MODULE: 9 ReactJs Intro

1. What is React Js?

A.React is a framework that employs Webpack to automatically compile React, JSX, and ES6 code while handling CSS file prefixes. React is a JavaScript-based UI development library. Although React is a library rather than a language, it is widely used in web development. The library first appeared in May 2013 and is now one of the most commonly used frontend libraries for web development.

2.What is NPM in React Js?

A.NPM, the Node package manager is used for managing and sharing the packages for either React or Angular. NPM will be installed along with Nodejs. Node. js can be downloaded and installed from the official NodeJs website.

3.What is Role of Node Js in react Js?

A.Reasons to Use Node JS and React JS Together
High Server Load: Using Node JS with React can help in handling requests and maintaining server load balance. JSON APIs: You can easily build JSON APIs for your application using Node JS. The reusability of the code enables sharing within React JS

4. What is CLI command In React Js?

A.React has its own command-line interface (CLI) commands. However, these CLI commands are currently only used to create a passable version of a react application using the command line.

5.What is Components in React Js?

A.Components are independent and reusable bits of code. They serve the same purpose as JavaScript functions, but work in isolation and return HTML. Components come in two types, Class components and Function components.

6.What is Header and Content Components in React Js?

A.Headers are compositions that extend standard navbar functionalities. They contain additional components like a jumbotron, sub-navbar, or image covers which serve as a containers for extra navigation elements - usually links, forms, or call-to-action buttons.

7.How to install React Js on Windows, Linux Operating System? How to Install NPM and How to check version of NPM?

A. How To Install React

Each operating system has slightly different requirements when installing React. Follow along with the appropriate set of instructions below for the operating system you use.

How To Install React on Windows
In this section, we'll guide you through the process of installing React on a Windows machine.

Follow these steps to get started:

Step 1: Install Node.js and npm
Step 2: Install Create React App
Step 3: Create a New React Project
Step 4: Go To the Project Directory and Start the Development Server

Step 1: Install Node.js and npm
Before installing React, you need to have Node.js and npm (Node Package Manager) installed on your system. If you haven't already installed them, follow these steps:

Visit the Node.js download page at: https://nodejs.org/en/download/
Download the installer for your Windows system (either the LTS or Current version is fine, but the LTS version is recommended for most users)
To install Node.js and npm, please run the installer and carefully follow the provided prompts.

After the installation is complete, you can verify that Node.js and npm are installed by opening a command prompt and running the following commands:
Check the NPM version
Npm -v

How To Install React on Linux
If you have a Linux-based system, you'll want to follow this set of steps to install React:

Step 1: Install npm
Step 2: Install create-react-app Utility
Step 3: Create and Launch Your First React Application
Step 1: Install npm
Login to your server as a sudo user and run the following command:

sudo apt install npm
Once the installation is complete, verify the version of npm installed using the command:

npm --version

8.How to check version of React Js?

