Course: CPRG-352
Created by: Nick Hamnett
Date: April 3, 2022

Lab 10

Introduction

It's recommended you have the following installed to complete this lab:

1. Visual Studio Code
2. GitHub Desktop
3. React Developer Tools (Chrome or Firefox)

React is not an easy technology to learn. This lab goes into React without going into what React is built off, which is server-side JavaScript (aka NodeJS) and Webpack. There are various resources you may want to look at before doing this lab:

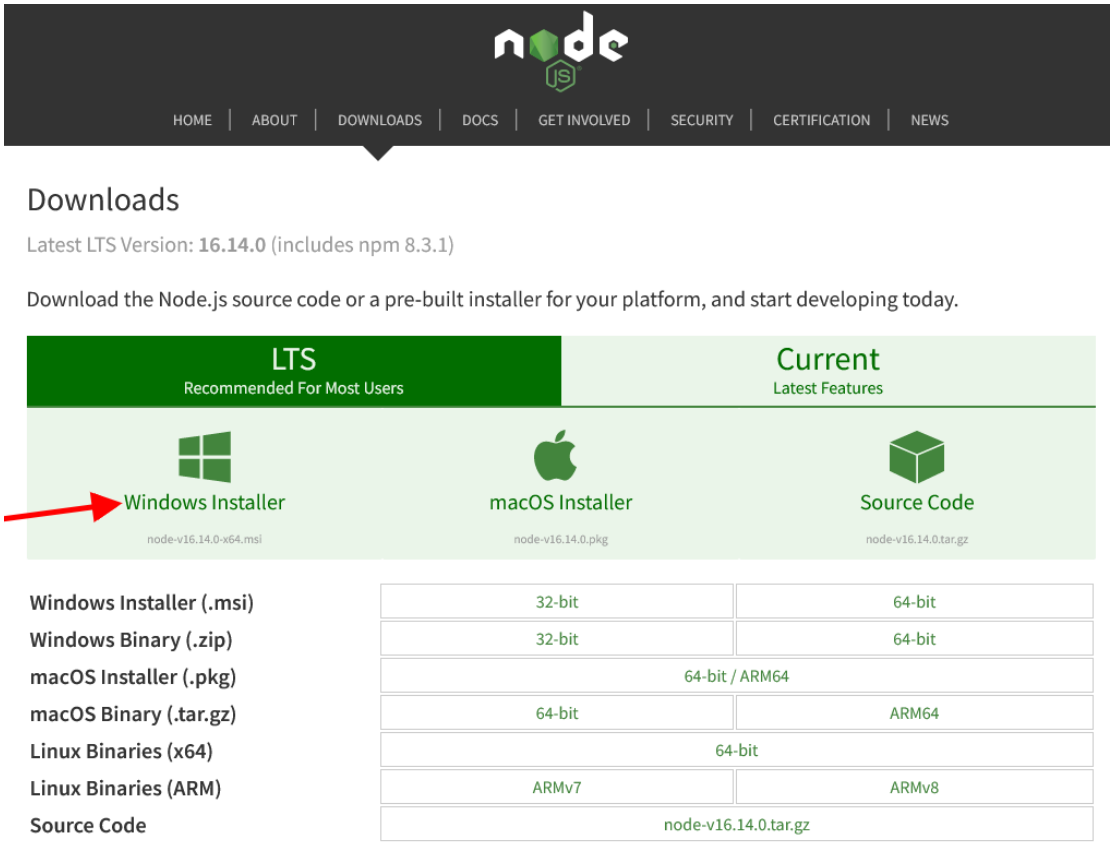
- https://www.youtube.com/watch?v=nt9M-rlbWc8
- https://www.codeinwp.com/blog/webpack-tutorial-for-beginners/
- https://www.pluralsight.com/blog/software-development/6-essential-skills-for-react-web-developers

It's recommended that you upload your solution to this lab on GitHub. It's especially important that you do multiple GitHub commits to both keep track of your progress and to show potential employers how you approach problems.

