Name – Mohit Shadija
Div – D15B
Roll - 54

IP Lab
Exp 8

Aim: Experiment to study the basics of React.

Theory:

Functional component:

This function is a valid React component because it accepts a single "props" (which stands for properties) object argument with data and returns a React element. We call such components "function components" because they are literally JavaScript functions.

eg:-

```
class Welcome extends React.Component {
  render() {
    return <h1>Hello, {this.props.name}</h1>;
  }
}
```

Class component:

React class based components are the bread and butter of most modern web apps built in ReactJS. These components are simple classes (made up of multiple functions that add functionality to the application). All class based components are child classes for the Component class of ReactJS

eg:-

```
import React from "react";

class App extends React.Component {
  render() {
    return <h1>GeeksForGeeks</h1>;
  }
}

export default App;
```

React Features

Currently, ReactJS gaining quick popularity as the best JavaScript framework among web developers. It is playing an essential role in the front-end ecosystem. The important features of ReactJS are as following.

- JSX
- Components
- One-way Data Binding
- Virtual DOM
- Simplicity
- Performance

React Props

Props stand for "Properties." They are read-only components. It is an object which stores the value of attributes of a tag and work similar to the HTML attributes. It gives a way to pass data from one component to other components. It is similar to function arguments. Props are passed to the component in the same way as arguments passed in a function.

Props are immutable so we cannot modify the props from inside the component. Inside the components, we can add attributes called props. These attributes are available in the component as this.props and can be used to render dynamic data in our render method.

eg: -

```
import React, { Component } from 'react';
class App extends React.Component {
  render() {
    return (
       <div>
         <h1> Welcome to the { this.props.name } </h1>
              <p> <h4> Here you will get to know so many things and your knowledge will be
enhanced . </h4> </p>
       </div>
    );
  }
}
export default App;
```

React State

The state is an instance of React Component Class can be defined as an object of a set of observable properties that control the behavior of the component. In other words, the State of a component is an object that holds some information that may change over the lifetime of the component. For example, let us think of the clock that we created in this article, we were calling the render() method every second explicitly, but React provides a better way to achieve the same result and that is by using State, storing the value of time as a member of the component's state. We will look into this more elaborately later in the article.

eg:-

```
class Greetings extends React.Component {
  state = {
    name: "World"
  };
  updateName() {
    this.setState({ name: "Simplilearn" });
  }
  render() {
      return(
          <div>
              {this.state.name}
          </div>
      )
  }
}
```

Code:

Router:

```
function App() {

  retur
  n (
                <div className='ui container'>
                  <Router>
                    <Header />
                    <div class="ui two column centered grid">
                      <div class="column">
                        <Switch>
                          <Route path='/' element={<ContactList contacts={contacts}
             getContactId={removeContactHandler} />} />
                          <Route path='/add' element={<AddContact
             addContactHandler={addContactHandler} />} />
                          <Route path='/contact/:contactId' element={<ContactDetail />} />
                        </Switch>
                      </div>
                    </div>
```

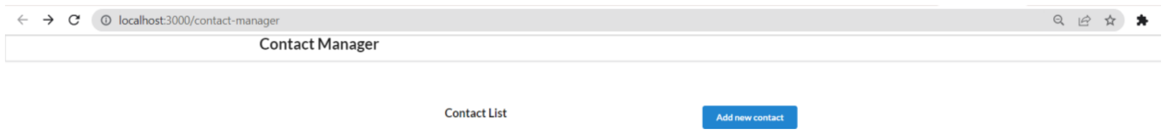
```
          </Router>
         </div>
        );
       }
```

Contact List component:

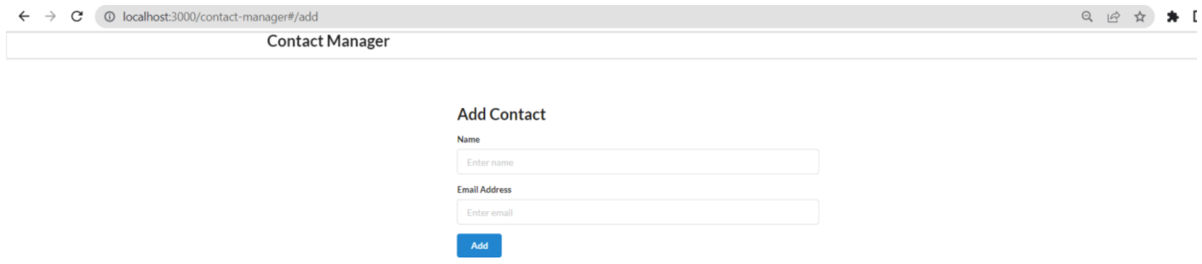
```
import React from "react";
import { Link } from 'react-router-dom';
import ContactCard from "./ContactCard";


const ContactList = ({contacts, getContactId}) => {

  const deleteContactHandler = (id) => {
    getContactId(id);
  };

  const renderContactList = contacts.map(contact => {
    return (
      <ContactCard key={contact.id} contact={contact} clickHandler={deleteContactHandler}
/>
    )
  });

  return (
    <div className="ui big celled list" style={{ marginTop:'70px', }}>
      <h3 className="header">
        Contact List
        <Link to='/add'>
          <button className="ui button blue right floated">Add new contact</button>
        </Link>
      </h3>

      {renderContactList}
    </div>
  );
};

export default ContactList;
```

Output:

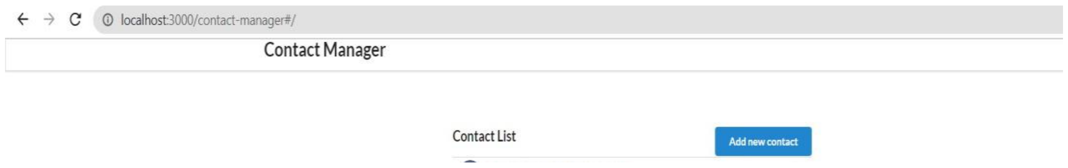
Path: /contact-manager#/



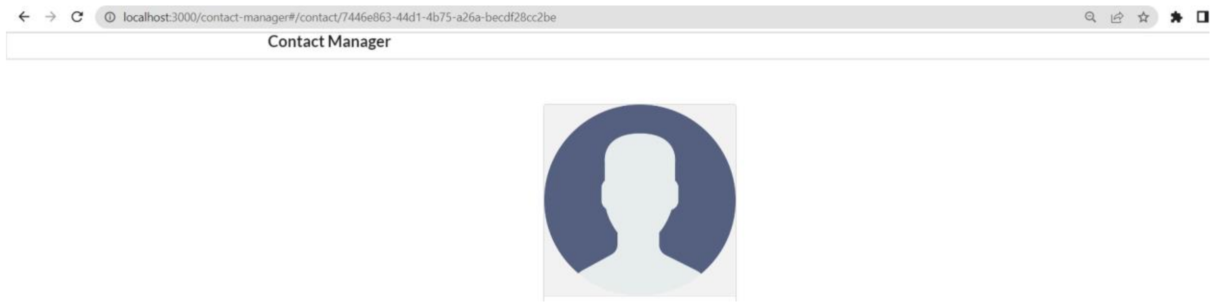
New contacts can be added to the contact list. Click on add new contact.



Enter credentials and add new contact and click on add.



To view contact details click on the link in the contacts list.


localhost:3000/contact-manager#/contact/7446e863-44d1-4b75-a26a-becdf28cc2be
Contact Manager

Conclusion:

Created contact management system in react js.