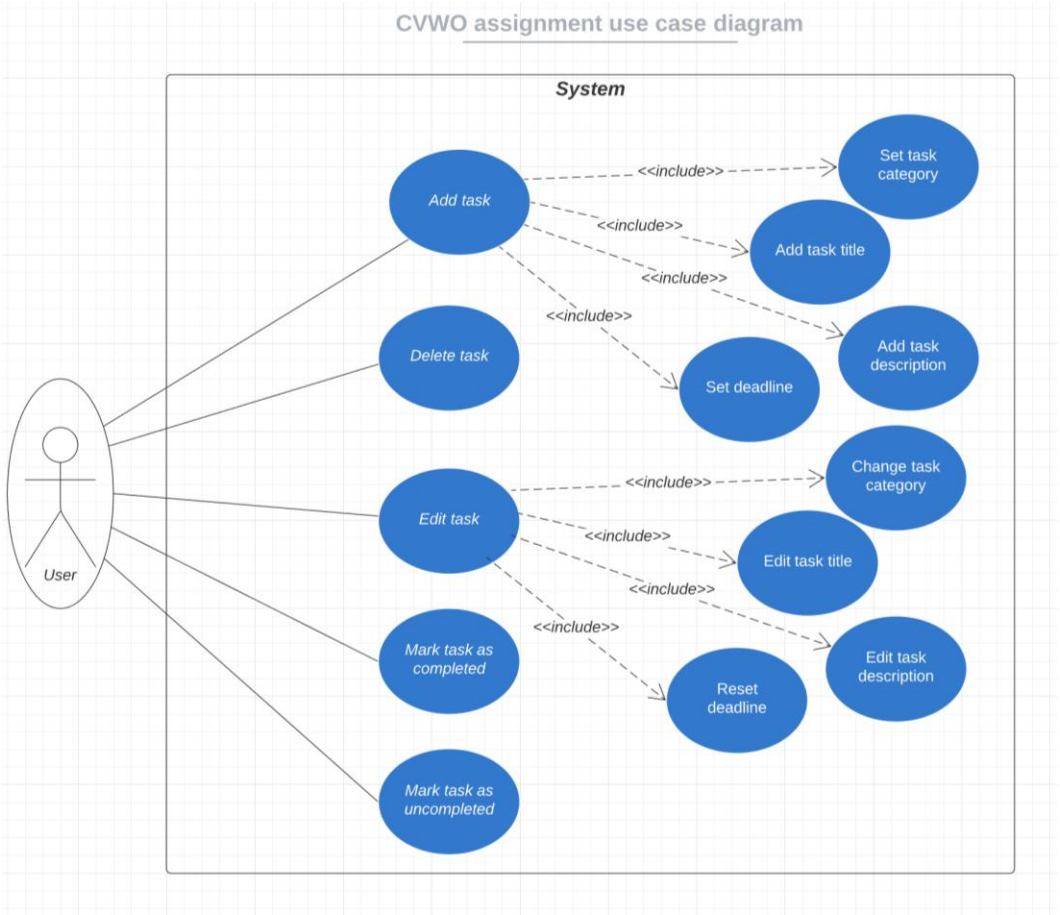

The diagram shown above is a simple use cases diagram for the todo list manager with a tagging system that performs basic CRUD operations. Before we begin, we need to make sure that the essential methods (such as those to create, delete, update, and view tasks) are defined properly in the task controller and resourceful routing has been done. (resources :tasks)

The root route should direct us to a webpage showing all our current tasks (such as tasks#index), this webpage should also allow us to add in new tasks, which can be done by adding in a link_to helper under the view index.html.erb, linking us to a form to create our new tasks. After specifying the category, title, description, and deadline of the new task, the tasks will then be saved and can be viewed when we return to our root route.

The process of updating a task is rather similar to that of creating one. It is also done with the link_to helper that links us to a form, which then creates a new task. But in this case, instead of saving this task as a new one, it overwrites the original task, thus the link_to helper should be added under the view of a specific task (show.html.erb) instead of the homepage listing down all the tasks.

To delete a task, it only required a controller action since our resourceful routing already provides the route for the action. Similar to updating a task, this action is also done on a specific task, thus the link_to helper should be added under the view of the task that we want to delete. (show.html.erb) By clicking on the link, the task shown on the current page will then be deleted.

Above is a simple description of how we can perform our CRUD operations with the task manager. However, I have yet to add in the tagging systems for the application. My current idea is to add in another model for categories and save a new task twice when its created, once under tasks and once under its specific category, but I have yet to try it out in my code. Other than that, I also found out that my current application can only be used for one single user. My next step is to solve this by allowing multiple users to log in with their own usernames and passwords.

Other things that have yet to be done includes the front end code (written in react) and deploying the app on Heroku.