



COMSATS University Islamabad (CUI), Lahore Campus
Department of Electrical & Computer Engineering

CSC336 – Web Technologies

Lab Manual v 1.0.1

Lab Resource Person

Muhammad Babar Ali

Theory Resource Person

Muhammad Babar Ali

FALL 2024

Name:

Registration No:

Program:

Batch:

Semester:

Laboratory 07

To reproduce a web app using React.js, with Component reuse and Virtual DOM

INTRODUCTION TO REACT.JS COMPONENT REUSE AND VIRTUAL DOM

1 Objective

The purpose of the lab is to introduce students to the most famous front-end web development library, called [React.js](#). Students will learn how to create react application from scratch. They will learn how to introduce react library in their application. They will also learn basic structure of a react application along with basic concepts of [react library](#).

2 Introduction to React.js

2.1 Why use React.js?

In the previous laboratory activities, you have learned Hypertext Markup Language ([HTML](#)), Cascaded Style Sheets ([CSS](#)) and bootstrap ([Bootstrap 5.0](#)). The development of production grade websites with only HTML and CSS is quite time-consuming, however Bootstrap framework is quite helpful in reducing development time. The “React.js” library can be used to reduce the development time even further with the help of its basic features such as components, states, and props. React.js also improves code reusability and shifts the programming paradigm from declarative coding (used in Vanilla Java Script) to imperative coding (used in React.js).

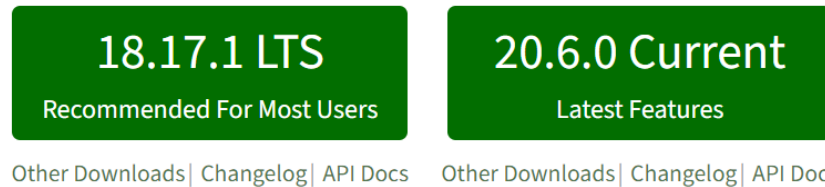
The react library is platform independent. A React application can be rendered on various platforms such as chrome (a web browser) or android (operating system). To enable any platform, react library is used along with the platform-specific library to render the contents on the user interface. The previous and current laboratories in the course uses web browser as a target platform. To enable this rendering, another library namely “ReactDOM” library will also be included in your application.

2.2 Pre-requisite for React.js

The react.js application uses Node.js to build the application locally on computer. You might have installed Node.js already in previous laboratory activities. If so, you can skip the discussion here and move to the next subsection. However, if did not install node.js already than visit <https://nodejs.org/> and download (and install as well) the latest long-term support (LTS) release version as shown in Figure 2.1.

Node.js® is an open-source, cross-platform JavaScript runtime environment.

Download for Windows (x64)



For information about supported releases, see the [release schedule](#).

Figure 2.1: Target view of the <https://nodejs.org/>

After the installation is complete, verify it by entering following command in command prompt available in the operating system:

```
$ node --version
```

(The output of this command must show the version of node.js, installed on your computer)

2.3 Creating React Application

The most common ways of creating react application are:

- “create-react-app” ([CRA](#)) – the npm tool
- Vite – A build tool
- Next.js – A full-stack React framework
- Remix – A full-stack React framework
- Gatsby – A React framework

Please note that the following discussion and the upcoming laboratory activity will use the first approach namely “create-react-app” to build an application.

3 Basic Concepts of React.js with an Example

This section will describe the basic concepts of React.js with step-by-step instructions in an exemplary application named “lab-react-app”.

3.1.1 Creating react application with “create-react-app”

Visit the website, <https://create-react-app.dev/docs/getting-started>. Read the given instructions carefully to create React application. In summary, you can use the following instruction after navigating to desired folder (neither desktop, nor a folder on desktop) where you want to create the react app:

- `$ npx create-react-app <new-app-name>`

The *<new-app-name>* can be any name suggested for the new react application which must not use capital case alphabets. The above instruction will install React.js — react and react-dom — locally and will create fresh React.js app in a folder *<new-app-name>*.

Figure 3.1 shows instructions to create an application namely “lab-react-app” in existing folder called “ReactApps”. Proceed with the following steps:

1. Create a folder named “ReactApps” in D drive on windows
2. Open this folder in VS code
3. Open integrated terminal in VS code – you will be at location D:\ReactApp
4. Insert the second command ***npx create-react-app lab-react-app*** to create the react application named “lab-react-app”

```
PS D:\ReactApps> npx create-react-app lab-react-app
```

Figure 3.1: Commands for creating application, “lab-react-app”

Once the above steps are complete, a folder named “lab-react-app” will be created in “ReactApps” folder.

