Lab14 : LifeCycle Hooks for Class Component

Lab14: Files Required:

1. index.html	2. index.js
3. index.css	4. App.js
5. MyCourses.js	6. Hello.js
7. Hai.js	

1. index.html

2. index.js

```
import React from 'react';
import ReactDOM from 'react-dom';
import 'bootstrap/dist/css/bootstrap.min.css';
import './index.css';
import App from './App';

ReactDOM.render(<App />, document.getElementById('myroot'));
```

3. index.css

4. App.js

```
import React, { Component } from 'react';

import 'bootstrap/dist/css/bootstrap.min.css';
import MyCourses from './MyCourses';
import Hello from './Hello';

class App extends Component {
 static displayName = "MyAppComponent";

 constructor(props){
  super(props);
  console.log("[App]-M1-Constrcutor");
   this.state = {
     companyName:'CourseCube',
     courseFlag:true,
   };
 }

 static getDerivedStateFromProps(myprops,mystate){
   console.log("[App]-M2-getDerivedStateFromProps()");
   //console.log("props - ",myprops);
   //console.log("state - ",mystate);
   return null;
 }
```

```
componentDidMount(){
  console.log("[App]-M4-componentDidMount()");
}

shouldComponentUpdate(nextProps, nextState) {
  console.log("[App] - shouldComponentUpdate");
  return true;
}

getSnapshotBeforeUpdate(prevProps, prevState) {
  console.log("[App] - getSnapshotBeforeUpdate");
  return null;
}

componentDidUpdate(prevProps, prevState, mysnapshot) {
  console.log("[App] - componentDidUpdate");
  //console.log("[App] -", mysnapshot);
}

componentWillUnmount() {
  console.log("[App] - componentWillUnmount");
}

static getDerivedStateFromError(error) {
  console.log("[App] - getDerivedStateFromError", error);
  return { hasError: true };
}

componentDidCatch(error, info) {
  console.log("[App] - componentDidCatch", error);
  console.log("[App] - componentDidCatch", info.componentStack);
}

// componentWillMount() {
//   console.log("[App] - componentWillMount");
// }
// componentWillReceiveProps() {
//   console.log("[App] - componentWillReceiveProps");
// }
// componentWillUpdate() {
//   console.log("[App] - componentWillUpdate");
// }
```

```
  showHideCourses = () => {
   let mycourseFlag= this.state.courseFlag;
   this.setState({
    courseFlag : ! mycourseFlag
   });
 }
  render() {
   console.log("[App]-M3-render()");

   let displayCourse = null;
   if(this.state.courseFlag===true){
    displayCourse = (
    <MyCourses
      hello="Hello Guys"
      hai="Hai Guys"/>
    );
   }
  return (
   <div className="container">
   <h1 className="text-center"> Welcome to {this.state.companyName}!!!</h1>
   <br/>

   <button className="btn btn-danger"
       onClick={this.showHideCourses} >
       Show Hide Courses
     </button>
   <br/>   <br/>
    {displayCourse}
    <br/>
    <table className="table bordered">
     <tbody>
     <tr>
      <td>   <Hello /> </td>
      <td>  <Hello mytrainer="Dande"/>  </td>
      <td>  <Hello mytrainer="Srinivas Dande" mycolor="Blue"/> </td>
     </tr>
     </tbody>
    </table>
   </div>
   );
 }
}

export default App;
```

5. MyCourses.js

```
import React, { Component } from 'react';

class MyCourses extends Component {

 static displayName = "MyCoursesComponent";

 constructor(props){
  super(props);

  console.log("[MyCourses]-Constrcutor");

  this.state = {
   trainerName:"Srinivas Dande",
   trainerEmail:"sri@jlc.com",
   mycourseList : [
    {cid:101,cname:"Angular 7",price:15000},
    {cid:102,cname:"React 16",price:15000},
    {cid:103,cname:"Java FSD",price:35000},
    {cid:104,cname:"MicroServices",price:25000},
    {cid:105,cname:"DevOps",price:15000},
   ]
  };
 }
  static getDerivedStateFromProps(myprops,mystate){
   console.log("[MyCourses]- getDerivedStateFromProps()");
  // console.log("props - ",myprops);
  // console.log("state - ",mystate);
   return null;
  }
  componentDidMount(){
   console.log("[MyCourses]-componentDidMount()");
  }
   shouldComponentUpdate(nextProps, nextState) {
   console.log("[MyCourses] - shouldComponentUpdate");
   return true;
  }
  getSnapshotBeforeUpdate(prevProps, prevState) {
   console.log("[MyCourses] - getSnapshotBeforeUpdate");
   let mysnapshot = {
    mymessage: "I am a SnapShot",
    trainerName: prevState.trainerName,
    trainerEmail: prevState.trainerName,
   };
   return mysnapshot;
  }
 componentDidUpdate(prevProps, prevState, mysnapshot) {
  console.log("[MyCourses] -componentDidUpdate");
 // console.log("[MyCourses] -", mysnapshot);
}


componentWillUnmount() {
  console.log("[MyCourses] - componentWillUnmount");
}

static getDerivedStateFromError(error) {
  console.log("[MyCourses] - getDerivedStateFromError", error);
  return { hasError: true };
}

componentDidCatch(error, info) {
  console.log("[MyCourses] - componentDidCatch", error);
  console.log("[MyCourses] - componentDidCatch", info.componentStack);
}

// UNSAFE_componentWillMount() {
//   console.log("[MyCourses] - UNSAFE_componentWillMount");
// }
// UNSAFE_componentWillReceiveProps() {
//   console.log("[MyCourses] - UNSAFE_componentWillReceiveProps");
// }
// UNSAFE_componentWillUpdate() {
//   console.log("[MyCourses] - UNSAFE_componentWillReceiveProps");
// }


render() {
  console.log("[MyCourses]-render()");

    let courseListToDisplay = this.state.mycourseList.map(
      (mycourse) => (
        <tr key={mycourse.cid}>
          <td> <h6> {mycourse.cid} </h6></td>
          <td> <h6> {mycourse.cname} </h6></td>
          <td> <h6> {mycourse.price} </h6></td>
        </tr>
      )
    );
```

```
      return (
         <div className="container">
          <br/>
          <div className="container">
          <table className="table">
            <tbody>
            {courseListToDisplay}
            </tbody>
          </table>
          </div>
           </div>
       )
    }
}

export default MyCourses;
```