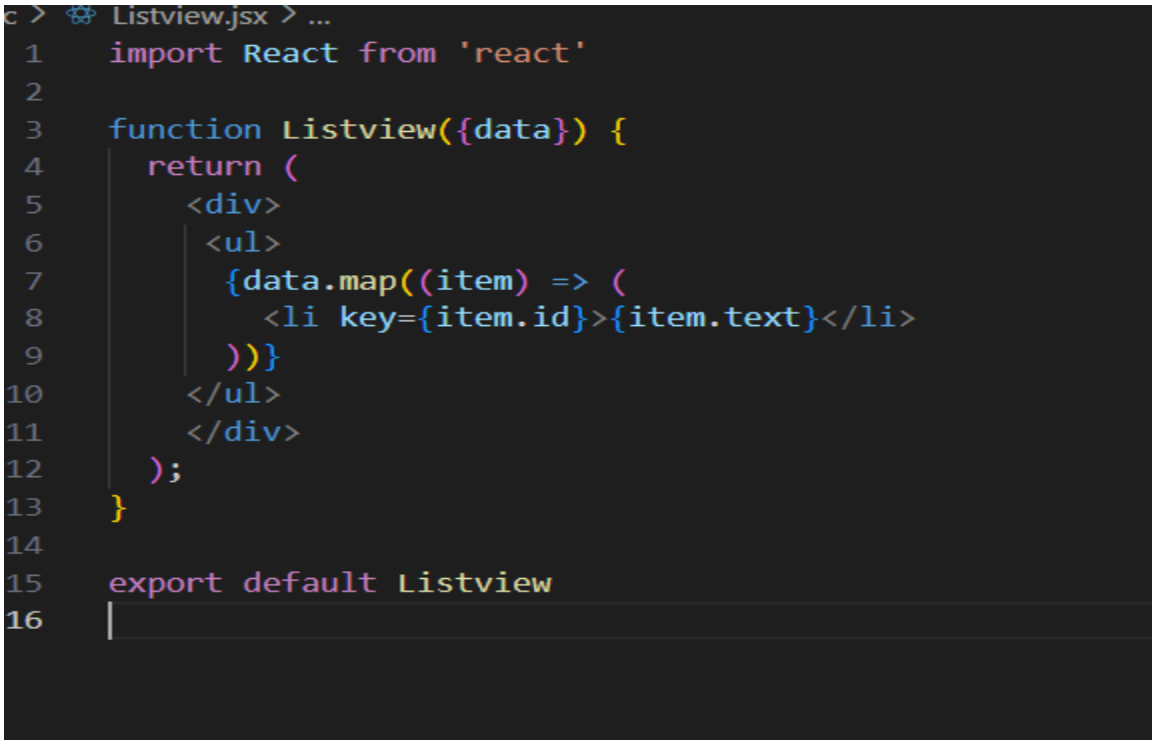
A.To check which React version is your project using you need to open the package. json. Take a look under the dependencies section. It should list all of the dependencies of your project and one of those should be React.

9.How to change in components of React Js?

A.setState() setState() enqueues changes to the component state and tells React that this component and its children need to be re-rendered with the updated state. This is the primary method you use to update the user interface in response to event handlers and server responses.

10.How to Create a List View in React Js?

A.



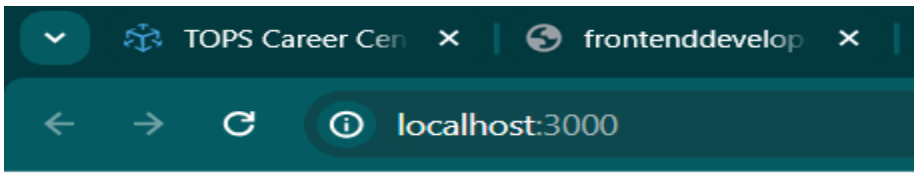
src > JS App.js > App

```jsx
import React from "react";
import Listview from "./Listview";

function App() {
    const dataList = [
        { id: 1, text: 'Item 1' },
        { id: 2, text: 'Item 2' },
        { id: 3, text: 'Item 3' },
        { id: 4, text: 'Item 4' },
      ];
  return (
    <div>
      <h1>List View Example</h1>
     <Listview data={dataList} />
    </div>
  );
}

export default App;

```



List View Example

- Item 1
- Item 2
- Item 3
- Item 4

11.Create Increment decrement state change by button click?

A.

IncDinc.jsx > IncDinc

```jsx
import { useState } from "react";

function IncDinc() {
  const [count, setCount] = useState(0);

  const handleIncrement = () => {
    setCount(count + 1);
  };

  const handleDecrement = () => {
    if (count > 0) {
        setCount(count - 1);
      }
  };
  return (
    <div>
      <h2>Counter: {count}</h2>
      <button onClick={handleDecrement}>Decrement</button>
      <button onClick={handleIncrement}>Increment</button>
    </div>
  );
}

export default IncDinc;
```