To run the created application, navigate to the application directory namely *<new-app-name>* and run the application using following commands respectively:

- ***\$ cd <new-app-name>***
- ***\$ npm start***

Figure 3.2 shows instructions to navigate to the application folder and starting the development server.

```
PS D:\ReactApps> cd .\lab-react-app\  
PS D:\ReactApps\lab-react-app> npm start
```

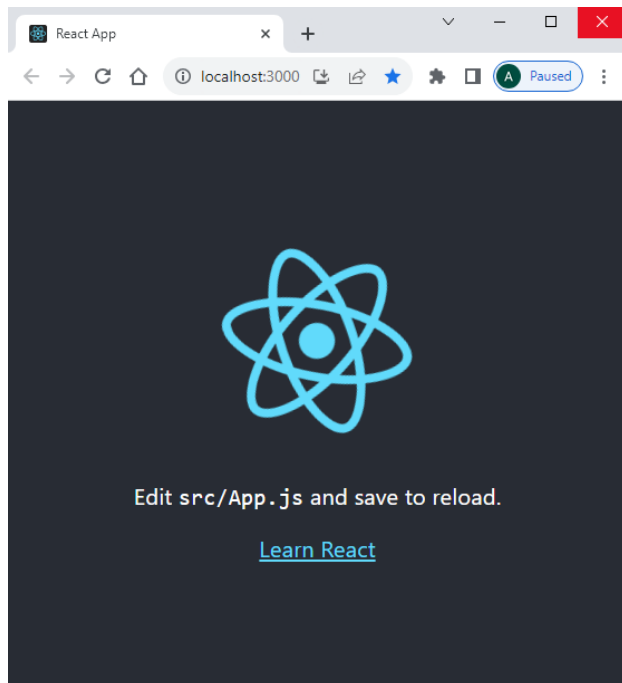
Figure 3.2: Navigating to “lab-react-app” and starting development server

Once the development server started, the web browser’s window will open, and you will see the default application user interface as shown in Figure 3.3 (a). The default directory structure of this react app is also shown in Figure 3.3 (b).

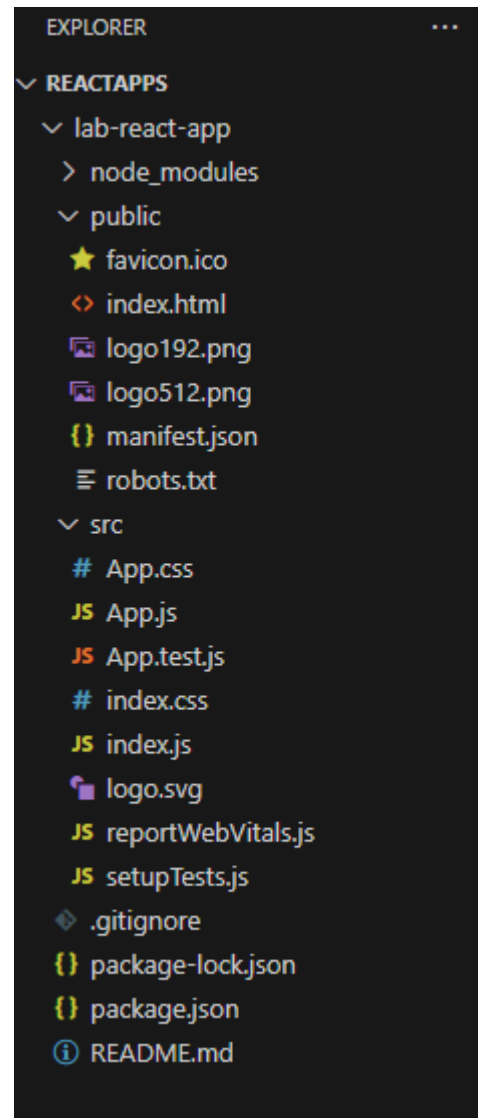
3.2 Components in React.js

To increase reusability of the code with certain level of flexibility, react offers component-based design of an application. Two types of components are available in react:

- Functional components
- Class-based components



a) Default output/ UI in web browser



b) Default directory structure

Figure 3.3: Default React application

3.2.1 Creating a functional component

A functional component can be defined as a Java Script function in multiple ways, given below:

- Using Named function
- Using Anonymous function
 - Unnamed function
 - Arrow function

with following restrictions:

- ✓ The name of the component function (or variable containing the function) must start with the capital letter and conventionally can follow Pascal Case format.
- ✓ The component function must return a value such as JSX code.
- ✓ If a component function is defined in a separate file as a module than the name of the file must start with capital letter and conventionally can follow Pascal Case format. This function (or variable containing the function) must be exported using “export” instruction to make it accessible from other files. If a function is only used in the same file where it is defined, then “export” instruction is not necessary.