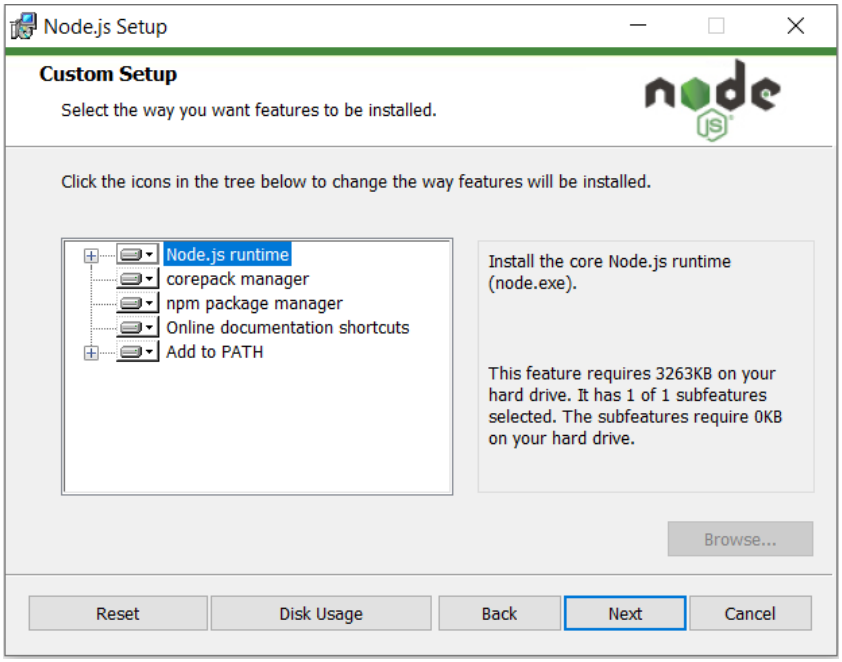
NodeJS

Download and Install NodeJS

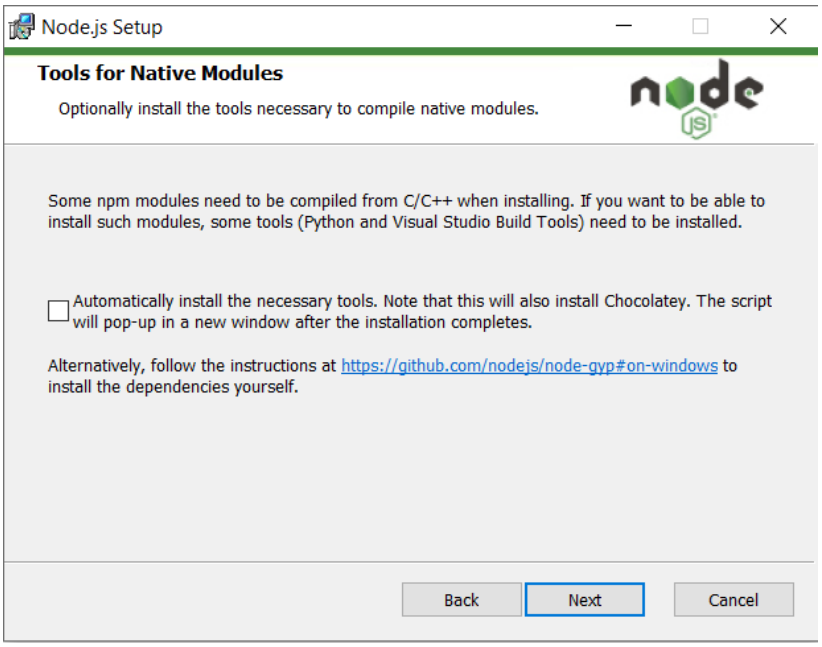
1. Go to https://nodejs.org/en/download/.
2. Select the “LTS” tab.
3. Click “Windows Installer” to download NodeJS for Windows.



