Premise:

- The internet is a wonderful tool that should be accessible to everyone regardless of their capabilities (ex visually impared)
- There are potentially 4 million people potentially using screen readers to surf the net and many websites are still terribly inaccessible
- Ontario and the rest of Canada are implementing measures to make the web more accessible:
    - **Beginning January 1, 2021:** all public websites and web content posted after January 1, 2012 must meet WCAG 2.0 Level AA other than criteria 1.2.4 (live captions) and 1.2.5 (pre-recorded audio descriptions)

Goal:

- To create a web application that will assess a website's accessibility level based on the guidelines of Web Content Accessibility Guidelines

How is this more complex than the mid-term project?

- Assessing a dom structure and elements of existing website
- Translating guidelines into a concrete set of examples/tests we can run

What tech stack do you plan on using?

- Node.js, React, Webpack, Storybook, Jest, Cypress, Amazon Web Services (continuous integration?)

What technology or part of the project will you have to spend the most time researching and/or testing prior to coding?

- What the guidelines for level AA of WCAG are
    - How can we translate this into concrete tests
- How do screen readers work and what web structure/elements make it easier for people using screen readers to navigate pages
- How to deploy the tests which will be stored in a server
- Retrieving a web site and analyzing the HTML, CSS, etc

What are your top 2 user stories?

- I operate a small/medium size website (less than 100 pages) and I would like to know if where I can improve on accessibility so I can meet the upcoming guidelines
- I am building a new website and would like a tool to check if my website is up to accesibility standards

Premise:

- Scrum teams use a technique called relative sizing to estimate their work. The most popular technique used for sizing is planning poker. However, there is another very useful method of sizing that is quicker and often more effective than planning poker, especially for teams that have many stories to size at any given time – affinity sizing. This process works extremely well when everyone is in the same room. It's much more difficult for geographically distributed teams. There are whiteboard apps that can be used to run a session like this, but as they are not designed for this purpose they require a lot of configuration and make the experience much less smooth than it could be. A tool well suited to visually allow a distributed team to perform affinity sizing doesn't exist.

Goal:

- Develop a web-based application to provide unbounded whiteboard facilities specially suited to the needs of affinity sizing.

How is this more complex than the mid-term project?

- Advanced React
- WebSockets
- Full TDD

What tech stack do you plan on using?

- Node.js, React, Webpack, Storybook, Jest, Cypress, Amazon Web Services (continuous integration?)

What technology or part of the project will you have to spend the most time researching and/or testing prior to coding?

- AWS
- Drag and drop of elements with automatic shuffling/rearrangement in React
- Possibly: Integrating with Jira via their API

What are your top 2 user stories?

- Need something
- Need something else