Let’s create the React component named “ShowMessage” in a separate file by following the given steps:

1. Create a file named “ShowMessage.jsx” in “src” folder of the react application as shown in Figure 3.4
2. Add the component function in the file from either of the following syntax:
 - a. Figure 3.4 uses legacy Java Script function syntax
 - b. Figure 3.5 uses unnamed (anonymous) function syntax
 - c. Figure 3.6 uses arrow function (anonymous) syntax – from ECMA 2015
3. Export the component function (or variable containing the function) by adding export instruction as shown on line 5 in Figure 3.4, 3.5 and 3.6

Let’s use the React component named “ShowMessage” in the “App.js” by following the given steps:

1. Import component as shown on line 1 in Figure 3.7
2. Add component as shown on line 7 in Figure 3.7

Once you have modified the “App.js” as shown in Figure 3.7, the browser window will display the UI as shown in Figure 3.8.

3.3 Reusability of a component, introduces properties (props)

To make a component reusable, it must be able to render content dynamically. Hence the concept of properties is introduced. Two types of properties can be passed to component:

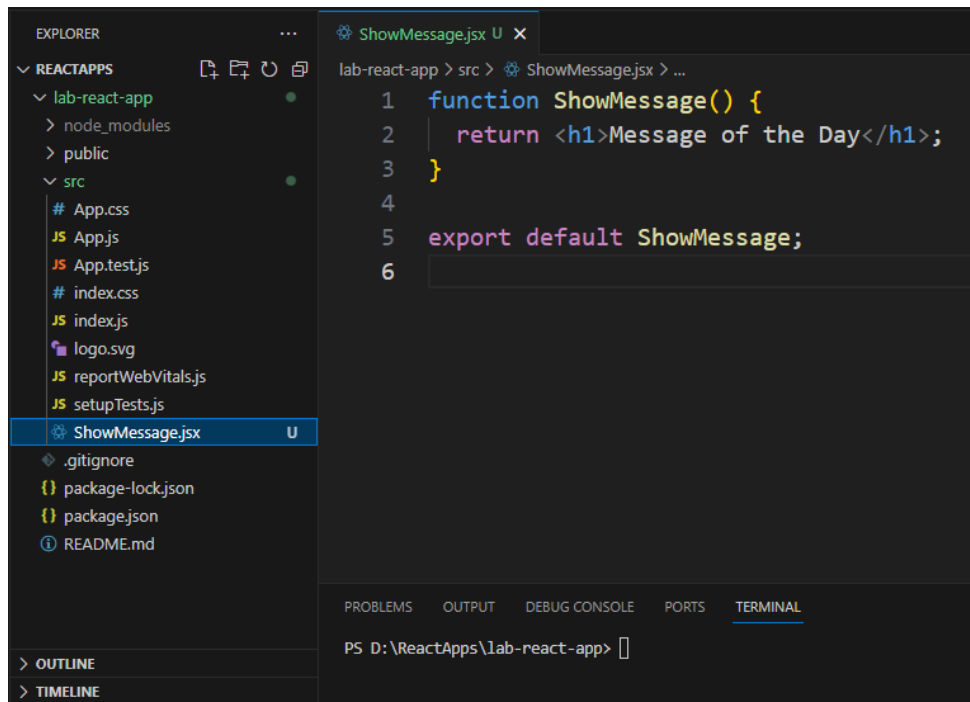
- via Attributes (i.e., attribute properties): It is also known as “passing props”,
- via Children (i.e., children property): It is also known as “[component composition](#)”.

Both above properties are bundled into an object which is passed an argument to the functional component (function of the component).

3.4 React DOM – The Virtual DOM

The DOM is the [document object model](#) which represents the user interface (UI) in memory. The HTML DOM is the DOM created by web browser from the *.html file so that the markup in *.html can be rendered in the form of web page.

A virtual DOM (VDOM) is another representation of the UI maintained in the memory which must be synched with the actual DOM so that only the changes in the virtual DOM can be reflected in the final UI.