6. Hello.js

```
import React, { Component } from 'react';
import Hai from './Hai';


class Hello extends Component {

  static displayName = "MyHelloComponent";

  static defaultProps = {
    mytrainer : "Srinivas",
    mycolor: 'Red'
  }


  constructor(props){
    super(props);

    console.log("[Hello]-Constrcutor");

    this.state = {
      message:"OK Guys",
    };

  }
```

```
 changeHelloMessage = ()=> {
   console.log("Button - Clicked");
   this.setState({
     message:"Hello Guys, How are you!!!",
   });
 }

 static getDerivedStateFromProps(myprops,mystate){
  console.log("[Hello]-getDerivedStateFromProps()");
  return null;
}

componentDidMount(){
  console.log("[Hello]-componentDidMount()");
}

shouldComponentUpdate(nextProps, nextState) {
  console.log("[Hello] -shouldComponentUpdate");
  return true;
}

getSnapshotBeforeUpdate(prevProps, prevState) {
  console.log("[Hello] - getSnapshotBeforeUpdate");
       return null;
}

componentDidUpdate(prevProps, prevState, mysnapshot) {
  console.log("[Hello] -componentDidUpdate");
}

 componentWillUnmount() {
  console.log("[Hello] - componentWillUnmount");
}

render() {
  console.log("[Hello]-render()");

    return (

       <div className="container">
         <button
         className="btn btn-primary"
           onClick={this.changeHelloMessage} >
          Show and Hide Hello
           </button>
           <br/>
          <h3> {this.state.message} </h3>
          <h3> MyTrainer : {this.props.mytrainer} </h3>
          <h3> MyColor : {this.props.mycolor} </h3>
         <br/>
          <Hai mymessage={this.state.message}/>
         </div>
      )
    }
}

export default Hello;
```

7. Hai.js

```
import React, { Component } from 'react';

class Hai extends Component {

 static displayName = "MyHaiComponent";

  constructor(props){
    super(props);
    console.log("[Hai]-Constrcutor");
    this.state = {
    }
  }

    static getDerivedStateFromProps(myprops,mystate){
      console.log("[Hai]-getDerivedStateFromProps()");
      return null;
    }

    componentDidMount(){
      console.log("[Hai]-componentDidMount()");
    }

    shouldComponentUpdate(nextProps, nextState) {
      console.log("[Hai] -shouldComponentUpdate");
      return true;
     // return false;
    }

    getSnapshotBeforeUpdate(prevProps, prevState) {
      console.log("[Hai] - getSnapshotBeforeUpdate");
           return null;
    }
```

```
   componentDidUpdate(prevProps, prevState, mysnapshot) {
    console.log("[Hai] -componentDidUpdate");
  }
   componentWillUnmount() {
    console.log("[Hai] - componentWillUnmount");
  }
  render() {
    console.log("[Hai]-render()");
      return (
        <div className="container">
        <br/>
        <h3> {this.props.mymessage} </h3>
        <h3> {this.props.mymessage} </h3>
        <br/>
        </div>
      )
  }
}

export default Hai;
```

Summary

A) What happens when React is Rendering Component First Time (Mounting)

1) Creates the Component Instance by Calling the Constructor
2) Calls getDerivedStateFromProps(props,State) - Static method
3) calls render() method - Instance method

   Inside render() method, You can write Some Java Script Cod and JSX Code
   YOu can include other components here inside the render() ---
   If any other component found here then that components lifecycle will be started.
4) Calls componentDidMount() method

B) What happens when Component State Changes

1) calls getDerivedStateFromProps(props,State) - Static method
2) shouldComponentUpdate() - returns boolean value
3) calls render() method - Instance method

   Inside render() method, You can write Some Java Script Cod and JSX Code
   YOu can include other components here inside the render() ---
   If any other component found here then that components lifecycle will be started.
4) calls getSnapshotBeforeUpdate() metod
5) calls componentDidUpdate() method

C) What happens when Component is removed from DOM

1) calls componentWillUnMount() method
2)Destroys the Component Instance

React Hooks

- Hooks are a new addition in React 16.8. They let you use state and other React features without writing a class.