4. Open the downloaded NodeJS installer and click “Next” to go to the license agreement screen.
5. Read through the end-user license agreement, check the “I agree” checkbox, and then click “Next”.
6. Leave the installation/destination folder as the default and click “Next”.
7. The next screen allows you to choose what features to include with NodeJS. Leave all the options as selected.



8. You will be asked to install the “Tools for Native Modules”. This is used for writing C or C++ code that will be called by JavaScript code. We won’t be needing this so leave the checkbox unselected.

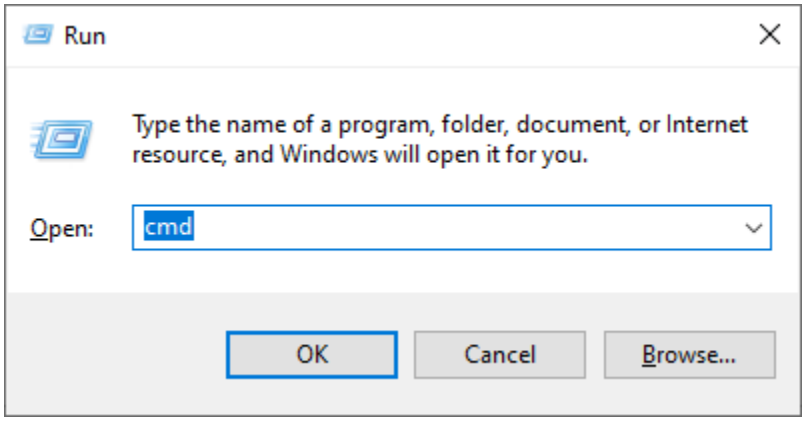


9. In the next screen, click “Install” to start installing NodeJS.
10. Once NodeJS is installed, click “Finish” to exit the installer.

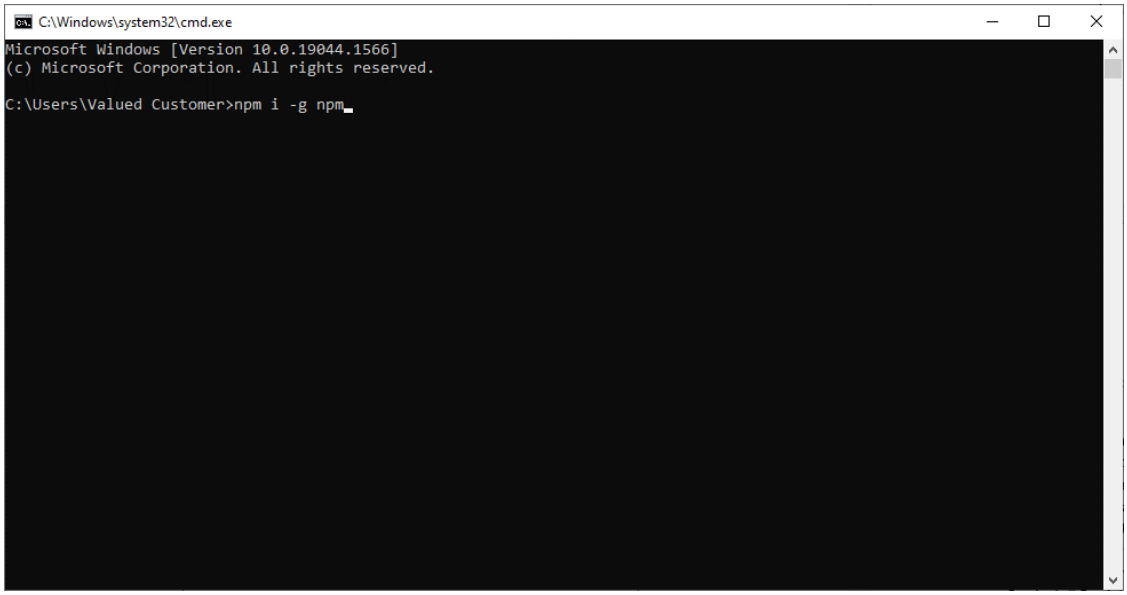
Configure the Node Package Manager (NPM)

