Random Quote Machine

This is a documentation of how I built this freeCodeCamp certification project. The app was run using **React** via **create-react-app**.

// 1: INSTALLED **Node.js**

Visited the Node.js website. Selected the *“Recommended For Most Users”*. Downloaded and installed Node, which also includes NPM (Node Package Manager). Link here: <https://nodejs.org/en/>.

For a need to check versions, run these:

node -v

then,

npm -v

Run the following on the terminal for necessary update to *Node.js* – must be 5.2.0 or newer to run **create-react-app**.

npm install -g npm

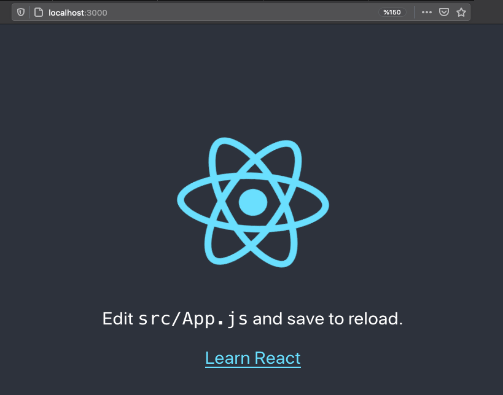
// 2: INITIAL APP INSTALLATION WITH **create-react-app**

Choose a directory. Run the following command:

npx create-react-app fcc-random-quote-machine

After a few seconds, initial configuration and directories were made *automatically*.

// 3: RUN THE INITIAL **app** SETUP

Go inside the created directory.

cd fcc-random-quote-machine

Run the **app** live into the browser:

npm start

The browser loaded this. It means the **create-react-app** initial installation was a success.

Proceed to start building the app with *React*.

// 4: PUSHED THE PROJECT TO MY GIT REPOSITORY

The app is pushed to the remote repo every time there are huge changes. For example: The initial installation, package manager installations, code addition/edition, etc.

// 5: INSTALLED **redux** AND **react-redux**

Within the **app** root directory. Ran the following code:

npm install redux react-redux

// 6: USE THE NEWER SYNTAX OF REACT

**Problem:** I used the older syntax of *React*, and simple text was not displayed on the client.

**Solution:** I adapted to the newer syntax on building *components*, it successfully displayed texts temporary in place for the **app** output.

// 7: STYLING

It can be done with inline styles.

I learned to import the separate stylesheet file to the component to apply style.

// 8: FONTAWESOME FONTS

Installed *Fontawesome* dependencies.

npm install --save @fortawesome/free-brands-svg-icons

npm install --save @fortawesome/free-solids-svg-icons

npm install --save @fortawesome/react-fontawesome

This is used for importing the *double quote* and *X* brand logo icons in *React*. Code below:

// Import statements

import { FontAwesomeIcon } from '@fortawesome/react-fontawesome';

import { faXTwitter } from '@fortawesome/free-brands-svg-icons';

import { faQuoteLeft } from '@fortawesome/free-solid-svg-icons';

// Icons as components – this is placed inside React components.

<FontAwesomeIcon icon={faQuoteLeft} />

<FontAwesomeIcon icon={faXTwitter} />

// 9: SIDE-EFFECTS MUST BE DONE AFTER RENDER OR RE-RENDER -**useEffect**

**Recap:** *React components* is coded out of pure functions – *stateful* and not. The *component* function must only calculate to the purpose of returning JSX. The JSX will be rendered be rendered to the DOM to update UI.

Any form of *manipulating* or *referencing* DOM element must be done after a successful render. This can be called *side-effects*.

The **useEffect** is a *React hook* designed for this purpose. It will run only when the *component* pure function has successfully returned and rendered JSX.

Any code different to the calculation of returning a JSX must be put inside this *hook*.

The **useEffect** must be provided with a proper *dependency*. Changes to that *dependency* will trigger the *effect* after a render. This is to avoid any unnecessary effects in every render.

The **useEffect** can have a *return statement*. This is often used to execute a *cleanup function*. The *cleanup function* is purposed to stop or undo what the effect is doing.

// 10: EXTERNAL RESOURCES – **public** folder

I learned that local resources must be manually placed inside of the *‘./public’* folder of the **create-react-app**.

My external fetches of the *quotes.json* was successful, it failed otherwise.

// 11: useState MUST BE SYNCHRONOUS

The *function* returned by **useState** to update *state* must be called *synchronously*. I used the *function* inside an **async** function and called it after the **await** process returns the value. The **async/await** allows to process *asynchronous* operations *synchronously.* This is when I used the data returned by the *Fetch* API as an update to my quotes *state*.