- Basically, using functions instead of classes can have a positive impact on the performance and the readability of the code

Rules for a Hooks

1. Only Call Hooks at the Top Level: You cannot call Hooks inside loops, conditionals, and nested functions. You should call it at the top level of your React function.

2. Do not call Hooks from a regular Javascript function. Always use a react function component or from a custom Hooks.

List of React Hooks

Basic Hooks	Additional Hooks
useState	useMemo
useEffect	useReducer
useContext	useRef
	useCallback
	useImperativeHandle
	useLayoutEffect
	useDebugValue

How do lifecycle methods correspond to Recct Hooks?

LifeCycle Hooks for Class Components	React Hooks for Fucntional Components
constructor()	Function components don't need a constructor. You can initialize the state in the **useState** call If computing the initial state is expensive, you can pass a function to useState.
getDerivedStateFromProps()	Schedule an update while rendering instead
shouldComponentUpdate()	**React.memo**
Render()	This is the function component body itself
componentDidMount componentDidUpdate componentWillUnmount	The **useEffect** Hook can express all combinations of these (including less common cases).
getSnapshotBeforeUpdate componentDidCatch getDerivedStateFromError	There are **no Hook equivalents** for these methods yet, may be added soon

useState() Hook

Usage

const [mystate, setMyState] = useState(myInitialState);

- Returns a stateful value, and a function to update it.
- During the initial render, the returned state (mystate) is the same as the value passed as the first argument (myInitialState).
- The setMyState function is used to update the state. It accepts a new state value and enqueues a re-render of the component.

Usage:

setMyState(newState);

Functional updates

- If the new state is computed using the previous state, you can pass a function to setMyState()
- The function will receive the previous value, and return an updated value.

Ex:

```
function Counter({initialCount}) {
  const [count, setCount] = useState(initialCount);
  return (
    <>
      Count: {count}
      <button onClick={() => setCount(initialCount)}>Reset</button>
      <button onClick={() => setCount(prevCount => prevCount - 1)}>-</button>
      <button onClick={() => setCount(prevCount => prevCount + 1)}>+</button>
    </>
  );
}
```

- What is the Differerence between setState()and useState() ?

Merging the New State with Current State.
Replace the Current State with NewState

useEffect() Hook

Usage:
useEffect(function);

- Accepts a function that contains important effectful code.

- Mutations, subscriptions, timers, logging, and other side effects are not allowed inside the main body of a function component. Doing so will lead to confusing bugs and inconsistencies in the UI.

- Write the Code for Mutations, subscriptions, timers, logging, and other side effects in side The function passed to useEffect

- The function passed to useEffect will run after the render is committed to the screen.

- By default, effects run after every completed render, but you can choose to fire them only when certain values have changed.

Cleaning up an effect

- Often, effects create resources that need to be cleaned up before the component leaves the screen, such as a subscription or timer ID. To do this, the function passed to useEffect may return a clean-up function.

- For example, to create a subscription:

```
useEffect(() => {
  const subscription = props.source.subscribe();
  return () => {
    // Clean up the subscription
    subscription.unsubscribe();
  };
});
```

- The clean-up function runs before the component is removed from the UI to prevent memory leaks.
- Additionally, if a component renders multiple times (as they typically do), the previous effect is cleaned up before executing the next effect.

Conditionally firing an effect

- The default behavior for effects is to fire the effect after every completed render.

- That way an effect is always recreated if one of its dependencies changes.

- However, this may be overkill in some cases, like the subscription example from the previous section. We don't need to create a new subscription on every update, only if the source prop has changed.

- To implement this, pass a second argument to useEffect that is the array of values that the effect depends on. Our updated example now looks like this:

```
useEffect(
  () => {
    const subscription = props.source.subscribe();
    return () => {
      subscription.unsubscribe();
    };
  },
  [props.source],
);
```

Now the subscription will only be recreated when props.source changes.

Note

- If you use this optimization, make sure the array includes all values from the component scope (such as props and state) that change over time and that are used by the effect.

Lab15 : useState() and useEffect() - React Hooks

Lab15: Files Required:

1. index.html	2. index.js
3. index.css	4. App.js
5. Courses.js	

1. index.html

2. index.js

3. index.css

4. App.js

```
import React,{ useEffect, useState}from 'react';

import 'bootstrap/dist/css/bootstrap.min.css';
import Courses from './Courses';

const App = () => {

  const [courseFlagState,setCourseFlagState]= useState(true);
```

```
useEffect( () => {
 console.log("[App] - 1st -useEffect()");
  //Http Calls
  //Timers

 return () =>{
   //CleanUp
   console.log("[App] -Cleanup-1st- useEffect()");
 }

});

useEffect(()=>{
  console.log("[App] - 2nd -useEffect()");
  //Http Calls
  //Timers


 return () =>{
  //CleanUp
  console.log("[App] -Cleanup-2nd- useEffect()");
}

},[]);

useEffect(()=>{
  console.log("[App] - 3rd - useEffect()");
  //Http Calls
  //Timers

 return () =>{
  //CleanUp
  console.log("[App] -Cleanup-3rd- useEffect()");
}

},[] );

const  hideShowCourses = ()=> {
 console.log("[App] - hideShowCourses");
  setCourseFlagState(!courseFlagState);
}

let mycourse = null;
if(courseFlagState === true){
 mycourse = ( <Courses/> );
}
```

```
   return (
    <div className="container">
    <h1 className="text-center"> Welcome to JLC!!!</h1>
    <br/>
    <button  className="btn btn-success"    onClick={hideShowCourses} >
        Hide Show Courses
     </button>
          <br/><br/><br/>
    {mycourse}
      </div>
     );
  }

export default App;
```

5. Courses.js

```
import React, { useState, useEffect } from "react";

const Courses = (props) => {
 const [coursesState, setCoursesState] = useState({
   trainerName: "Srinivas Dande",
   trainerEmail: "sri@jlc.com",
   mycourseList: [
    { cid: 101, cname: "Angular 7", price: 15000 },
    { cid: 102, cname: "React JS", price: 15000 },
    { cid: 103, cname: "Java Full Stack", price: 36000 },
    { cid: 104, cname: "Spring MicroServices", price: 22000 },
    { cid: 105, cname: "DevOps", price: 15000 },
   ],
 });

 useEffect(() => {
   console.log("[Courses] - 1st - useEffect()");
   //Http Calls
   //Timiers etc

   return () => {
    console.log("[Courses] - 1st - useEffect() - Cleanup");
    //Cleanup Work here
   };

 });
```

```
useEffect(() => {
  console.log("[Courses] - 2nd - useEffect()");
  //Http Calls
  //Timiers etc

  return () => {
    console.log("[Courses] - 2nd - useEffect() - Cleanup");
    //Cleanup Work here
  };

}, []);

useEffect(() => {
  console.log("[Courses] - 3rd - useEffect()");
  //Http Calls
  //Timiers etc

  return () => {
    console.log("[Courses] - 3rd - useEffect() - Cleanup");
    //Cleanup Work here
  };

}, [courseState.trainerName,courseState.trainerEmail]);

const showAllCourses = () => {
  console.log("showAllCourses - Called");

  setCoursesState({
    //trainerName: "Srinivas Dande",
    //trainerEmail: "srinivas@jlc.com",
    mycourseList: [
      { cid: 101, cname: "Angular 7", price: 12000 },
      { cid: 102, cname: "React JS", price: 12000 },
      { cid: 104, cname: "Spring MicroServices", price: 15000 },
    ],
  });
};
```

```
 let courseListToDisplay = coursesState.mycourseList.map(
  (mycourse, myindex) => (
   <tr key={myindex}>
    <td>
     <h6> {mycourse.cid}</h6>
    </td>
    <td>
     <h6> {mycourse.cname}</h6>
    </td>
    <td>
     <h6>{mycourse.price}</h6>
    </td>
   </tr>
  )
 );

 return (
  <div className="container">
   <div className="container" style={{ float: "right" }}>
    <button className="btn btn-primary mybutton" onClick={showAllCourses}>
     Show All Courses
    </button>
   </div>
   <div className="Container">
    <table className="table table-striped table-light table-bordered">
     <tbody>{courseListToDisplay}</tbody>
    </table>
   </div>
   <h2> Trainer Name : {coursesState.trainerName}</h2>
   <h2> Trainer Email : {coursesState.trainerEmail}</h2>
  </div>
 );
};

export default Courses;
```

Lab16 : Passing Function ref as props and using Font AweSome Icons

Lab16: Files Required:

8. index.html	9. index.js
10. index.css	11. App.js
12. Header.js	13. LeadsList.js
14. Lead.js	

8. index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <meta  name="viewport"
      content="width=device-width, initial-scale=1, shrink-to-fit=no">

    <script   src="https://kit.fontawesome.com/3a7b30e168.js"
      crossorigin="anonymous">
    </script>

    <title>JLC Lead Manager</title>
  </head>
  <body>
    <noscript>
      You need to enable JavaScript to run this app.
    </noscript>
    <div id="myroot"></div>

  </body>
</html>
```

9. index.js

```
import React from "react";
import ReactDOM from "react-dom";
import "./index.css";
import App from "./App";

ReactDOM.render(<App />, document.getElementById("myroot"));
```

10. index.css

11. App.js

```
import React, { Component } from "react";

import "bootstrap/dist/css/bootstrap.min.css";
import LeadsList from "./components/LeadsList";
import Header from "./components/Header";
```

```
class App extends Component {
  render() {
    return (
      <div className="App container">
        <Header mybrand="Lead Manager" />
        <LeadsList />
      </div>
    );
  }
}

export default App;
```

12. Header.js

```
import React from "react";
import PropTypes from "prop-types";

const Header = props => {
  return (
    <nav className="navbar navbar-expand-lg navbar-dark bg-danger">
      <a className="navbar-brand" href="">
        {props.mybrand}
      </a>

