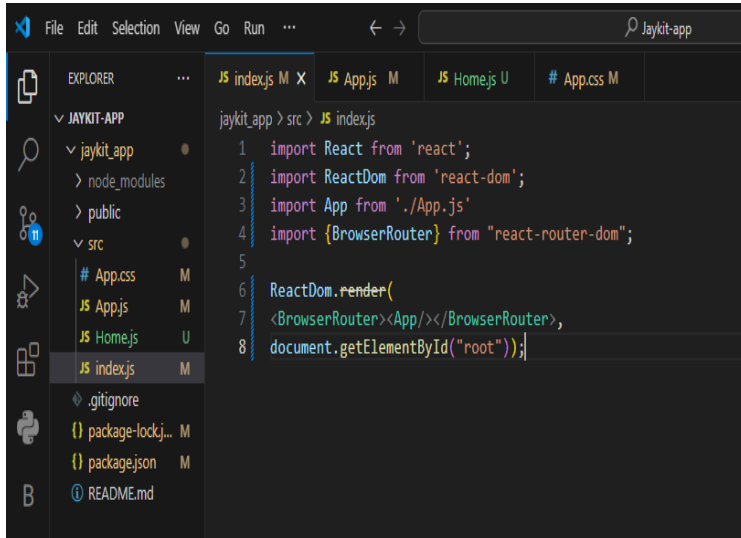


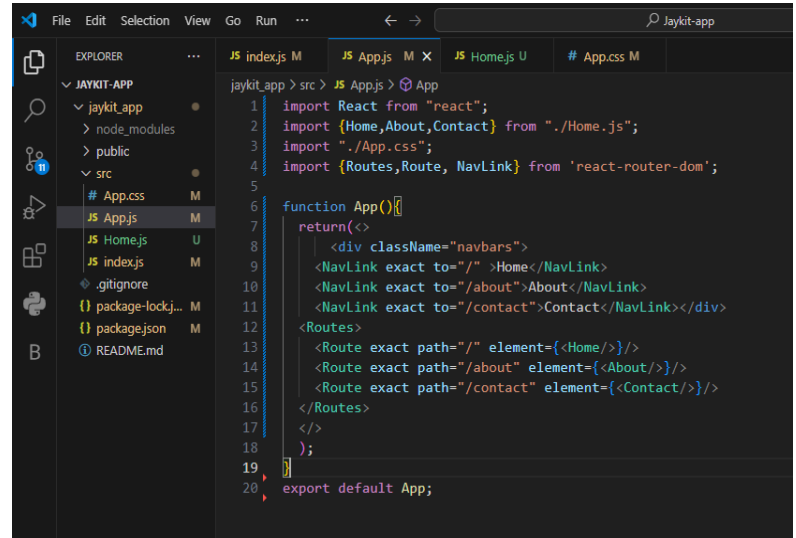
Experiment no:09

1)index.js



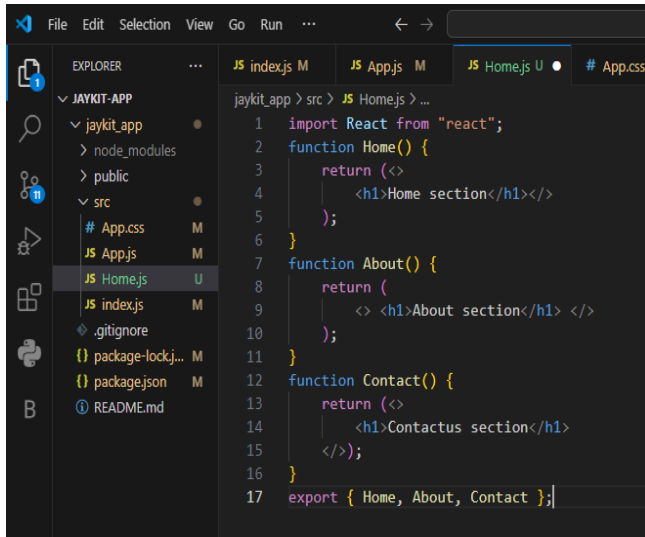
```
1 import React from 'react';
2 import ReactDOM from 'react-dom';
3 import App from './App.js'
4 import {BrowserRouter} from "react-router-dom";
5
6 ReactDOM.render(
7   <BrowserRouter><App/></BrowserRouter>,
8   document.getElementById("root"));
```