The Node Package Manager (often abbreviated as NPM) provides developers with access to 1000s of different NodeJS packages. We will be using NPM to create our React app, so we need to make sure it is setup, configured, and up to date:

1. Open the command prompt. To do this, press the Windows key + R, enter "`cmd`" and click "OK".



2. In the command prompt, enter "`npm install --global npm`" or "`npm i -g npm`" to ensure NPM is the latest version.



Create React App

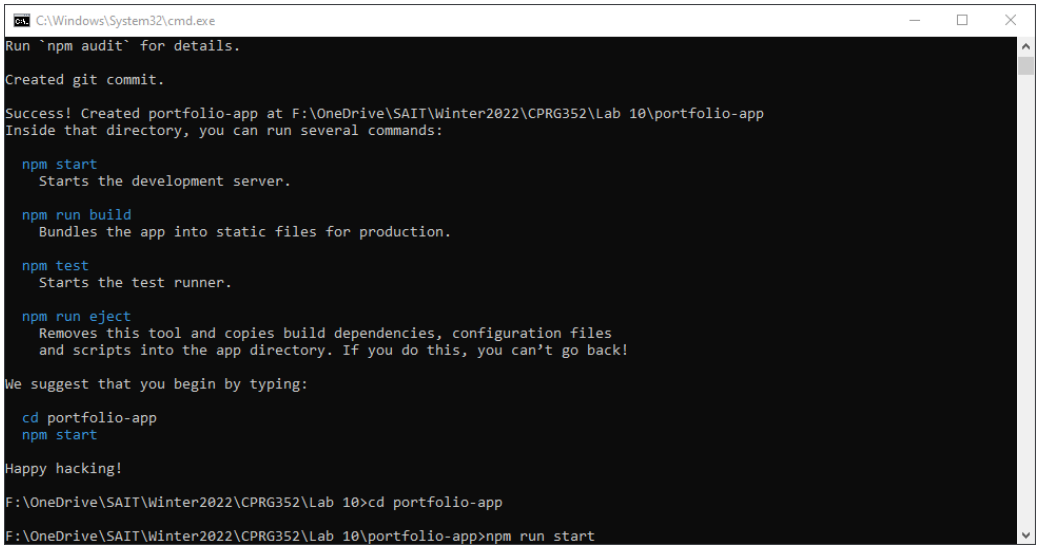
The first thing we're going to do is create a React app from an existing template.

1. Open the command prompt in the directory where you want to create the React app.

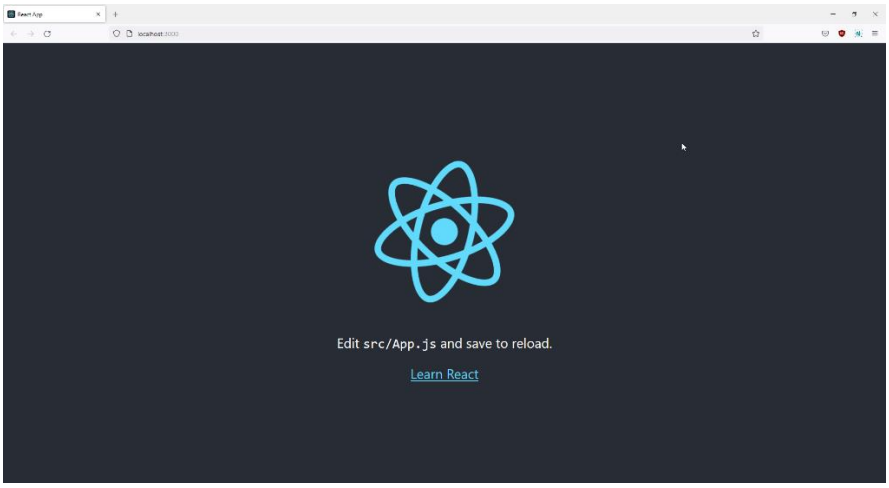
2. The next step is creating the template for the React. There are two options here:
   a. Enter "`npx create-react-app portfolio-app`" into the command prompt to create a React + JavaScript template. This uses JavaScript eXtended (JSX), which allows HTML tags to be embedded in JavaScript (with some subtle differences with attributes).
   b. Enter "`npx create-react-app --template=typescript portfolio-app`" into the command prompt to create a React + Typescript template. You are on your own to learn TypeScript. The TypeScript Handbook has great information on learning TypeScript with (or without) existing knowledge of JavaScript or Java.