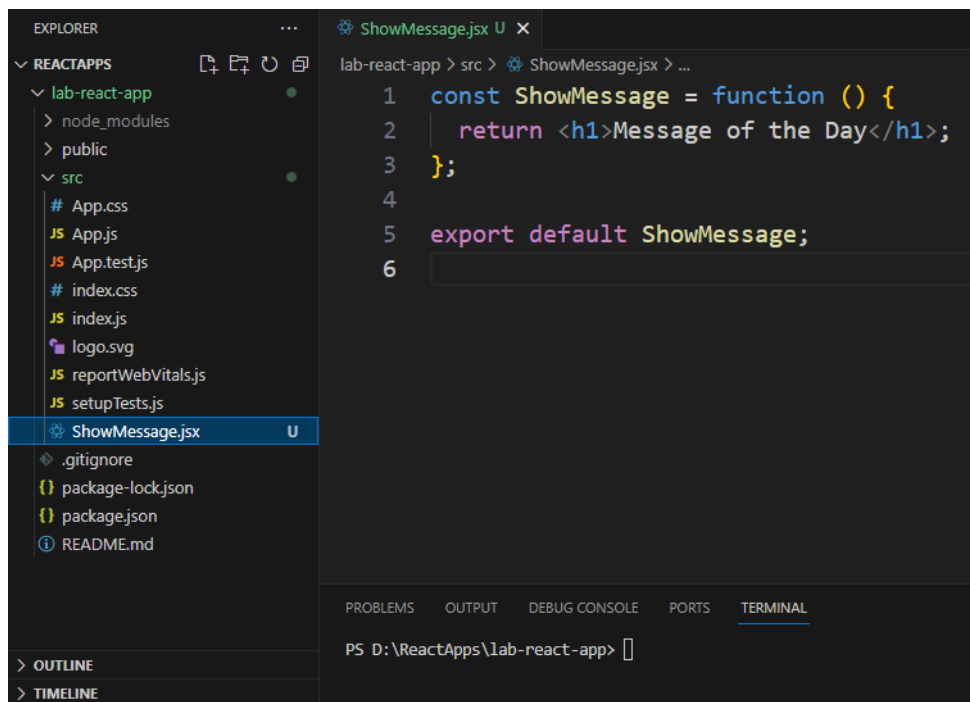


The screenshot shows the Visual Studio Code editor with a project named 'lab-react-app'. The Explorer sidebar on the left shows the file structure, with 'ShowMessage.jsx' selected. The main editor area displays the code for 'ShowMessage.jsx' with the following content:

```
1 function ShowMessage() {  
2   return <h1>Message of the Day</h1>;  
3 }  
4  
5 export default ShowMessage;  
6
```

The bottom of the editor shows the 'TERMINAL' tab with the command prompt 'PS D:\ReactApps\lab-react-app>'.

Figure 3.4: Named function – Functional component

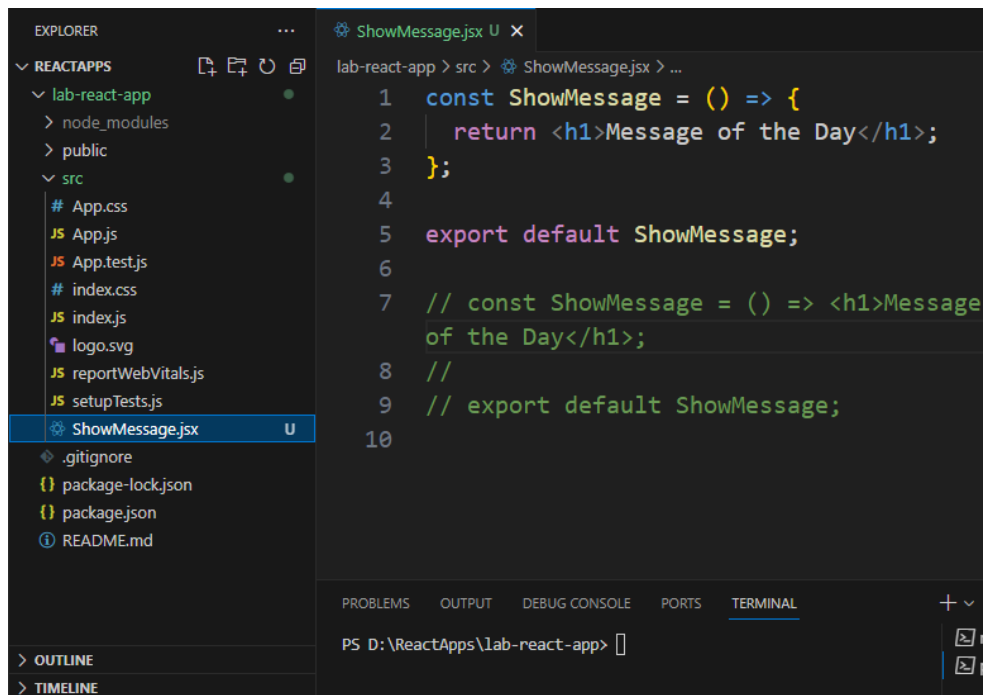


The screenshot shows the Visual Studio Code editor with the same project 'lab-react-app'. The Explorer sidebar on the left shows the file structure, with 'ShowMessage.jsx' selected. The main editor area displays the code for 'ShowMessage.jsx' with the following content:

```
1 const ShowMessage = function () {  
2   return <h1>Message of the Day</h1>;  
3 };  
4  
5 export default ShowMessage;  
6
```