2)App.js



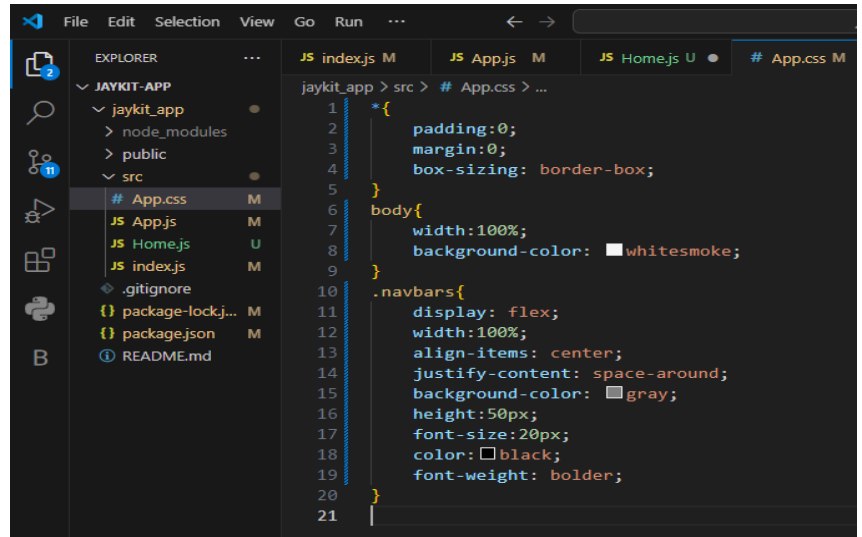
```
1 import React from "react";
2 import {Home,About,Contact} from "./Home.js";
3 import "./App.css";
4 import {Routes,Route, NavLink} from 'react-router-dom';
5
6 function App(){}
7 return(<>
8   <div className="navbars">
9     <NavLink exact to="/" >Home</NavLink>
10    <NavLink exact to="/about">About</NavLink>
11    <NavLink exact to="/contact">Contact</NavLink></div>
12    <Routes>
13      <Route exact path="/" element={<Home/>}>
14      <Route exact path="/about" element={<About/>}>
15      <Route exact path="/contact" element={<Contact/>}>
16    </Routes>
17  </>
18 );
19
20 export default App;
```

3)Home.js



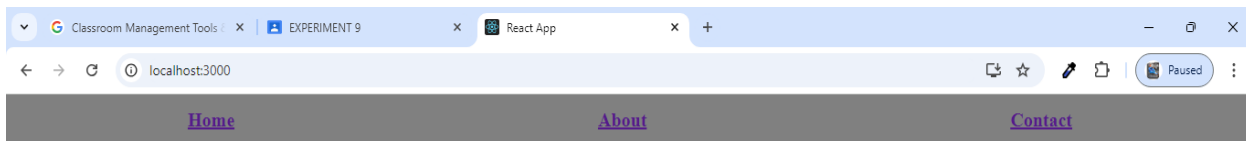
```
1 import React from "react";
2 function Home() {
3   return (<>
4     <h1>Home section</h1></>
5   );
6 }
7 function About() {
8   return (
9     <> <h1>About section</h1> </>
10  );
11 }
12 function Contact() {
13   return (<>
14     <h1>Contactus section</h1>
15   </>);
16 }
17 export { Home, About, Contact };
```

4)App.css

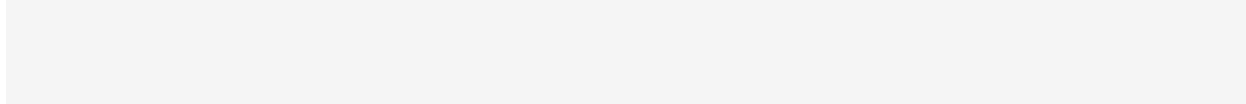


```
1 *{
2   padding:0;
3   margin:0;
4   box-sizing: border-box;
5 }
6 body{
7   width:100%;
8   background-color: whitesmoke;
9 }
10 .navbars{
11   display: flex;
12   width:100%;
13   align-items: center;
14   justify-content: space-around;
15   background-color: gray;
16   height:50px;
17   font-size:20px;
18   color: black;
19   font-weight: bold;
20 }
21
```

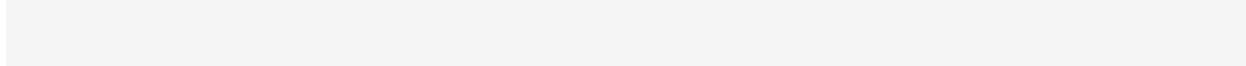
Output:



Home section



About section



Experiment 08

Aim: Write a program to demonstrate Form Handling in React js.