Increment and Decrement Example

Counter: 5

Decrement | Increment

MODULE: 10 List and Hooks

12. Explain Life cycle in Class Component and functional component with Hooks.

A. Function

```jsx
import React, { useState, useEffect } from 'react'
import Imgfunli from './Imgfunli';

function Funlifecyc() {
  const [name, setName] = useState("Rajesh Nagar"); // single

  // mutltiple property object
  const [data, setData] = useState({
    email: "raj@gmail.com",
    age: 32,
    number: 1,
    isImage: false
  })

  useEffect(()=>{
    console.log(' component DidMount/Update '); // component load
  },[data.number]);


  return (
    <div>
   <button onClick={() => setName('Akash Nagar')}>change</button>
      <h1>{name}</h1>

      <hr />
      <button onClick={() => { setData({ ...data, email: "rajeshnagar@gmail.com", age: 34 }) }}>change</button>
      <h1>Hi my email is : {data.email} and my age is : {data.age}</h1>
      <hr />

      <button onClick={() => setData({ ...data, number: data.number + 1 })}>+</button>
      <h1>{data.number}</h1>
      <button onClick={() => setData({ ...data, number: data.number - 1 })}>+</button>

      <hr />

      <button onClick={() => setData({ ...data, isImage: false })}>Hide</button>
      <button onClick={() => setData({ ...data, isImage: true })}>Show</button>
```

The useEffect Hook allows you to perform side effects in your components.

Some examples of side effects are: fetching data, directly updating the DOM, and timers.

useEffect accepts two arguments. The second argument is optional.
useEffect(<function>, <dependency>)
Life Cycle in Func Compo
new hooks introduced useEffect() work as three life cycle

Class.

```
import React, { setState ,Component } from 'react'
import Imgclass from './Imgclass';
export default class Classlife extends Component {

    constructor() {
        super();
        this.state = {
            name: "Rajesh nagar",
            email: "raj@gmail.com",
            age: 32,
            number: 1,
            isImage: false
        }
    }

    //autocall when any update in component
    componentDidUpdate() {
        console.log(' component DidUpdate / update');
    }

  render() {
    return (
      <div>
        <div>
                <button onClick={() => this.setState({ name: "Akash Nagar" })}>change</button>
                <h1>{this.state.name}</h1>

                <hr />
                <button onClick={() => { this.setState({ email: "rajeshnagar@gmail.com", age: 34 }) }}>change</
                <h1>Hi my email is : {this.state.email} and my age is : {this.state.age}</h1>

                <hr />

                <button onClick={() => this.setState({ number: this.state.number + 1 })}>+</button>
                <h1>{this.state.number}</h1>
                <button onClick={() => this.setState({ number: this.state.number - 1 })}>+</button>
```

componentDidMount(){} // birth// run when component ready to use

=>Means Component Call means mount / use
componentDidUpdate(){} // marriage// run when any update in compo

=>Means Component setState means state update or Update
componentWillUnmount(){}// end // run when component remove

13.Create React app with modules and lazy loading (Admin-user module with child Router and outlet)

Adminhome.jsx > ...

```jsx
import React from 'react'

function Adminhome() {
  return (
    <div>
      <h2>Admin Home</h2>
      {/* Your admin home component content */}
    </div>
  )
}

export default Adminhome

```

```jsx
import React from 'react'

function Adminpro() {
  return (
    <div>
      <h2>Admin Profile</h2>
      {/* Your admin profile component content */}
    </div>
  )
}

export default Adminpro
```

Admin.jsx > Admin

```jsx
import React from "react";

import { Route, Link, Switch } from 'react-router-dom';

// Lazy load child components
const Adminhome = React.lazy(() => import('./Adminhome'));
const Adminpro = React.lazy(() => import('./Adminpro'));

function Admin() {
  return (
    <div>
    <h1>Admin Dashboard</h1>
      <nav>
        <ul>
          <li>
            <Link to="admin/home">Home</Link>
          </li>
          <li>
            <Link to="/admin/profile">Profile</Link>
          </li>
        </ul>
      </nav>
      <React.Suspense fallback={<div>Loading...</div>}>
        < Switch>
          <Route path="/admin/home" component={Adminhome} />
          <Route path="/admin/profile" component={Adminpro} />
        </ Switch>
      </React.Suspense>
    </div>
  );
}

export default Admin;
```

App.js > App

```jsx
import React from "react";
import { BrowserRouter as Router, Route, Switch } from "react-router-dom";
import Admin from "./Admin";


function App() {
  return (
    <>
      <Router>
        <div>
          <Switch>
            {/* Other routes */}
            <Route path="/admin" component={Admin} />
          </Switch>
        </div>
      </Router>
    </>
  );
}

export default App;

```