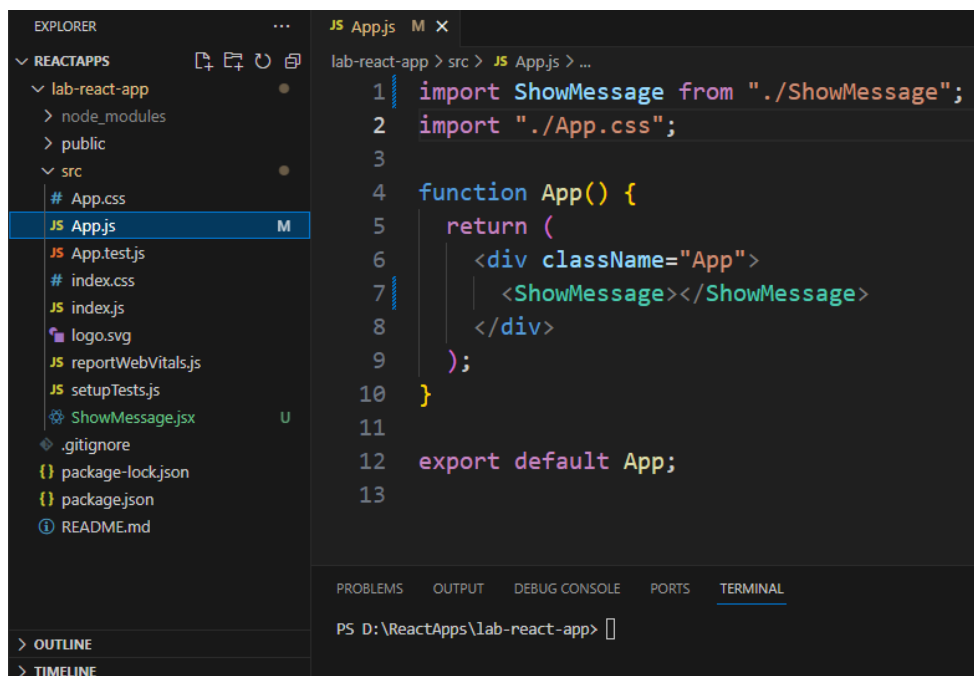
The bottom of the editor shows the 'TERMINAL' tab with the command prompt 'PS D:\ReactApps\lab-react-app>'.

Figure 3.5: Unnamed function – Functional component



```
1  const ShowMessage = () => {
2    return <h1>Message of the Day</h1>;
3  };
4
5  export default ShowMessage;
6
7  // const ShowMessage = () => <h1>Message
  of the Day</h1>;
8  //
9  // export default ShowMessage;
10
```

Figure 3.6: Arrow function – Functional component



```
1  import ShowMessage from "./ShowMessage";
2  import "./App.css";
3
4  function App() {
5    return (
6      <div className="App">
7        <ShowMessage></ShowMessage>
8      </div>
9    );
10 }
11
12 export default App;
13
```

Figure 3.7: Using/ Calling React component

React also uses the concept of Virtual DOM. React creates a virtual DOM ([ReactDOM](#)) and observes the change in it. Once it observes any change in virtual DOM, it compares virtual

DOM with the actual DOM in the browser. Finally, only the changes will be rendered instead of the complete page, if required. In this way, the changes will be reflected on web page faster as compared to re-rendering the full page.



Figure 3.8: Output UI in the web browser

4 Importing Bootstrap and Bootstrap-icons

Although it is not necessary to use bootstrap with React, this activity relies on bootstrap classes. Proceed with the following steps to introduce bootstrap to the react application.

Continuing with the react application developed so far in section 3.0, let's introduce Bootstrap in the react application with the following steps:

1. Installing Bootstrap in the react application "lab-react-app", give following instruction in terminal navigated to "D:\ReactApps\lab-react-app" as shown in Figure 4.1:

```
PS D:\ReactApps\lab-react-app> npm install bootstrap@5.3.2
```

Figure 4.1: Adding bootstrap to application's node modules

2. Import bootstrap's CSS to the application by adding "@import" CSS rule in "D:\ReactApps\lab-react-app\src\index.css", as shown in Figure 4.2. Before adding the import command remove all existing CSS rules of the default react app from the "index.css".