      <div className="collapse navbar-collapse" id="navbarNav">
        <ul className="navbar-nav">
          <li className="nav-item active">
            <a className="nav-link" href="">
              Home
            </a>
          </li>
          <li className="nav-item active">
            <a className="nav-link" href="">
              Leads
            </a>
          </li>
          <li className="nav-item active">
            <a className="nav-link" href="">
              Courses
            </a>
          </li>
        </ul>
      </div>
    </nav>
  );
};
```

```
Header.defaultProps = {
  mybrand: "My App"
};

Header.propTypes = {
  mybrand: PropTypes.string.isRequired
};

export default Header;
```

13. LeadsList.js

```
import React, { Component } from "react";
import Lead from "./Lead";

class LeadsList extends Component {
  state = {
    myleadsList: [
      { leadId: 101, name: "Sri", email: "sri@jlc", phone: 12345, course: "Java FSD" },
      { leadId: 102, name: "vas", email: "vas@jlc", phone: 55555, course: "Angular" },
      { leadId: 103, name: "srinivas", email: "srinivas@jlc", phone: 99999, course: "React" },
    ]
  };

  deleteMyLead = (leadId) => {
    console.log("LeadsList -deleteLead() ");
    const { myleadsList } = this.state;
    const myleads = myleadsList.filter(mylead => mylead.leadId !== leadId);
    this.setState({ myleadsList: myleads });
  };

  render() {
    return (
      <div>
        {this.state.myleadsList.map(mylead => (
          <Lead
            key={mylead.leadId}
            lead={mylead}
            mydeleteClickHandler={this.deleteMyLead.bind(this, mylead.leadId)}
          />
        ))}
      </div>
    );
  }
}
export default LeadsList;
```

14. Lead.js

```
import React, { Component } from "react";
import PropTypes from "prop-types";

class Lead extends Component {
 state = {
  showLeadInfo: false
 };

 onShowClick = () => {
  this.setState({
   showLeadInfo: !this.state.showLeadInfo
  });
 };

 onDeleteClick = () => {
  this.props.mydeleteClickHandler();
 };

 render() {
  const { lead } = this.props;
  let myleadInfo = null;
  if (this.state.showLeadInfo) {
   myleadInfo = (
    <ul className="list-group display-6">
     <li className="list-group-item"> Email : {lead.email} </li>
     <li className="list-group-item"> Phone : {lead.phone} </li>
    </ul>
   );
  }
  return (
   <div className="card card-body mb-4">
    <h4>
     {this.props.lead.name}  - Interested for {this.props.lead.course}
     <i
      onClick={this.onShowClick}
      className="fas fa-sort-down"
      style={{ cursor: "pointer", color: "green", fontSize: "50px" }}
     />
     <i
      onClick={this.onDeleteClick}
      className="fas fa-times"
      style={{ cursor: "pointer", float: "right", color: "red" }}
     />
    </h4>
```

```
        {myleadInfo}
      </div>
    );
  }
}

Lead.propTypes = {
  lead: PropTypes.object.isRequired,
  mydeleteClickHandler: PropTypes.func.isRequired
};

export default Lead;
```

Lab17 : Using Context API and Reducers

Lab17: Files Required:

1. index.html	2. index.js
3. index.css	4. appContext.ts
5. App.js	6. Header.js
7. LeadsList.js	8. Lead.js

1. index.html

```
<!DOCTYPE html>
<html lang="en">
 <head>
  <meta charset="utf-8" />
  <meta  name="viewport"
   content="width=device-width, initial-scale=1, shrink-to-fit=no">

  <script   src="https://kit.fontawesome.com/3a7b30e168.js"
   crossorigin="anonymous">
  </script>

  <title>JLC Lead Manager</title>
 </head>
 <body>
  <noscript>
   You need to enable JavaScript to run this app.
  </noscript>
  <div id="myroot"></div>

 </body>
</html>
```

2. index.js

```
import React from "react";
import ReactDOM from "react-dom";
import "./index.css";
import App from "./App";

ReactDOM.render(<App />, document.getElementById("myroot"));
```

3. index.css

4. appContext.js

```
import React, { Component } from "react";

const myContext = React.createContext();
const myreducer = (state, action) => {
  console.log("[appContent] - myreducer",action.type);
  switch (action.type) {
    case "DELETE_LEAD":
      return {
        myleadsList: state.myleadsList.filter(
          mylead => mylead.leadId !== action.payload
        )
      };
    default:
      return state;
  }
};

export class MyApplicationData extends Component{
   state = {
    myleadsList: [
      { leadId: 101, name: "Sri", email: "sri@jlc", phone: 12345, course: "Java FSD" },
      { leadId: 102, name: "vas", email: "vas@jlc", phone: 55555, course: "Angular" },
      { leadId: 103, name: "sd", email: "sd@jlc", phone: 99999, course: "React" },
      { leadId: 104, name: "ds", email: "ds@jlc", phone: 99999, course: "React" },
    ],
    mydispatch: (action) => {
      this.setState(state => myreducer(state, action));
    }
  }

  render(){
    return(
      <myContext.Provider value={this.state}>
      {this.props.children}
      </myContext.Provider>
    );
  }
}

export const MyDataConsumer = myContext.Consumer;
```

5. App.js

```
import React, { Component } from "react";
import "bootstrap/dist/css/bootstrap.min.css";
import "./App.css";
import Header from "./components/Header";
import LeadsList from "./components/LeadsList";
import { MyApplicationData } from "./appContext";
```

