ENSF381 – Lab08

Section: L01/L02/L03 *[Keep only one]* *[They must keep only one, if they made mistake write a note in the feedback section. Don't deduct any mark for wrong value]*

Date: 2024-03-11 *[They must write the correct date. For L01 March 11, for L02/L03March 13. If they made mistake write a note in the feedback section. Don't deduct any mark for wrong value]*

Created by:		
First name	Last name	UCID
A	B	123
C	D	456

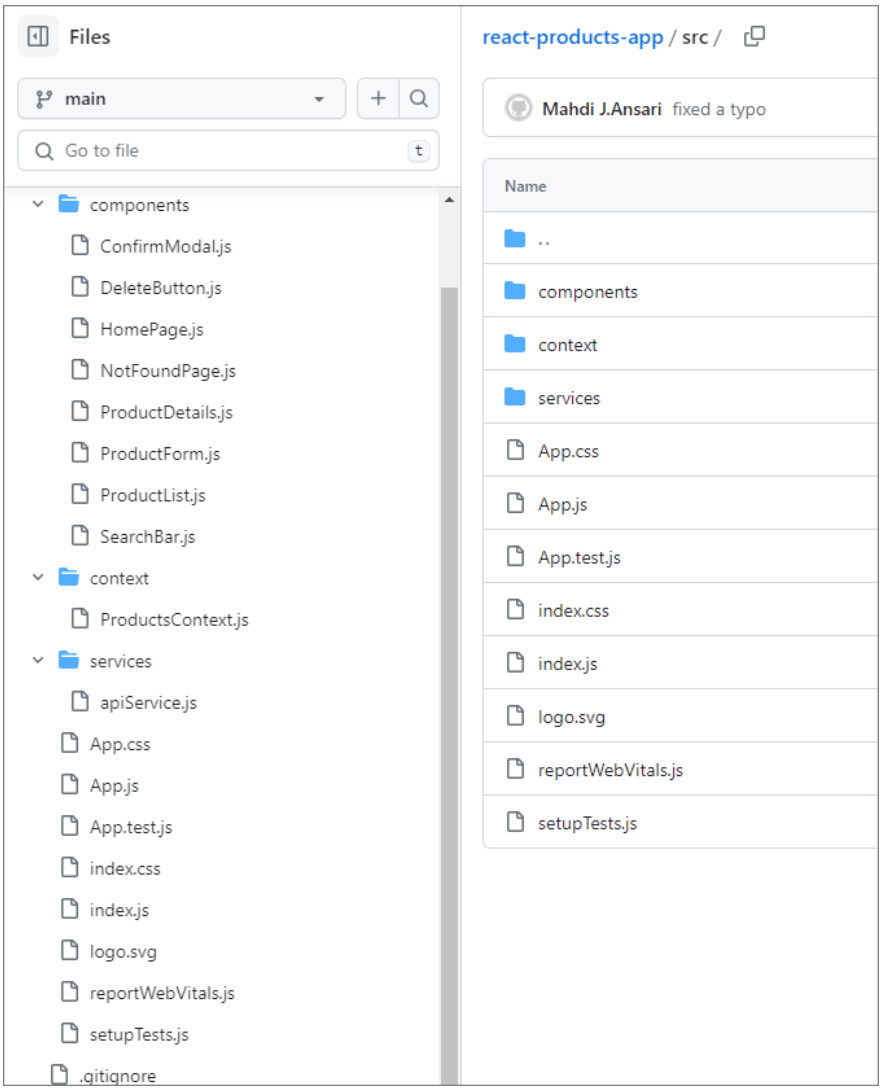
[Check the name. If they made mistake write a note in the feedback section. Don't deduct any mark for wrong name or wrong UCID]

2.4.1

They must provide a GitHub address. For example, https://github.com/mjza/react-products-app.

It must be public with at least 2 members.

