React Coding Standard Style Guidelines

- Indenting:
    - Use 4 spaces within components
    - Use 2 spaces within functions
    - Previous collaborators have used tab spaces, but one should avoid them since different operating systems may have different values for tab spaces.
- Newlines:
    - Use newlines to separate components/functions for easier readability
- Variable naming:
    - use camelCase for local names within a component.
    - Use underscore if you're assigning a prop to a variable or vice versa; this avoids name reuse unreadability.
- Filename naming:
    - Use uppercase, and name it after what the file's purpose is. Be specific, so instead of EditModal, use EditProfileModal or EditDeviceModal, depending on application.
    - Folderize your components. Make sure everything in a directory are related, so it's easier to modularize.
- Component naming:
    - For a functional stateless component, you can use the ES6 standard: **= () =>**
    - Use PascalCase
- Function naming
    - Use camelCase.
- DRY
    - Split your code into multiple smaller functions, so you can reuse certain functions. Use React Contexts to import these functions.
    - Create multiple files instead of writing a big file.
        - Separate components into their own file into the folder *components*.
- React Props:
    - If you have 3 or more properties, put them on their own line both in the instance and in the render function. This way we avoid stuff like (<{[({})]}>)
- Dynamic renders:
    - Map out the arrays/list

Git Standard Style Guidelines

Since this product has many modular user stories, follow these Git guidelines for better organization and less merge conflicts. (Also see: https://www.dropbox.com/scl/fi/li3v3znv2nynsl3uo2urb/115-Git-instructions.paper?dl=0&rlkey=yj9ptbey9a9pugh64htv49e6z)

Setting up:

When creating a new feature branch :

```
git checkout 115
git checkout -b
```

Push branch to origin so everyone can see it:

```
git push -u origin -f
```

Set upstream:

```
git branch --set-upstream-to=115
```

Good to go!

Working on own branch/user story:

Pushing changes to a branch:

```
git commit origin
```

Merging changes back into team 115 branch

```
git push origin <branch with new changes>:115
git checkout 115 git pull origin
```

Resolve merge conflicts

Commit & push changes to 115 branch when done

Error Corrections/Rewinding:

Remove files from git add:

```
git reset
```

Restore a file back to the version of it on another branch:

```
git checkout origin/[branch_to_copy] [filename]
```