```
class App extends Component {
  render() {
    return (
      <MyApplicationData>
      <div className="App container">
       <Header mybrand="Lead Manager"></Header>
       <LeadsList/>
      </div>
      </MyApplicationData>
    );
  }
}
export default App;
```

6. **Header.js (Same as Lab16)**
7. **LeadsList.js**

```
import React, { Component } from "react";
import { MyDataConsumer } from "../appContext";
import Lead from "./Lead";

class LeadsList extends Component {
  render() {
    return(
      <MyDataConsumer>
       {
         value => {
          const { myleadsList} = value;
         return (
          <div>
           {myleadsList.map(
              mylead => ( <Lead   key={mylead.leadId} lead={mylead} />)
           )}
          </div>
         );
        }
       }
      </MyDataConsumer>
    );

  }
}

export default LeadsList;
```

8. Lead.js

```
import React, { Component } from "react";
import PropTypes from "prop-types";
import { MyDataConsumer } from "../appContext";

class Lead extends Component {

state = {
showLeadInfo: false
};

onShowClick = () => {
this.setState({
showLeadInfo: !this.state.showLeadInfo
});
};

onDeleteClick = (myleadId, mydispatch) => {
console.log("[Lead]- onDeleteClick");

mydispatch( { type: "DELETE_LEAD",payload: myleadId} );

};


render() {
const { lead } = this.props;
let myleadOtherInfo = null;
if (this.state.showLeadInfo) {
myleadOtherInfo = (
<ul className="list-group">
<li className="list-group-item"> Email : {lead.email} </li>
<li className="list-group-item"> Phone : {lead.phone} </li>
</ul>
);
}

return(
<MyDataConsumer>
{
value => {
const { mydispatch } = value;
const { lead } = this.props;
return (
<div className="card card-body mb-3">
<h4>
{lead.name}  - Interested for {lead.course}
<i
onClick={this.onShowClick}
className="fas fa-sort-down"
style={{ cursor: "pointer",color:"green",fontSize:"35px"}}
/>
<i
onClick={this.onDeleteClick.bind(this, lead.leadId, mydispatch)}
className="fas fa-times"
style={{ cursor: "pointer", float: "right", color: "red" }}
/>
</h4>
{myleadOtherInfo}
</div>
);
}
}
</MyDataConsumer>
);


}
}

Lead.propTypes = {
lead: PropTypes.object.isRequired,
};


export default Lead;
```

Lab18 : Using Context API

Lab18: Files Required:

1. index.html	2. index.js
3. StudentContext.ts	4. App.js
5. StudentComponent.js	

1. index.html

2. index.js

```
import React from "react";
import ReactDOM from "react-dom";
import App from "./App";

ReactDOM.render(<App />, document.getElementById("myroot"));
```

3. StudentContext.js

```
import React, { Component } from "react";

//1.Create the Context
const studentContext = React.createContext(null);

export class MyStudentProvider extends Component{
  //2. Define the State
  state = {
    studentId: 101,
    studentName: "Dandes",
    email: "dandes@jlc.com",
    phone: 55555,
  }

  render(){
    return(
      //3.Specify the Provider
      <studentContext.Provider value={this.state}>
      {this.props.children}
      </studentContext.Provider>
    );
  }
}

//4. Export the Consumer
export const MyStudentConsumer = studentContext.Consumer;
```

4. App.js

```
import React, { Component } from "react";

import "bootstrap/dist/css/bootstrap.min.css";

import { MyStudentProvider } from "./StudentContext";
import  StudentComponent  from "./StudentComponent";

class App extends Component {
  render() {
    return (
      <MyStudentProvider>
      <div className="container">
       <StudentComponent/>
      </div>
      </MyStudentProvider>
    );
  }
}

export default App;
```

5. StudentComponent.js

```
import React, { Component } from "react";
import { MyStudentConsumer } from "./StudentContext";

class StudentComponent extends Component {
  render() {
    return (
      <MyStudentConsumer>
        {
         mystudentData => {

          return(
              <div className="container">
              <br />
              <b><u>
                 <h2 className="text-center"> Student Info</h2>  </u> </b>
              <br />
              <div>
                <h3> Student ID : {mystudentData.studentId}</h3>
                <h3> Student Name : {mystudentData.studentName}</h3>
                <h3> Email Id: {mystudentData.email}</h3>
                <h3> Phone No : {mystudentData.phone}</h3>
              </div>
            </div>
            );
          }
        }
      </MyStudentConsumer>

    );
  }
}
export default StudentComponent;
```

Lab19 : Using Context API

Lab19: Files Required:

1. index.html	2. index.js
3. StudentContext.ts	4. CourseContext.ts
5. App.js	6. StudentComponent.js
7. CourseComponent.js	8. TestComponent.js

1. index.html

2. index.js

```
import React from "react";
import ReactDOM from "react-dom";
import App from "./App";

ReactDOM.render(<App />, document.getElementById("myroot"));
```

3. StudentContext.js

```
import React, { Component } from "react";

//1.Create the Context
const studentContext = React.createContext(null);

export class MyStudentProvider extends Component{
  //2. Define the State
  state = {
    studentId: 101,
    studentName: "Dandes",
    email: "dandes@jlc.com",
    phone: 55555,
  }
  render(){
    return(
      //3.Specify the Provider
      <studentContext.Provider value={this.state}>
      {this.props.children}
      </studentContext.Provider>
    );
  }
}

//4. Export the Consumer
export const MyStudentConsumer = studentContext.Consumer;
```

4. CourseContext.js

```
import React, { Component } from "react";

//1.Create the Content
const courseContext = React.createContext(null);

export class MyCourseProvider extends Component{
  //2. Define the State
  state = {
    courseId: "C-11",
    courseName: "Spring MicroServices",
    cost: 18000,
    trainer: "Srinivas Dande",
  }

  render(){
    return(
      //3.Specify the Provider
      <courseContext.Provider value={this.state}>
      {this.props.children}
      </courseContext.Provider>
    );
  }
}

//4. Export the Consumer
export const MyCourseConsumer = courseContext.Consumer;
```

5. App.js

```
import React, { Component } from "react";

import "bootstrap/dist/css/bootstrap.min.css";

import { MyStudentProvider } from "./StudentContext";
import { MyCourseProvider } from "./CourseContext";

import  StudentComponent  from "./StudentComponent";
import  CourseComponent  from "./CourseComponent";
import  TestComponent  from "./TestComponent";

class App extends Component {
 render() {
  return (
   <MyStudentProvider>
    <MyCourseProvider>
   <div className="container">

    <StudentComponent/>
    <CourseComponent/>
    <hr/>
    <TestComponent/>

   </div>
   </MyCourseProvider>
   </MyStudentProvider>
  );
 }
}

export default App;
```

6. StudentComponent.js

```
import React, { Component } from "react";
import { MyStudentConsumer } from "./StudentContext";

class StudentComponent extends Component {
 render() {
  return (
   <MyStudentConsumer>
    {
     mystudentData => {
      const { studentId, studentName, email, phone } = mystudentData;
      return(
        <div className="container">
        <br />
        <b><u>
          <h2 className="text-center"> Student Info</h2>  </u> </b>
        <br />
        <div>
         <h3> Student ID : {studentId}</h3>
         <h3> Student Name : {studentName}</h3>
         <h3> Email Id: {email}</h3>
         <h3> Phone No : {phone}</h3>
        </div>
       </div>
       );

     }
    }
   </MyStudentConsumer>

  );
 }
}
export default StudentComponent;
```

7. CourseComponent.js

```
import React, { Component } from "react";

import { MyCourseConsumer } from "./CourseContext";

class CourseComponent extends Component {
 render() {
  return (
   <MyCourseConsumer>
    {
     mycourseData => {
      const { courseId, courseName, cost, trainer } = mycourseData;
      return(
        <div className="container">
        <br />
        <b><u>
          <h2 className="text-center"> Course Info</h2>  </u> </b>
        <br />
        <div>
        <h3> Course Id: {courseId}</h3>
           <h3> Course Name : {courseName}</h3>
           <h3> Cost : {cost}</h3>
           <h3> Trainer : {trainer}</h3>
        </div>
       </div>
       );

     }
    }
   </MyCourseConsumer>

  );
 }
}
export default CourseComponent;
```

8. TestComponent.js

```
import React, { Component } from "react";

import { MyStudentConsumer } from "./StudentContext";
import { MyCourseConsumer } from "./CourseContext";

class TestComponent extends Component {
render() {
return (
    <MyStudentConsumer>
    {
    mystudentData => {
    const { studentId, studentName, email, phone } = mystudentData;
    return(
              <MyCourseConsumer>
              {
              mycourseData => {
              const { courseId, courseName, cost, trainer } = mycourseData;
              return (
              <div className="container">
              <br />
              <b><u>
              <h2 className="text-center"> Student Info</h2>  </u> </b>
              <br />
              <div>
              <h3> Student ID : {studentId}</h3>
              <h3> Student Name : {studentName}</h3>
              <h3> Email Id: {email}</h3>
              <h3> Phone No : {phone}</h3>
              <h3> Course Id: {courseId}</h3>
              <h3> Course Name : {courseName}</h3>
              <h3> Cost : {cost}</h3>
```

```
                <h3> Trainer : {trainer}</h3>
                </div>
                </div>
                )

                }
                }
                </MyCourseConsumer>
        );

        }
        }
        </MyStudentConsumer>
    );

    }
    }
    export default TestComponent;
```

Lab20 : Using Context API

Lab20: Files Required:

1. index.html	2. index.js
3. courseTypes.js	4. courseContext.ts
5. App.js	6. CourseComponent.js

1. index.html

2. index.js

```
import React from "react";
import ReactDOM from "react-dom";
import App from "./App";

ReactDOM.render(<App />, document.getElementById("myroot"));
```

3. courseTypes.js

```
export const UPDATE_CNAME="UPDATE_CNAME";
export const UPDATE_PRICE="UPDATE_PRICE";
export const UPDATE_DURATION="UPDATE_DURATION";
export const UPDATE_TRAINER="UPDATE_TRAINER";
```

4. courseContext.js

```
import React, { Component } from "react";
import {UPDATE_CNAME,UPDATE_PRICE } from './courseTypes';
import { UPDATE_DURATION,UPDATE_TRAINER} from './courseTypes';

//1.Create the Context
const courseContext = React.createContext(null);

//5.Define the Reducers
const myreducer = (currentState,action) => {
console.log("C","[courseContext]",action.type,action.payload);
switch(action.type){
  case UPDATE_CNAME:
    return {
      courseName:action.payload
    };
  case UPDATE_PRICE:
    return {
       price:action.payload
    };
  case UPDATE_DURATION:
    return {
       duration:action.payload
    };
    case UPDATE_TRAINER:
      return {
         trainer:action.payload
      };
      default:
      return currentState
}
}

export class MyCourseProvider extends Component{
  //2. Define the State Properties and Dispatch function
  state = {
    courseId: "C-101",
    courseName: "Master React ",
    price: 10000,
    duration:"50 Hrs",
    trainer: "SD",
    mydispatch: (action) => {
      console.log("B","courseContext - mydispatch");
      this.setState(state=> myreducer(this.state,action));
    }
  }

  render(){
    return(
      //3.Specify the Provider
      <courseContext.Provider value={this.state}>
      {this.props.children}
      </courseContext.Provider>
    );
  }
}

//4. Export the Consumer
export const MyCourseConsumer = courseContext.Consumer;
```

5. App.js

```
import React, { Component } from "react";
import "bootstrap/dist/css/bootstrap.min.css";

import { MyCourseProvider } from "./CourseContext";
import  CourseComponent  from "./CourseComponent";

class App extends Component {
 render() {
    return (
      <MyCourseProvider>
     <div className="container">

      <CourseComponent/>

     </div>
     </MyCourseProvider>
    );
  }
}

export default App;
```

6. CourseComponent.js

```
import React, { Component } from "react";

import {UPDATE_CNAME,UPDATE_PRICE } from './courseTypes';
import { UPDATE_DURATION,UPDATE_TRAINER} from './courseTypes';

class CourseComponent extends Component {

updateCourseName = (updatedCname,mydispatch) => {
console.log("A","CourseComponent - updateCourseName",updatedCname);
mydispatch({
type:UPDATE_CNAME,
payload:updatedCname
});
}

updatePrice = (updatedPrice,mydispatch) => {
console.log("A","CourseComponent - updatePrice",updatedPrice);
mydispatch({
type:UPDATE_PRICE,
payload:updatedPrice
});
}

updateDuration = (updatedDuration,mydispatch) => {
console.log("A","CourseComponent - updateDuration",updatedDuration);
mydispatch({
type:UPDATE_DURATION,
payload:updatedDuration
});
}
```

```
updateTrainer = (updatedTrainer,mydispatch) => {
console.log("A","CourseComponent - updateTrainer",updatedTrainer);
mydispatch({
type:UPDATE_TRAINER,
payload:updatedTrainer
});
}

render() {
return (

<MyCourseConsumer>
{
mycourseData => {

const { courseId, courseName, price, duration,trainer } = mycourseData;
const { mydispatch } = mycourseData;

return(
<div className="container">
<br />
<b><u>
<h3 className="text-center"> Course Info</h3>  </u> </b>
<br />
<div>
<table className="table">
<tbody>
<tr>
<td><h4> Course Id: {courseId}</h4> </td>
<td> </td>
</tr>
<tr>
<td><h4> Course Name : {courseName}</h4> </td>
<td>
<button onClick={this.updateCourseName.bind(this,"Master React 16",mydispatch)}
className="btn btn-success">
Update Course Name
</button>
</td>
</tr>
<tr>
<td><h4> Price : {price}</h4> </td>
<td>
<button onClick={this.updatePrice.bind(this,13000,mydispatch)} className="btn
btn-success">
Update Price
</button>
</td>
</tr>
<tr>
<td><h4> Duration : {duration}</h4> </td>
<td>
<button onClick={this.updateDuration.bind(this,"75 Hrs",mydispatch)}
className="btn btn-success">
Update Duration
</button>
</td>
</tr>
<tr>
<td><h4> Trainer : {trainer}</h4> </td>
<td>
<button onClick={this.updateTrainer.bind(this,"Srinivas Dande",mydispatch)}
className="btn btn-success">
Update Trainer
</button>
</td>              </tr>
</tbody>
</table>

</div>
</div>

);
}
}
</MyCourseConsumer>

);
}
}
export default CourseComponent;
```