My Process

1. The first step I did when working on this project is running it using “npm start”, reading the package.json file to see where my scripts come from.
2. After the project loaded to the web, I opened up the console to see if there any imminent problems.
3. The third step was playing around with the project and documenting everything that wasn’t working properly (even though it was written in the assignment)
4. In this step I took some time to read the code, see if there are any parts that I am not familiar with. See if the architecture of the project seems right.
5. Noting to myself how a task object should be written
6. Reading online if there is a better way to implement the Context part and changing it accordingly (while commenting out the old parts)
7. Starting solving the problems one after another, going from the most important ones (making the tasks appear) to the most complicated one (editing an existing task)
8. After fixing the way the context was implemented in the code, allowing children division access to the state of the app (todos, setTodos) , it was pretty fast fixing some of the issues.
9. Adding the edit functionality took extra effort, considering what is the best way to implement this. I considered two options:
    a. On “E” click -> instead of the label, a text box would appear with the given label inside, the “E” button would turn into “V” button (to

complete the edit). This approach would require adding additional code to the checkbox component, adding extra logic between todo-list component and the checkbox. It seemed needlessly messy.

b. Giving the form component access to a given task. When a client wants to edit a given task, it is erased from the to-do list, the label is then updated in the state as task. The task then appears inside the form component. This approach required adding additional access to a different variable in state. **I went with this approach.**

10. The rest of the project required some kind of method to allow the data inside the check-list to be persistent. Since we are not using any database, I thought about using the localStorage method. I wasn't sure how much effort should be given to this part.

11. The last part was uploading the project to Heroku, which took a while because I was missing the build script required.

Significant functions

- **TodosProvider** – creating the initial state by either reading it from the localStorage or a default empty list to contain all future tasks. It also provides access to a current task being written.
- **Todo-list**:
    - **toggleCheck** : changing task checked state (done or not done)
    - **onDelete**: using filter function, erase any task with the same id given

    - **onEdit**: erasing the current task and placing the label inside the form's text box
- **Todo results**: checking each task if checked===true and adding to count. I didn't like the implementation of this function since everytime there is an update with the task list, it needs to count the entire list again**. I thought about adding additional logic but I thought maybe I'll be investing time on something is not the point of this home-assignment.**
- **Todo-form:**
    - **handleAddToDo:** using uuid library each task is given a unique id. A task object is created, inside the label is the current task variable from state. Once the function is completed task var is set to '' .