**Development Instructions**

1. Update your repo with the new starter code (see above).
2. Change directories into your challenge7 folder: cd challenge7
3. Install the project dependencies by running: npm install
4. Obtain an API key from Open Weather Map (see above).
5. In your challenge7 folder, open the package.json file. In order to be able to properly publish to Github Pages, you must update the homepage url to use your challenges- repo name instead of challenges-jasonnutter. So if your challenges repo is named challenges-uwnetid, the homepage property should be updated to "homepage": "https://info343a-au17.github.io/challenges-uwnetid/challenge7"
6. To run your app locally, cd challenges7 and run npm start
7. To run lint, run npm run lint
8. To publish your app to Github Pages, run npm run deploy (you need to do this every time you want to update your app on Github Pages). You do not need to manually merge your challenge7 branch into gh-pages, as npm run deploy will do that for you.

**Project files and folders**

* build: Folder contains files generated from running npm run build. You should not edit any files in this folder.
* node\_modules: Folder contains project dependencies downloaded from running npm install. You should not edit any files in this folder.
* public: Folder contains static files that you want to use [outside the module system](https://github.com/facebookincubator/create-react-app/blob/master/packages/react-scripts/template/README.md#using-the-public-folder).
* src: Folder contains JavaScript modules, React Components, and CSS files for your app.
* .eslintc.json: Configuration file for eslint. You should not edit this file.
* .npmrc: Configuration file for npm. You should not edit this file.
* package-lock.json: Lock file for npm. You should not manually edit this file (npm will automatically edit this file).
* package.json: Manifest file for npm. Contains list of dependencies and scripts to use with npm run. If you add npm modules to your project using npm install, npm will automatically edit this file.

**Requirements**

For this assignment, the overall look and feel of your application is up to you, however, the functionality must satisfy the following requirements:

**A. Look / Feel**

1. Your application must work correctly on both small and large screens (from 320px to 1200px, at least).
2. It should be styled with more than just the default browser styling. It is recommended to use a CSS Framework such as Bootstrap, [Material Design Lite](https://getmdl.io/), or [Foundation](http://foundation.zurb.com/).
3. Regardless of which CSS Framework you use, your app must have a consistent and appealing look and feel.

**B. Hosting**

1. Any data from for your app must be stored in the browser’s localStorage.
2. Your page should be hosted on Github Pages. **Important**: To automatically deploy your app to Github Pages, run npm run deploy. You must also update the homepage property inside of the package.json file to be challenges-{your github user name} instead of challenges-jasonnutter.

**C. Pages**

1. There must be a React component named App where users can view the weather information. It should be rendered into the div with an id of root.

**D. Weather Page**

1. The page must display the name of your weather app at the top of the page and in the title of the browser tab.
2. There must be an input box where a user can search for a location by postal code or city name.
3. When the user submits a location search, your app must retrieve the current weather about that location from the [openweathermap.org API](https://openweathermap.org/current).
4. If the user provides a location that doesn’t exist, or the openweathermap.org API returns any other error when searching for the city, you must show an error to the user (e.g. “Invalid location”).
5. The page must show **at least** the following information about the weather:
   1. The name of the location (e.g. if the user enters “98115”, it must show “Seattle”).
   2. The “short” and “long” description of the current weather. (e.g. “Clouds” and ”scattered clouds”)
   3. An [icon](https://openweathermap.org/weather-conditions) for the current conditions.
   4. The temperature in Fahrenheit (with no decimal places).
   5. A button to Save that location to their saved locations.
6. When the user clicks the button Save the current location, the query the user provided for that location must be saved in a list of locations, and that list of locations must be visible to the user on the page.
7. The current list of saved locations must be saved to the browser’s local storage.
8. When the user clicks a previously saved location, the page must show the current weather for that location.
9. The user must have the ability to remove a location from the saved locations list.
10. If the user refreshes their page (or closes their browser and reopens your app), the page must show the previously saved locations.
11. Users should not be able to save queries that are invalid (i.e. an error was returned when searching for the location).
12. When the page first loads, if the user has any saved locations, the page should show the weather for the first location in the list.

**E. Technical Requirements**

1. All HTML elements in your app **must** be written in React. You may not add additional HTML elements to index.html directly (other than script, link, or other meta tags).
2. You cannot use any other JS libraries such as jQuery for DOM manipulation or traversal. You may use the jQuery AJAX API instead of fetch or XMLHttpRequest.
3. You must create separate React components for each main piece of UI (e.g. WeatherSearch, WeatherCurrent, WeatherSaved, etc).
4. Each React component must be written in its own file (1 component per file).
5. Your solution must not contain any errors reported by running npm run lint. Warnings are okay, however, it is recommended to fix all warnings as well.

**How to Submit your Assignment**

All code for this challenge must be committed to the **challenge7** branch in your repo. Push this branch to Github, open a pull request to your **master** branch (do not merge it yourself), and then submit the URL to your pull request on Canvas. In the description of your pull request, please include a link to your app on Github Pages, and please list any extra credit options you completed.

**Extra Credit**

* **Firebase** (5 points): Instead of using localStorage, store the saved locations on Firebase, using either anonymous authentication or traditional email/password accounts. If you provide functionality for a user to sign up/sign in, that UI must also be written exclusively in React (it can be on a new html page, which redirects the user to the weather page).
* **Styling** (5 points max): Style your app with more than just the default styling for your CSS framework.
* **Additional Data** (5 points max): In addition to the basic weather data, using the OWM API to display more information about the current weather in an informative, useful, and pleasing way.
* **User preferences** (5 points): Provide controls for the user to change their preferences (e.g. changing from temperature imperial units to metric units). These preferences must also persist between sessions.
* **Current location** (5 points): When the page first loads, show the weather data for the user’s current location. Alternatively, provide a button the user can click to load the weather data for their current location.