CPSC 436I - Assignment 4

DUE DATE: **Monday, July 6 (10 PM Vancouver time)**

- You will most likely be using the same repo as your assignment 3, so submit by creating a branch called “assignment_4”, and do not push any more code after the due date
- **Make sure your repo is still private**

Postings Website… Continued!

For the fourth assignment, you’ll be adding more functionality to your postings website, now with Node and Express! It will build on the previous assignment, connecting your DB to your front end, so start from your completed assignment 3, and begin to incorporate the requirements below.

We’re expecting the following:

1) You MUST use Node AND Express (Express App Generator is acceptable), and start your assignment from your completed assignment 3. Remember, you will be asked to explain sections of your code.

2) The initial data (initial set of messages) should be loaded using a GET request to your server, and should get the messages from your Mongo DB.

3) You can add a new message using a PUT or POST request, which will store it in your Mongo DB.

4) One additional server request of your choice. Examples could be:
    - ○ Editing an existing message
    - ○ Clearing the list of messages
    - ○ Getting details for a message

5) You will need to handle the calls asynchronously (because they’re asynchronous calls). You can either do this in a component or in an action. If you want to use it in an action, you’ll need to use redux-thunk, which is the standard way to do it. It is a bit more advanced, however, so if you don’t feel confident, just make the call from the component.

6) Something cool and extra! This is wide open for you to explore, and try to push your knowledge and boundaries.
    For example:
    - Some kind of new component
    - A spinner while making a request (until it finishes)

- Some more advanced styling (animations or transition)
- As listed above, editing a message

As described in the individual assignment rubric, your code will need to meet these requirements and be functional, up to perhaps a few minor glitches in tricky cases. Note that functionality includes both user-visible and console-visible issues.

It’s up to you! We’re hoping that you’ll use the above requirements as a guide, but that you’ll let your imagination take over, and build something unique and interesting!

You should be ready to demo this to a TA during your second week lab, and should be ready to answer questions about it, as well as explaining what you’ve done.

HAVE FUN!!!