After completing the above steps in this section, you will observe font change in the UI.

Continuing further with the react application, let's introduce Bootstrap-icons in the react application with the following steps:

1. Installing Bootstrap-icons in the react application "lab-react-app", give following instruction in terminal navigated to "D:\ReactApps\lab-react-app" as shown in Figure 4.3:
2. Import bootstrap-icons' CSS to the application by adding "@import" CSS rule in "D:\ReactApps\lab-react-app\src\index.css", as shown in Figure 4.4.

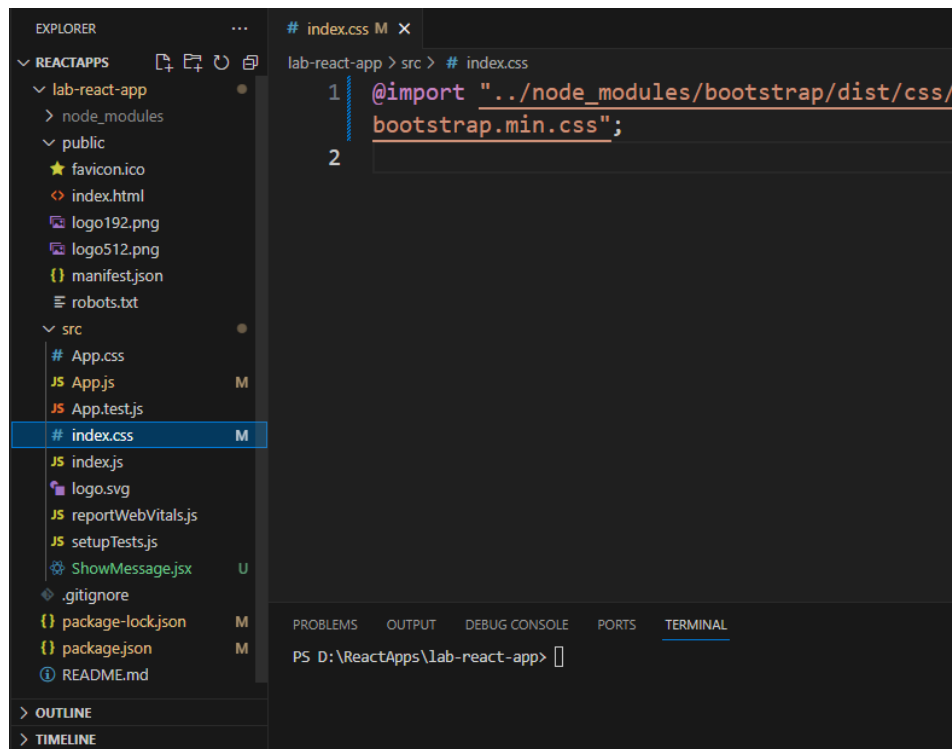


Figure 4.2: Adding bootstrap in the application

```
PS D:\ReactApps\lab-react-app> npm install bootstrap-icons
```

Figure 4.3: Adding bootstrap-icons to application's node modules

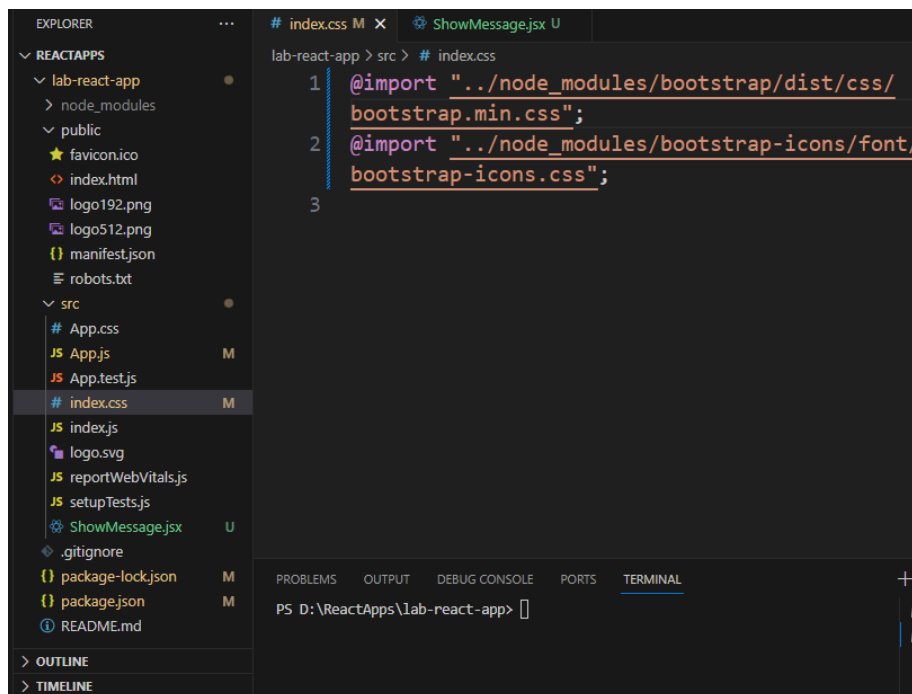


Figure 4.4: Adding bootstrap-icons in the application

5 Adding Inline Styling in JSX

In HTML (neither JSX nor React), the inline styling can be added to any HTML element by adding the style attribute which accepts string value. Therefore, a style attribute has a string value where this string is in the form of CSS property-value pairs separated by semicolon “;”. For instance, Figure 5.1 shows the style attribute which is passed to span HTML element to change its text color and font size.

```
<span style="font-size: 2rem; color: red">Inline Styling in HTML</span>
```

Figure 5.1: Adding inline styling to HTML Element

In JSX (and in React), the inline styling can be added to any JSX element by adding the style attribute which takes Java Script object value. Therefore, a style attribute has an object value having property-String value pairs separated by comma “,” and matches the CSS property-value pairs. The CSS property which has a dash “-” in its name must be used either as a string property. Alternatively, a camel case naming format can also be used for such properties.

For example, the custom style to change color of the text and font can be added to span JSX element as shown in Figure 5.2.

```
<span style={{fontSize:"2rem", color:"red"}}>Inline Styling in JSX</span>  
<span style={{font-size:"2rem", color:"red"}}>Inline Styling in JSX</span>
```

Figure 5.2: Adding inline styling to JSX Element

6 Adding Images in React

In HTML, an image can be added with the help of tag. The “src” attribute is used to add the local address of an image (such as ./images/abc.jpg). The address might be available in any local directory, or it can be remote as well.

Using assets from “public” folder: In React, the images can be added directly by using local path/address if the path is available from “public” folder of the react application. For instance, you have an image named “background.jpg” in “images” folder in “public” folder then the path of that image can be taken by using “images/ background.jpg” path as value to “src” attribute of an tag.

```

```

Figure 6.1: Adding image when “images” folder is placed in “public” folder – Path is relative to “public” folder

Using assets from “src” folder: In React, the images cannot be added directly by using local path/address if the path is available from any folder other than “public” (such as “src”

folder). In that case, you need to first import an image then you can use that import as a value to “src” attribute of an tag. For instance, you have an image named “background.jpg” in “images” folder in “src” folder then the image must be first imported using following command. Later you can use that import to use as a value to “src” attribute of an tag to add to that image.

```
import productImage from "../images/TV_OLED.jpg";

{/* Somewhere in the JSX code */}
<img width="100%" src={productImage} alt="Product Image" />
```

Figure 6.2: Adding image when “images” folder is not placed in “public” folder – Imported path is relative to current file

7 Activity

Use React to develop an application namely “online-purchase-portal” from the initial markup “index-markup.html” provided in section 10.1 (or use markup developed in Lab 04 which already includes Bootstrap classes in it) for payment portal of an online store. The application must show UI as shown in Figure 7.1. Develop an app by performing the following tasks which can be verified in your final code:

Task 01: Splitting the UI

Split the web application into eight components (including <App> component) which must be nested in the source code as shown in the following parent-child hierarchy:

- App
 - Header
 - MakeOrder
 - a. SelectedProduct
 - b. Summary
 - MakePayment
 - a. SelectedPayment
 - b. PaymentDetails

Task 02: Component Reusability

In this task, you will make the components reusable and dynamic.

To make the <SelectedPayment> component reusable and dynamic, use an approach of “[component composition](#)”, as mentioned in section 3.3. Therefore, pass the UI of four payment-methods to <SelectedPayment> component from <MakePayment> using children property of <SelectedPayment>.

To make rest of components reusable and dynamic, use an approach of “passing props”, as mentioned in section 3.3.

Task 03: Bootstrapping the React

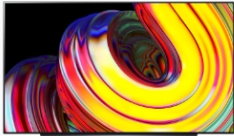
Add bootstrap as described in section 4 to produce GUI, shown in Figure 7.1.

Brain Drain

[*OPTIONAL*] Define local states in a component where needed.

Your Shopping Cart

Select delivery options



change

OLED TV 195 cm (77 Inch) [Model Year 2022]

Condition:

UsedBrand-new

Unit Price: € 2000/-

Quantity: 3

Order Summary

Change delivery address

Items Cost: € 6000
Delivery Insurance: € 35
Delivery Charges: Free for members

Total Purchase: € 6035
VAT: € 1146 (included)

Confirm payment details

Enter bank details:

BIC (Swift-Code):

IBAN:

Account Holder's Name:

☒ Save my bank details for future purchases?