3. Change into the directory of the React app using the command "`cd portfolio-app`".

4. Run the newly created React app with the command "`npm run start`".



5. Your web browser should open with your React app.



Hello World (with Bootstrap)

Now that you've seen the nice and pretty React webpage, we're going to clear it all out to build our own webpage with Bootstrap.

To do this, open the "src/App.js" file and:

1. Delete the JSX/HTML code inside the return statement, leaving the `<div className="App">` container tag.

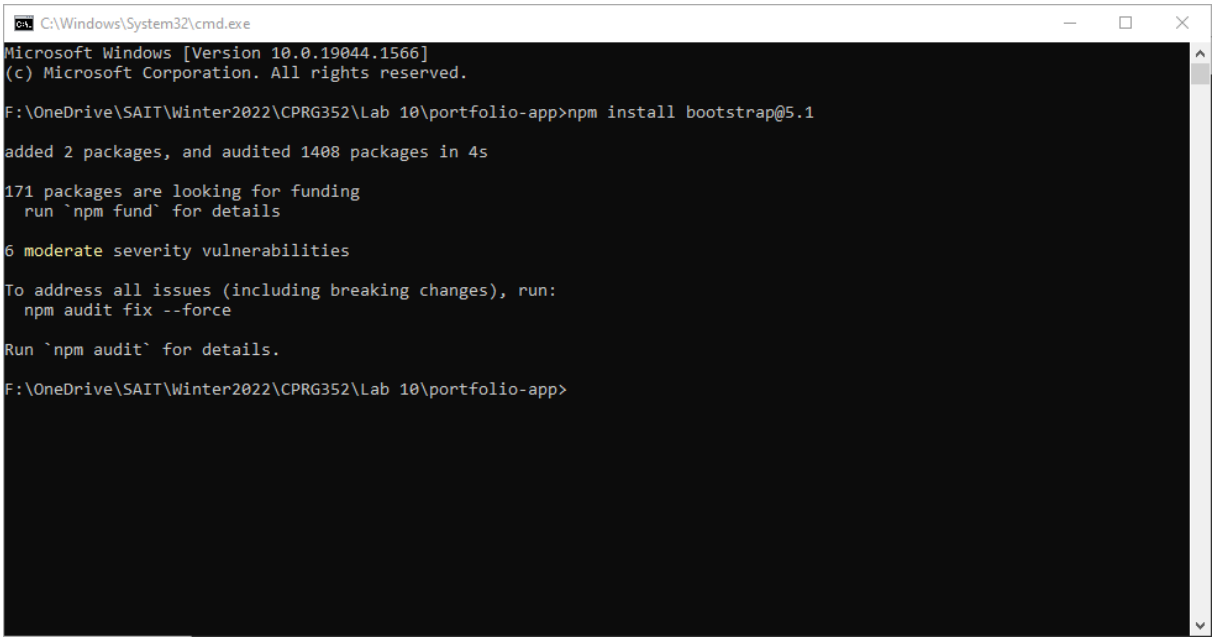
2. Since we're not using the React logo anymore, remove the top line `import logo from './logo.svg';` from the source code. You can also delete the logo.svg file from the src folder.

3. Insert `<h1>Hello World!</h1>` into the `<div>` container.

Bootstrap is an existing set of CSS styles and JavaScript code that makes it easier to build beautiful responsive websites. Bootstrap Icons provides well, icons. We need to install the needed NPM packages. To do this:

1. Open the command prompt in the same folder as “portfolio-app”.

2. Install Bootstrap v5.1 and Bootstrap Icons v1.8 with the command:
   `npm install bootstrap@5.1 bootstrap-icons@^1.8.1`

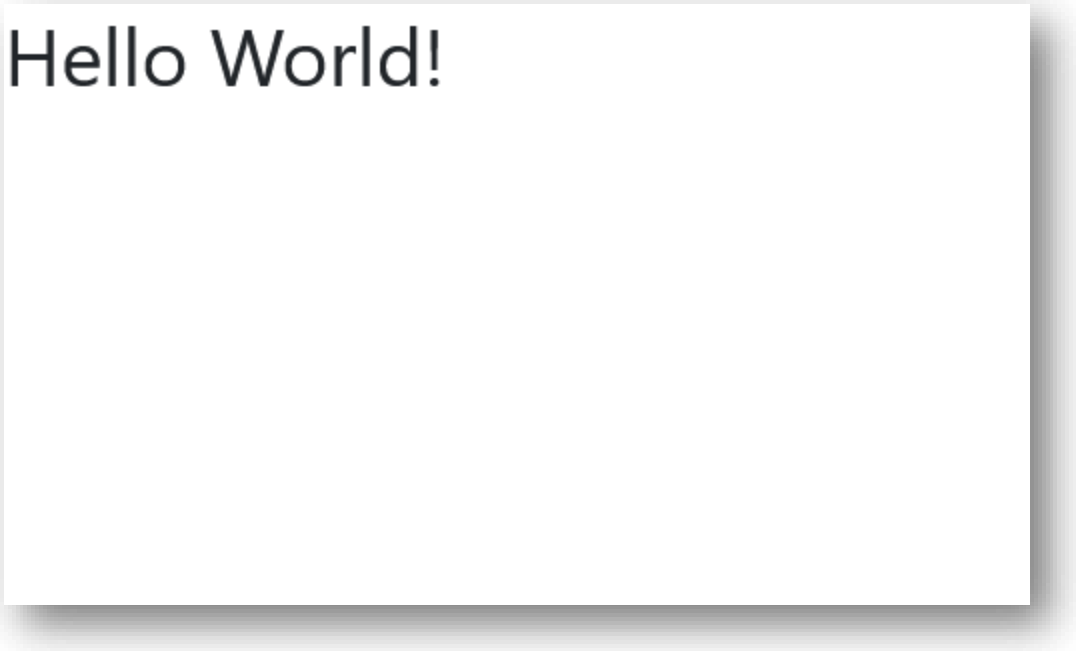


3. Open the “src/App.css” file and:

   a. Delete everything inside of it.

   b. Replace the contents with:
      ```
      @import "~bootstrap/dist/css/bootstrap.css";
      @import "~bootstrap-icons/font/bootstrap-icons.css";
      ```

4. Start your React app (if it's not already) and it should look like the following (without the outside border):



Checkpoint

You may want to save your React app (in its current state) to a GitHub repository so you can easily spin up a React + Bootstrap app in the future.

Static HTML to React

Some developers prefer to come up with a static HTML webpage template before changing it to React. It makes it easier when you're not experienced with React and it's simple to copy + paste the code into React.

Open the "index.html" file in the provided resume template.

Copy all the HTML inside the <body> tag:

```
<!DOCTYPE html>
<html>

<head>
    <title>Joe Blow</title>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet"
        integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3" crossorigin="anonymous">
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.7.2/font/bootstrap-icons.css">
    <link href="style.css" type="text/css" rel="stylesheet">
</head>

<body>
    <header class="text-white bg-dark">
        <div class="container py-5">
            <div class="row">
                <div class="col-xs-3 col-lg-2 text-center">
                    <img src="https://www.seekpng.com/png/detail/110-1100707_person-avatar-placeholder.png"
                        alt="Person - Avatar Placeholder@seekpng.com" class="border border-5 rounded-circle img-fluid"
                        style="max-width: 150px; max-height: 150px;">
                </div>
                <div class="col text-center text-lg-start">
                    <h1>Joe Blow</h1>

                    <p class="lead">Software development student at SAIT (Southern Alberta Institute of Technology).</p>
```

Paste it inside the container tag in the "src/App.js" file of your React app.

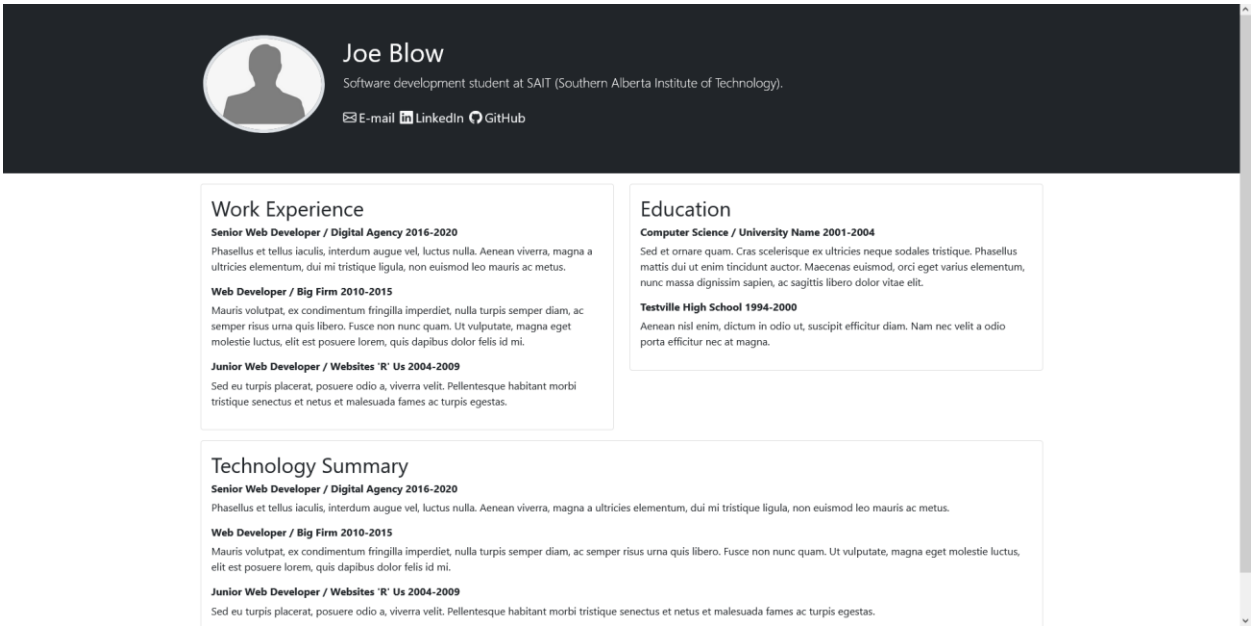
To get the pasted HTML code work with JSX you will need to make some corrections:

- Make the `<img>` tag self-closing by changing the end from `>` to `/>`.

- Change the `class` attribute names to `className`. You may not get an error about this, but JSX uses *className*, not *class*.

- Check out https://www.freecodecamp.org/news/html-vs-jsx-whats-the-difference/ to see if there is there's anything else that needs to be fixed. Try starting the React app to see if there's any Webpack errors in the command line. Also check the Web Developers Tools console in your web browser to see if there's any errors there.

Your React app should look like the following after following the previous steps correctly:



Separate Out Components

One of the advantages of React is the ability to not just separate out the different web pages, but also separate and re-use different parts/components inside a webpage.

First, place the header in a separate component by doing the following:

1. Create a file called “Header.js” in the src/ folder.

2. Add the following to the “src/Header.js” file.

```
function Header() {
    return (

    );
}

export default Header;
```

3. Cut the <header> tag (and everything inside it) from the src/App.js file.

4. Paste the <header> tag inside of the return statement (leaving the round brackets) in the src/Header.js file.

```
function Header() {
    return (
        <header className="text-white bg-dark">
        <div className="container py-5">
            <div className="row">
                <div className="col-xs-3 col-lg-2 text-center">
                    <img src="https://www.seekpng.com/png/detail/110-1100707_person-avatar-placeholder.png"
                        alt="Person - Avatar Placeholder@seekpng.com" className="border border-5 rounded-circle img-fluid" style={{ maxHeight: "150px" }} />
                </div>
                <div className="col text-center text-lg-start">
                    <h1>Joe Blow</h1>

                    <p className="lead">Software development student at SAIT (Southern Alberta Institute of Technology).</p>

                    <ul className="list-inline">
                        <li className="list-inline-item my-2"><a href="#" className="fs-5 link-light text-decoration-none"><i
                            className="bi bi-envelope pe-1"></i>
                            <span className="text-center">E-mail</span></a></li>
                        <li className="list-inline-item my-2"><a href="#" className="fs-5 link-light text-decoration-none"><i
                            className="bi bi-linkedin pe-1"></i>
                            LinkedIn</a></li>
                        <li className="list-inline-item my-2"><a href="#" className="fs-5 link-light text-decoration-none"><i
                            className="bi bi-github pe-1"></i>
                            GitHub</a></li>
                    </ul>
                </div>
            </div>
        </div>
    </header>
    );
}

export default Header;
```

Next, import and render the Header component in the src/App.js file by doing the following:

1. Import the Header component by adding the following to the top of the file:
   ```
   import Header from './Header.js';
   ```

2. Render the Header component by placing the following where the <header> tag was:
   ```
   <Header />
   ```

Separate out other components from the App component. You can create components for the following parts of the webpage:

- Work Experience Card
- Education Card
- Technology Summary Card

Customizing

Customize the resume to your liking. You can:

- Insert an actual (appropriate) picture of yourself. Add the image file to the src folder and see https://create-react-app.dev/docs/adding-images-fonts-and-files/ on how to link to it from JSX.

- Change the name to your name.

- Add links for your e-mail, LinkedIn, and GitHub accounts.

- Add links to other websites you have a profile on.
  See https://icons.getbootstrap.com/#icons for a list of different icons to use.

- Include a technology summary (technologies you know) and/or a portfolio (projects you've worked on).

Further Improvements

There are various things you can do to improve this app beyond what's covered in this lab:

- Experiment with props (short for properties), which allow you to specify your own HTML attributes and content for your own components. For example: Rather than having the work experience items hard coded in the component, specify properties in the main App component that the Work Experience component will display.

- Experiment with state. As an example: Have the header background and foreground switch between light and dark mode depending on the time of day, when the user clicks a button, or on random.
    - If you're feeling extra adventurous, introduce a global state (rather than a state for each component) to your web application with React Redux. This article contains information on the different types of state and how to handle each one with React.

- Try creating React class components (rather than function components). The components are rendered the same, the difference is how the components are defined.

- Add a contact form that users fill out to send you e-mails. This tutorial goes over creating a form that is processed using ExpressJS and Nodemailer.

- Use the Reactstrap component library instead of ordinary HTML tags to easily integrate with Bootstrap.

- Integrate another third-party React component library. There's 1000s of different libraries available, you just must make sure you choose one that will work.

- Host your React app on a live webserver. Take advantage of the GitHub Student Developer Pack to get a free domain name and web hosting.

- Come up with your own portfolio design template. Also available with the GitHub Student Developer Pack is Bootstrap Studio, which allows you to easily design websites with Bootstrap.

- Let loose. Change it up to whatever you're liking. Potential employers may want to look at your React app and judge it based off how much effort you put into it.

Final Product

Use the following GitHub repository as an example final product:

https://github.com/nick-hamnett/portfolio-app-final