It must have the following structure:

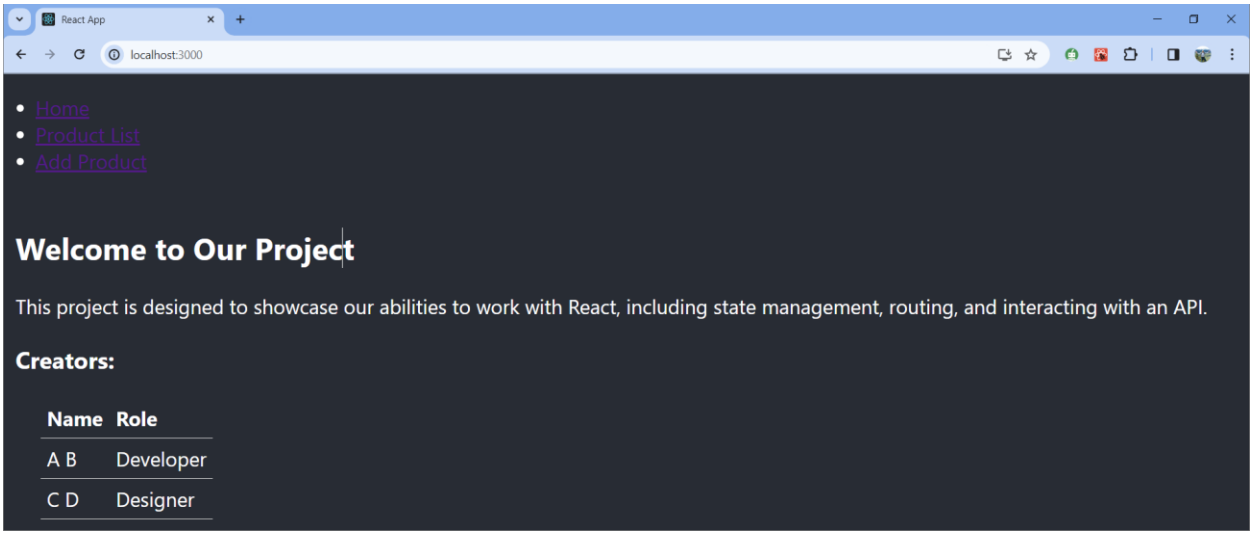


It must have 3 commits with these comments (Initialize project using Create React App, Step B, Step C)

If all conditions have been provided, then give 5 marks. For each missing point deduct 0.5 marks and comment it in the feedback box.

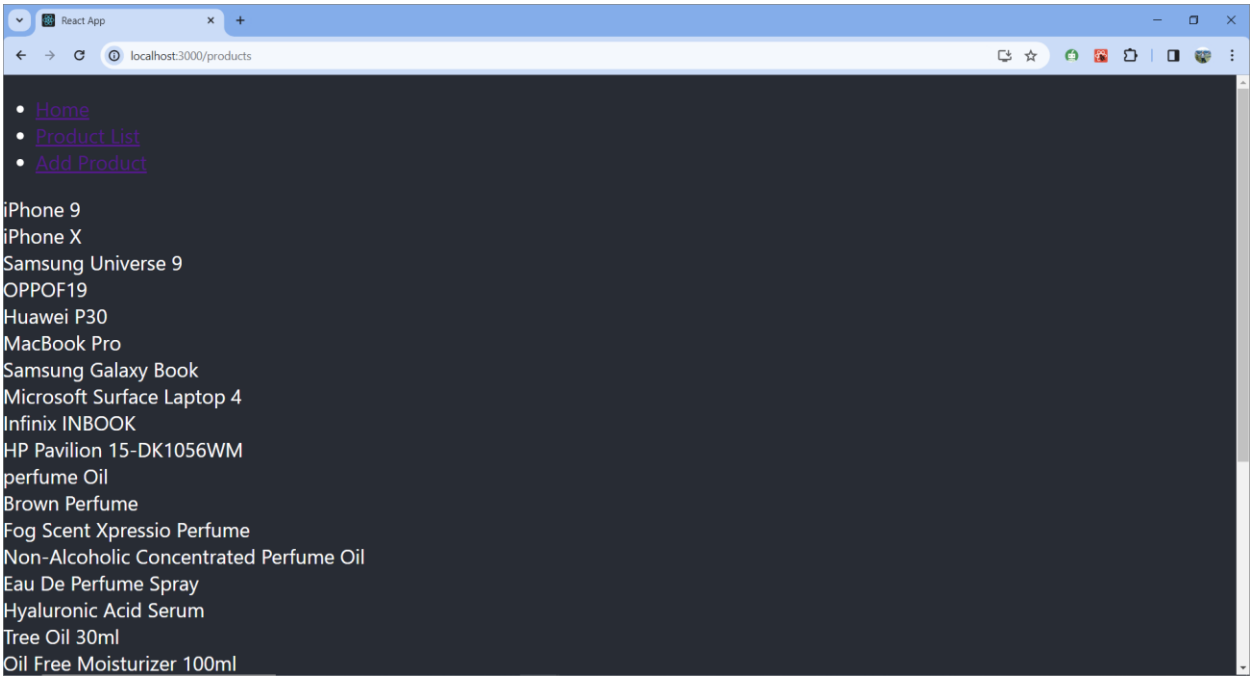
3.8.3

[5 marks]



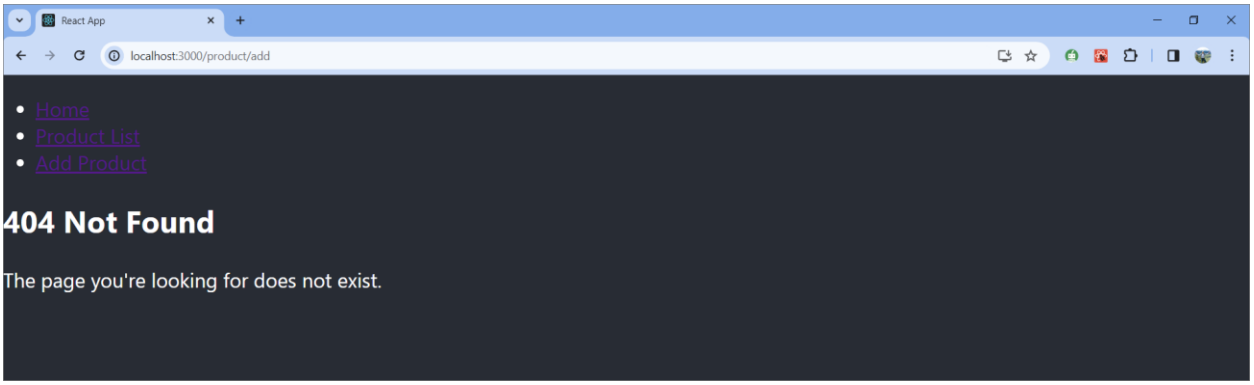
3.8.4

[5 marks]



3.8.5

[5 marks]



3.8.6

[10 marks, they must provide 2 pros and 2 cons, each 2.5 marks. They must provide a valid reference for their answer. Search engines, ChatGPT and Wikipedia or Me or TA are not accepted reference. Check the validity of reference. Sometimes they just drop a URL. If the reference is not valid, give a zero.]

Here are some sample answers:

Using a JSON object for styling in JavaScript applications, often seen in React projects, involves defining CSS styles within a JavaScript object, as shown in your **styles** object example. This approach, known as inline styling, when applied via the **style** attribute, has several pros and cons.

Pros

1. **Local Scope:** Styles defined in a JSON object are scoped to the component they are defined in. This minimizes the risk of style conflicts across the application and ensures that styles are not inadvertently overridden by other CSS.
2. **Dynamic Styling:** Since styles are just JavaScript objects, it's straightforward to dynamically alter styles based on component state or props. This can be particularly useful for conditional styling based on user interactions or data.
3. **JavaScript Power:** Utilizing JavaScript for styling means you can use variables, functions, and conditions to generate styles, making your styling more adaptable and potentially reducing repetition.
4. **Co-location:** Keeping styles close to their respective components can improve readability and maintainability, especially in component-based architectures like React. It's easier to see immediately how a component is styled without having to switch between files.

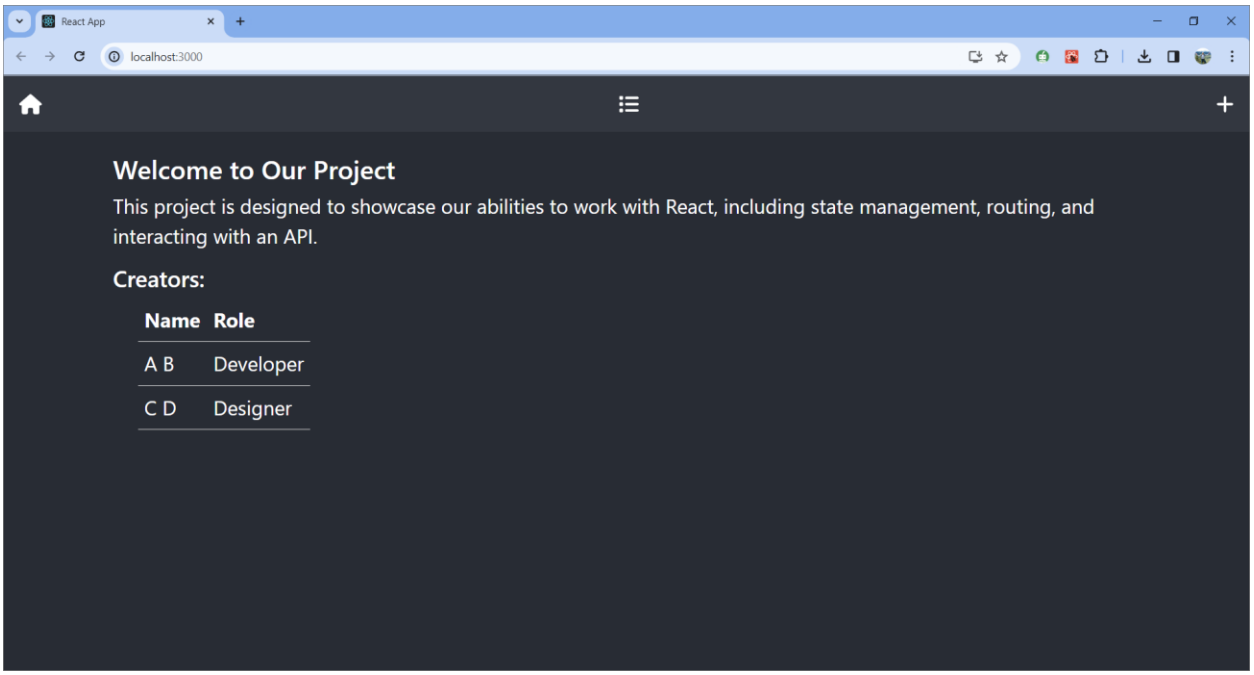
Cons

1. **Performance:** Inline styles can lead to slower performance compared to traditional CSS stylesheets. Each element with inline styles is treated as a unique entity by the browser, which can increase the time it takes to render components, especially in large applications.
2. **Reusability:** Styles defined in a JavaScript object are harder to reuse across different components without passing the style object around or duplicating the style definition. This can lead to less efficient code and potential inconsistencies.
3. **CSS Features Limitation:** Some CSS features, like pseudo-classes (**:hover**, **:active**, etc.) and media queries, cannot be directly implemented within inline styles and require workarounds, such as additional JavaScript logic or combining with external stylesheets.
4. **Separation of Concerns:** While co-location (keeping styles with components) can be seen as an advantage, it also blurs the lines between styling and functionality, potentially complicating component logic and violating the separation of concerns principle.

Reference: example.com or some papers

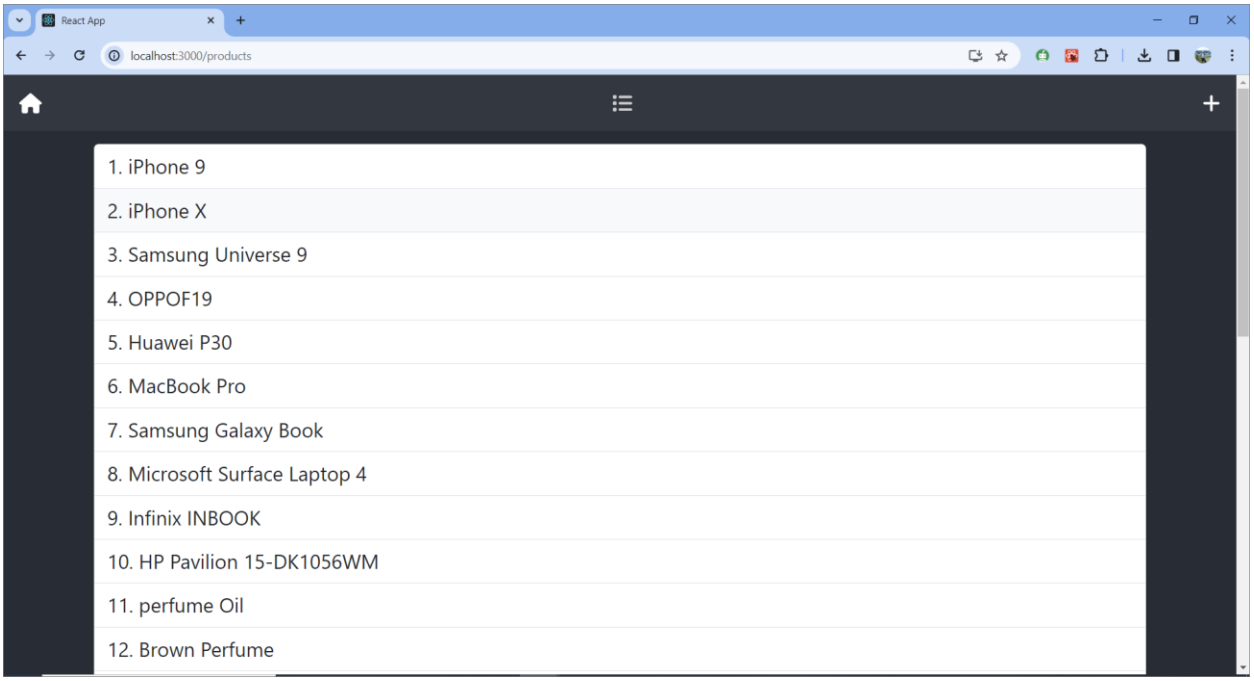
4.3.3

[5 marks, instead of A B and C D their names had to be replaced]



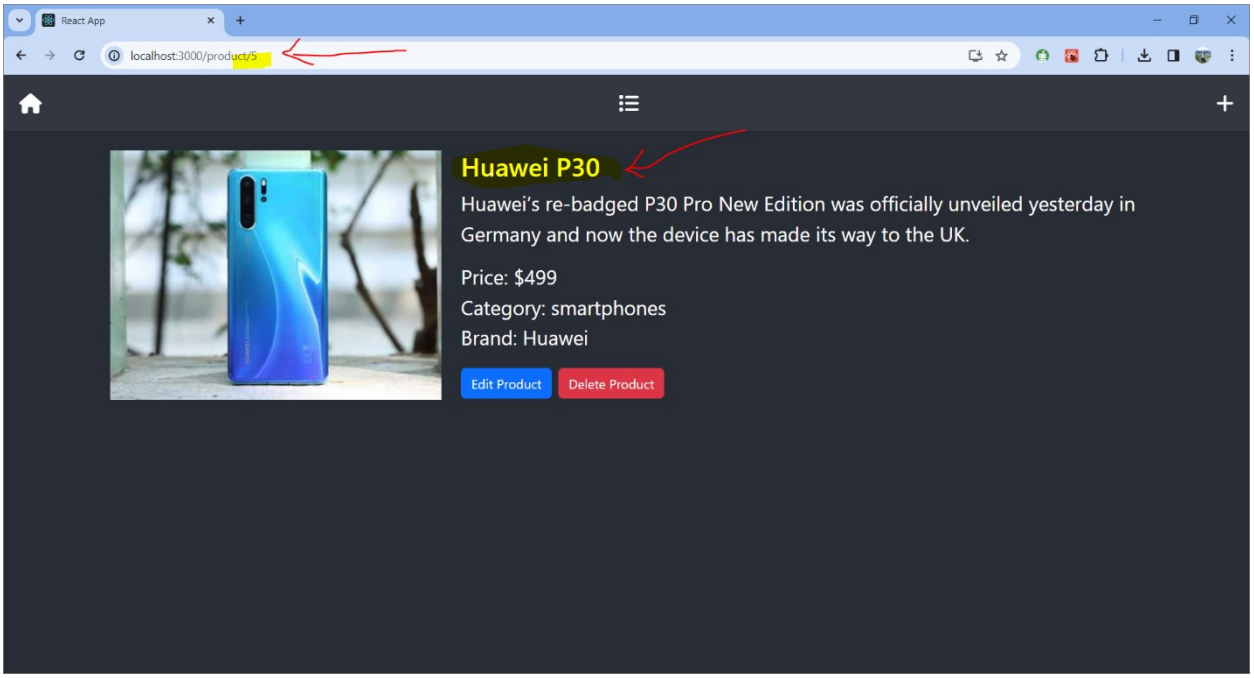
4.3.4

[5 marks]



4.3.5

[5 marks, notice to the url and title. They must be same]



4.3.6

[5 marks, notice to the url and title. They must be same]