Choose payment method

☐ Credit Card

☐ PayPal

☒ Personal Bank Account

☐ Bank Financing

Figure 7.1: UI with Bootstrap and React

8 External Links

Creating React App	https://create-react-app.dev/docs/getting-started
React Components	https://react.dev/reference/react/components
Building Custom Components	https://react.dev/learn/your-first-component
React Hooks	https://react.dev/reference/react/hooks
Babel	https://babeljs.io/

9 Rubric for Lab Assessment

The student performance for the assigned task during the lab session was:			
Excellent	The student completed assigned tasks without any help from the instructor and showed the results appropriately.	4	
Good	The student completed assigned tasks with minimal help from the instructor and showed the results appropriately.	3	
Average	The student could not complete all assigned tasks and showed partial results.	2	
Worst	The student did not complete assigned tasks.	1	

Instructor Signature: _____ Date: _____

10 Additional Resources

10.1 Initial markup file

index-markup.html

```
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <title>Payment</title>
5   </head>
6   <body>
7     <header>
8       <span>Your Shopping Cart</span>
9     </header>
10    <main>
11      <section>
12        <article>
13          <div>
14            <h2>Select delivery options</h2>
15            <div>
16              <div>
17                <div>
18                  
23                <div>
24                  <a href="#change">change</a>
25                </div>
26              </div>
27            <div>
28              <h4>OLED TV 195 cm (77 Inch) [Model Year 2022]</h4>
29              <span>Condition:</span>
30              <div>
31                <button>Used</button>
32                <button>Brand-new</button>
33              </div>
34
35              <div>Unit Price: 2000/-</div>
36
37              <label for="qty"
38                >Quantity:
39              <select name="quantity" id="qty">
```

```

40         <option value="1">1</option>
41         <option value="2">2</option>
42         <option value="3">3</option>
43         <option value="4">4</option>
44         <option value="5">5</option>
45     </select>
46     </label>
47 </div>
48 </div>
49 </div>
50 </div>
51 <div>
52     <h2>Order Summary</h2>
53     <div>
54         <div>
55             <div>
56                 <a href="#" role="button">Change delivery address</a>
57             </div>
58             <div>
59                 <span>Items Cost:</span>
60                 <span>6000</span>
61             </div>
62             <div>
63                 <span>Delivery Insurance:</span>
64                 <span>35</span>
65             </div>
66             <div>
67                 <span>Delivery Charges:</span>
68                 <span>Free for members</span>
69             </div>
70             <span>
71                 <span>Total Purchase:</span>
72                 <span>6035</span>
73             </span>
74             <div>
75                 <span>VAT:</span>
76                 <span>1146 <a href="#VAT">(included)</a></span>
77             </div>
78         </div>
79     </div>
80 </div>
81 </article>
82 <div></div>
83 <article>
84     <div>

```



```

85     <h2>Confirm payment details</h2>
86     <div>
87         <h4>You bank details:</h4>
88         <form action="abc">
89             <div>
90                 <label for="bic">BIC (Swift-Code):</label>
91                 <input type="text" id="bic" /> <label for="iban">IBAN:</label>
92                 <input type="text" id="iban" />
93                 <label for="accholder">Account Holder's Name:</label>
94                 <input type="text" id="accholder" />
95             </div>
96             <div>
97                 <input type="checkbox" id="savechk" />
98                 <label for="savechk">
99                     >Save my bank details for future purchases?</label>
100             </div>
101         </form>
102     </div>
103 </div>
104 <div>
105     <h2>Choose payment method</h2>
106     <div>
107         <form action="xyz">
108             <div>
109                 <label for="cc">
110                     ><input name="paymentmethod" type="radio" id="cc" /> Credit
111                     Card</label>
112                 >
113                 <label for="pp">
114                     ><input name="paymentmethod" type="radio" id="pp" />
115                     PayPal
116                 </label>
117                 <label for="ba">
118                     ><input name="paymentmethod" type="radio" id="ba" checked />
119                     <span> Personal Bank Account</span>
120                 </label>
121                 <label for="bf">
122                     ><input name="paymentmethod" type="radio" id="bf" />
123                     Bank Financing
124                 </label>
125             </div>
126         </form>
127     </div>
128 </div>
129 </div>

```

```
130     </article>
131     <article></article>
132 </section>
133 </main>
134 </body>
135 </html>
136
137
```

10.2 Content image

TV_OLED.jpg