Theory:

React Forms

Forms are an integral part of any modern web application. It allows the users to interact with the application as well as gather information from the users. Forms can perform many tasks that depend on the nature of your business requirements and logic such as authentication of the user, adding user, searching, filtering, booking, ordering, etc. A form can contain text fields, buttons, checkbox, radio button, etc.

Creating Form

React offers a stateful, reactive approach to build a form. The component rather than the DOM usually handles the React form. In React, the form is usually implemented by using controlled components.

There are mainly two types of form input in React.

1. Uncontrolled component
2. Controlled component

Uncontrolled component

The uncontrolled input is similar to the traditional HTML form inputs. The DOM itself handles the form data. Here, the HTML elements maintain their own state that will be updated when the input value changes. To write an uncontrolled component, you need to use a ref to get form values from the DOM. In other words, there is no need to write an event handler for every state update. You can use a ref to access the input field value of the form from the DOM.

Controlled Component

In HTML, form elements typically maintain their own state and update it according to the user input. In the controlled component, the input form element is handled by the component rather than the DOM. Here, the mutable state is kept in the state property and will be updated only with **setState()** method.

Controlled components have functions that govern the data passing into them on every **onChange event**, rather than grabbing the data only once, e.g., when you click a **submit button**. This data is then saved to state and updated with **setState()** method. This makes component have better control over the form elements and data.

A controlled component takes its current value through **props** and notifies the changes through **callbacks** like an onChange event. A parent component "controls" this changes by handling the callback and managing its own state and then passing the new values as props to the controlled component. It is also called as a "dumb component."

Conclusion: Thus, In this experiment we have learned and implemented about Forms in React js.

CODE:

App.js

```
import './App.css';
import { Form, Welcome } from './components';

function App() {
  return (
    <div className="App">
      <Form/>
    </div>
  );
}

export default App;
```

Form.jsx

```
import React from 'react'
import { useState } from 'react'
import './form.css'

const Form = () => {
  const [name, setName] = useState("");
  const [age, setAge] = useState("");
  const [phone, setPhone] = useState("");
  const [email, setEmail] = useState("");

  const submit=(event)=>{
    event.preventDefault();
    alert(` ${name}, ${age}`)
  }

  return (

    <section className='section'>

      <div className='form'>
        <div className="elements name">
          <span>Name </span>
          <input type="text" value={name}
onChange={{(e)=>{setName(e.target.value)}} placeholder='Enter Your name' />
        </div>
        <div className="elements age">
          <span>Age </span>
          <input type="number" value={age}
onChange={{(e)=>{setAge(e.target.value)}} placeholder='Enter Your age' />
        </div>
      </div>
    </section>
  )
}
```

```

    <div className="elements phone_no">
      <span>Phone Number </span>
      <input type="text" value={phone}
onChange={(e)=>{setPhone(e.target.value)}} placeholder='Enter Your Phone
Number' />
    </div>
    <div className="elements email">
      <span>Email </span>
      <input type="text" value={email}
onChange={(e)=>{setEmail(e.target.value)}} placeholder='Enter Your email' />
    </div>
    <button onClick={submit}>Submit</button>
  </div>
</section>
)
}

export default Form

```

Form.css

```

.section{
  display: flex;
  justify-content: center;
  align-item: center;
}
.form{
  display: flex;
  flex-direction: column;
  background-color: #f7f4ff;
  border-radius: 20px;
  padding: 43px 28px;
}

.elements{
  margin-top: 21px;
}

.elements span{
  margin-right: 9px;
  font-weight: 600;
}

input{
  padding: 4px 9px;
  border-radius: 20px;
  border: 1px solid #cbc9c9;
}

```

```
button{  
  width: 90px;  
  height: 28px;  
  margin: 0 auto;  
  padding: 3px 0;  
  margin-top: 10px;  
  border-radius:8px;  
  border: 1px solid #cbc9c9;  
}
```

OUTPUT: