React

The goal of this exercise is to familiarize you with React – a JavaScript library for building user interfaces. In this exercise, however, we'll do things a little bit differently. First, I'll direct you to an online course covering the subject and then I'll give you the assignment. Remember, you are not supposed to send me the solution to the online course – it is just for you to learn the technology. You don't even have to do the course if you'll learn all of this from somewhere else. It is only one of many materials available for you to learn from and it is the one I'm personally the biggest fan of. You should only send me your solution to the assignment and that's all I'll judge. Nevertheless, I highly, highly recommend you go through it because it is worth it!

The course

So, the course I'm recommending is this one. It is (in my humble opinion) a perfect online course on React AND it is free AND it is built around an awesome in-browser tool which allows you to play around with the code being discussed live during the presentation. The course spans over a lengthy time of 6 hours, however, you'll quickly learn that you can skip through much of the content once you start getting the hang of it + for the purposes of the assignment, you can pretty much stop after the 49th lesson (although I highly recommend going through the whole course since it's awesome☺).

The roadmap

Here's a knowledge roadmap for you. Make sure you understand the concepts described here. If you'll go through the course recommended above you're pretty much covered.

1. Make sure you know how to write a simple app from scratch (index.html → index.js → JSX → import React).
2. Add a functional component.
3. Extract the component out to a separate file.
   - import React
   - export default MyComponent
   - import MyComponent
   - separate Components folder
4. Nesting components (calling a component inside another component)
5. Styling your components
   - className
   - You can't style components, just predefined html elements.
6. JSX vs JavaScript
   - going from JSX back to regular JS: {expression}
7. Writing inline styles

- style attribute accepts an object (watch out for double parentheses: {{rules}})
- watch out for camelCase CSS properties: background-color →backgroundColor
- 10px → “20px”
- When are they useful?

8. Passing parameters to functional components
   - *props*
   - analogy to regular html: <img src=”...” → <MyComponent someParam=”…”
9. Lists
   - use JS *map* function to create multiple components out of an array of objects/values
   - **!!! key** – why does every component on a list need it, why it has to be unique and constant, and why is it so important (virtual DOM) **!!!**
10. Class components
    - *render* function
    - *this.props*
11. State management
    - *props* are immutable
    - State is the only thing that can be changed in a given component – make sure you understand why is that.
    - Constructor*: super()*
    - *this.state = {}*
12. Events
    - Analogy to regular JS: onclick=”f()” → onClick={f()}
    - **!!!** Watch out for the *this* reference! (bind or arrow functions) **!!!**
13. Changing state
    - One does not simply change the state of a component...
    - *setState()*
    - 2 versionsof *setState()* – when to use which one
14. Component lifecycle
    - *render*
    - *componentDidMount*
    - *componentWillReceiveProps* [deprecated]
    - *componentWillUnmount*
    - *componentDidUpdate*
15. Conditional rendering
    - Rendering components depending on some conditions.
16. Forms
    - Two sources of information: 1) form 2) state – we want the state to be the only source of information
    - Solution: controlled components (https://reactjs.org/docs/forms.html)
17. Business (smart) components vs presentational (dumb) components

The assignment

After you get the hang of the library, your assignment is to write an app which will allow you to store, rank and search through your collection of… whatever it is that you're collecting. The app should display a **sortable** and **searchable** list of all of your items with the option of **adding** and **deleting** items. Each item should consist of at least a **name**, **description**, **image**, and **rating**. The user should be able to change the rating of added items (making the items fully editable is optional). The app should have at least a basic styling which you can do by hand or rely on an existing library, e.g., https://react-bootstrap.github.io/.

Remember! You are only supposed to write the front-end of the application, so you don't have to store the changes you make to the list anywhere permanently! The initial list of items should be loaded from a **local JSON file**, but after that, you don't need to update the file with the changes you make through the app!

Good luck!