MP3

In this **Mini-Project**, you will recreate your **MP-1** assignment, either from scratch or by copy/paste in **React**. Regardless of your decision, as to copy/paste or start from scratch, your new "**React Resume**" must look and function exactly like the way it did in **MP-1**. If in **MP-1** there were bugs and/or other issues please fix them here; also, if you lost points for not following instructions on **MP-1** correct those mistakes so you don't lose points again.

Task-1:

Your <u>Online Resume</u> from **MP-1** had many files (e.g., 6 **HTML** files, couple of **CSS** files, and some **JS)**. One of the main purposes of **React** is to reduce **DRY** (Do Not Repeat Yourself). So, you should <u>identify</u> every repetitive code in your <u>Online Resume</u> and replace it with a **React Component/Hook**. For example, all of your internal pages had a **Header**, a **Nav**, a **Main**, and a **Footer**, but except for the **Main**, all the other sections are the exact duplicate of themselves in every single internal page. So currently (in <u>MP-1</u>) you must have about est. (**7 x identical-Headers**), (**7 x identical-Navs**), and (**7 x identical-Footers**).

In **React** you get to create those repetitive sections of your code, using a **React Components/Hooks**, and <u>recycle</u> them as many times as you need.

```
 1   import React from "react";
 2   import styled from 'styled-components';
 3
 4   const HeaderWrapper : …  = styled.header`
 5       position: relative;
 6       background-color: orange;
 7       display: flex;
 8       flex-direction: column;
 9       align-items: center;
10       justify-content: center;
11       font-size: calc(2px + 2vw);
12       color: white;
13   `;
14
15   const StyledButton : … =styled.button`
16       position: absolute;
17       z-index: 1;
18       top: 6vh;
19       right: 1.75vw;
20       background-color: crimson;
21       border-radius: 35%;
22   `;
23
     2 usages  new *
24   export function Header({onLoginClick}){
25       return(
26           <HeaderWrapper>
27               <h1>Taymaz Davoodi</h1>
28               <p>My Online Resume</p>
29               <StyledButton type="button" onClick={onLoginClick} id="log-in-btn">
30                   Login
31               </StyledButton>
32           </HeaderWrapper>
33       )
34   }
```

```
10       <!--Header starts here:-->
11       <header>
12           <div id="header-parent">
13               <div id="header-child-1">
14                   <h1>Taymaz Davoodi</h1>
15               </div>
16               <div id="header-child-2">
17                   <p>My Online Resume</p>
18               </div>
19           </div>
20       </header>
21       <!--Header end-->
```

- All of the following HTML tags, <u>should be converted to</u> **Components**:
  <header></header>
  <nav></nav>
  <main></main>
  <footer></footer>

- Regardless of approach, your code <u>should not have more than one</u> **HTML** file (**index.html**).

- All of your <u>React components should be rendered</u> in your **App.jsx/tsx** file and you should try to compartmentalize your file and folders so that related files are grouped together.

- In your **MP-1**, you had implemented a **JavaScript Calculator.** You <u>must replace your repetitive functions inside your JS Calculator with</u> **Hooks/Custom Hooks**.

- Remember that when you need to change the **State** of one component from another component, you have to use React's **useState Hook**. Your <u>Components must adhere to the</u> **Single-Tasking** principal/convention, (you may **not** have more than one UI/component per file).

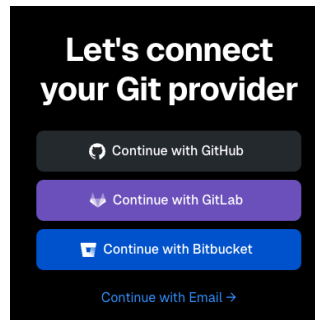- You <u>must use React's</u> **RouterProvider** to link your Navigation Component to the Browser Router.

<u>Task-2</u>:

Deployment:
1. In your terminal, navigate to the root of the directory of your project (**mp-3**) and use the command **git init**
2. Use the command **git status** to see the current status of the files within your directory. Right now, all files should be untracked
3. Use the command **git add** *filename* or **git add .** (to add all files). This tells Git about any changes you've made to this file since its last commit.
4. Once you have added all relevant files, use the command **git commit -m "*your commit message*"** to commit your changes. This confirms your staged changes with Git. Be sure to include a helpful/informative commit message when using this command.
5. Create a new GitHub repository by visiting https: github.com and clicking **New** on the left next to **Top Repositories**
6. Give your repository the same name as the directory you're working in (**mp-3**) and click **Create repository**
7. Follow the directions under **or push an existing repository from the command line** in your terminal
8. You may be prompted by Git to use the command **git push –set-upstream origin main**. If you are, do so.
9. Your files should now be on GitHub! Refresh your GitHub tab if you don't see your files.
10. Once you saw your files, **Save the URL**:
    (e.g., *https://github.com/YOUR_USER_NAME/mp-3*).

Hosting:

1. Visit https://vercel.com/signup and set up a free (Hobby) account (you might have done this already in **Dis-1**).

2. Connect your GitHub account to your Vercel account.

3. Once connected all of your GitHub repositories will be listed in Vercel. Select **mp-3** from the list and push the **import** button next to it.

4. In the next screen you don't need to change anything; just click **Deploy**. Vercel will take a few seconds to process the files and will prompt you with a "Congratulations" message and a screenshot of your hosted app.

5. Click on the screenshot, which should open a new browser-tab, **save its URL**: (e.g., _https://mp-3.vercel.app_).

Submission:

- Navigate to Blackboard >> Mini Projects >> MP-3 >> Create Submission

- Paste both saved URLs in the text filed:

  (e.g., _https://github.com/YOUR_USER_NAME/mp-3_).

  (e.g., _https://mp-3.vercel.app_).

- **You Link might be different from the ones above, so make sure that both open to your assignment before you submit them.**

- Add comments if you like, and click **submit**.

Note:

DON'T _UPDATE/PUSH/change_ your assignment's GitHub repo after the **due-date**. **Altering your assignment after its due-date is considered cheating** and at minimum, will result in an automatic 0 on that assignment.

You may change or alter that repository after you received a grade on that assignment.