How to create a React app lab

Let's create the basis for our React movie reservation system. We'll start off by creating the bare-bones React app and then add some components to it.

Creating a starter site

1. Open a command window and create a new React app called "client". You can either use npx to run create-react-app or use npm to install it, then run create-react-app. Either way, create a new React project called "client". Remember, this command may take several minutes to run mostly because of the npm installation of the supporting libraries.
2. Notice that there is now a new directory called client. cd to it. Look around at the files created.
3. Take a look at the package.json file created. Browse through the dependencies. Notice how few there are.
4. In that package.json file, add this line and make sure your JSON is valid:

```
"proxy": "http://localhost:3007/",
```

5. This will tell our app that there is API data available and to fetch it there. We'll use that a ton later.

Adding static assets to your site

Do you see the folder called public? This is where static assets go, things like fonts, movies, and images.

6. Look in the starters folder and you'll see a sub-folder called img. Move that folder (and all of its sub-folders) under public/img.
7. Take a look at what is in there. You should see some movie posters. Whenever Dinner-and-a-Movie gets a new movie poster, we can put it in there so it'll be available to our web application. We'll use these posters in future lab exercise.

Using the development server

8. In a command window, ensure you're in the *client* directory you just created and do this:

```
npm start
```

9. Look at the message. Make sure it says that it is running a server at a particular port.
10. Point your browser to localhost at that port. Make sure your page comes up. You should see a simple React boilerplate page in the browser.

Hey look, it refreshes automatically!

11. Keeping your browser open, edit the App.js file in your favorite IDE.
12. Do you see the return statement? Replace everything being returned from the App function with this:

```
<h1>Hello world</h1>
```

13. Hit save and watch the browser window. You should see your application reload without refreshing. This is the file watcher in action.

Now let's put in something that will be useful.

Making a navigation menu

14. In the starters/html folder, find a file called App_menu.html. Copy all of its contents and paste it over the top of your <h1>Hello world</h1> message.

15. Hit save and watch the browser window.
16. You should see a bunch of links. Do you?

These don't really go anywhere yet and they don't look too good but be patient and we'll fix all of that in coming labs. Congratulations! You're finished.