

BSc (Hons) in Information Technology
Specializing in Software Engineering
Year 3 - 2022
SE3040 – Application Frameworks
Lab 08 – ReactJS

1. Init a npm project
2. Install “parcel@next” as a dev dependency.
3. Install “react react-dom” as dependencies.
4. Create 2 files index.html and index.jsx.
5. Add a <div> to the html with the id “app”.
6. Import index.jsx into index.html at the bottom. After the </div> close tag and before the </body> close tag.
7. Add the following code to index.jsx. You are creating your first React element and mounting it to the <div> created.

```
import React from 'react';
import {render} from 'react-dom';

render(<h1>Hello to React</h1>, document.getElementById('app'));
```

8. Now run the app by adding a start script to package.json file “parcel index.html”. Now you should be able to see your application by running “npm start” and login into <http://localhost:1234>.
9. Create a React component App.jsx.

```
import React from 'react';

export default class App extends React.Component {
  constructor(props) {
    super(props);
  }

  render() {
    return <h1>Hello to React</h1>;
  }
}
```

10. Mount the App component in the index.jsx.
11. Run the application and check the output.
12. Create a list of posts as a const in the App.jsx file (not inside the component).

```
const posts = [
  {
    id: 1,
    name: 'React',
    description: 'Best UI library'
  }, {
    id: 2,
    name: 'Node',
    description: 'Server side JS'
  }
];
```

13. Render these posts in the App.jsx.

```
render() {  
  return <div>  
    <table>  
      <thead>  
        <tr>  
          <th>ID</th>  
          <th>Name</th>  
          <th>Description</th>  
        </tr>  
      </thead>  
      <tbody>  
        {posts.map(post => {  
          return <tr key={post.id.toString()}>  
            <td>{post.id}</td>  
            <td>{post.name}</td>  
            <td>{post.description}</td>  
          </tr>  
        })}  
      </tbody>  
    </table>  
  </div>  
};
```

14. Run the application and check.

15. Now, let's break this down to child components.

a. create a functional component to display data of a one item as a table row.

```
import React from 'react';  
  
export default function PostListItem(props) {  
  const {post} = props;  
  return <tr>  
    <td>{post.id}</td>  
    <td>{post.name}</td>  
    <td>{post.description}</td>  
  </tr>  
}
```

b. Add another component to hold the posts list.

```
import React from 'react';  
import PostListItem from './PostListItem';  
  
export default class Posts extends React.Component {  
  constructor(props) {  
    super(props);  
  }  
}
```

BSc (Hons) in Information Technology

Specializing in Software Engineering

Year 3 - 2022

SE3040 – Application Frameworks

Lab 08 – ReactJS

```
render() {  
  const {posts} = this.props;  
  return <div>  
    <table>  
      <thead>  
        <tr>  
          <th>ID</th>  
          <th>Name</th>  
          <th>Description</th>  
        </tr>  
      </thead>  
      <tbody>  
        {posts.map(post => {  
          return <PostListItem  
key={post.id.toString()} post={post}/>  
        })}  
      </tbody>  
    </table>  
  </div>;  
}
```

- c. Now import and add the Posts component to App.

```
render() {  
  return <div>  
    <Posts posts={posts}/>  
  </div>;  
}
```

16. Run the application and check the output.

17. Add another component to show a Post.

```
import React from 'react';  
export default class Post extends React.Component {  
  constructor(props) {  
    super(props);  
  }  
  
  render() {  
    const {post} = this.props;  
    return <div>  
      <div>  
        <p>ID: {post.id}</p>  
      </div>  
      <div>  
        <p>Name: {post.name}</p>  
      </div>  
    </div>  
  }  
}
```

BSc (Hons) in Information Technology

Specializing in Software Engineering

Year 3 - 2022

SE3040 – Application Frameworks

Lab 08 – ReactJS

```

    <div>
      <p>Description: {post.description}</p>
    </div>
  </div>;
}

}

```

18. Introduce the state to Posts component in the constructor.

```

this.state = {
  post: null
};

```

19. Add a method to set this newly added state in the Posts component.

```

selectPost(post) {
  this.setState({post: post})
}

```

20. Pass this method to Post component created. This is after the table in the Posts component. Post component is rendering conditionally.

```

<div>
  {this.state.post ? <Post post={this.state.post}/> : ''}
</div>

```

21. Add an action to PostListItem component as the last column. Make sure you add a new <th> element to <thead> as well.

```

<td><a onClick={() => selectPost(post)}>Select</a></td>

```

22. Pass the method selectPost to PostListItem component.

```

<PostListItem key={post.id.toString()} post={post}
  selectPost={post => this.selectPost(post)}/>

```

23. Let's install react router. "react-router-dom".

24. Let's add a new component to hold Posts related components. Remove the duplicate posts and other stuff from the App component.

```

import React from 'react';
import Posts from './Posts';

const posts = [
  {
    id: 1,
    name: 'React',
    description: 'Best UI library'
  }, {
    id: 2,
    name: 'Node',

```

BSc (Hons) in Information Technology
Specializing in Software Engineering
Year 3 - 2022
SE3040 – Application Frameworks
Lab 08 – ReactJS

```
        description: 'Server side JS'
    }
];

export default class PostsHolder extends React.Component {
    constructor(props) {
        super(props);
    }

    render() {
        return <Posts posts={posts}/>;
    }
}
```

25. Run the application and check the output.

26. Let's add routing to App component to land in the PostHolder.

```
import React from 'react';
import {BrowserRouter as Router, Switch, Route} from 'react-router-dom';
import PostsHolder from './components/PostsHolder';

export default class App extends React.Component {
    constructor(props) {
        super(props);
    }

    render() {
        return <Router>
            <Switch>
                <Route exact path="/">
                    <PostsHolder/>
                </Route>
            </Switch>
        </Router>
    }
}
```

27. Run the application and check the output.

28. Let's introduce a method to add a new Post to PostHolder.

```
addNewPost({name, description}) {
    posts.push({id: posts.length + 1, name, description});
}
```

29. Add a component AddPost to add a post. Introduce name and description to state and bind them to a form. Handle the onChange event to update the state.

```
import React from 'react';
```

BSc (Hons) in Information Technology

Specializing in Software Engineering

Year 3 - 2022

SE3040 – Application Frameworks

Lab 08 – ReactJS

```
export default class AddPost extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      name: '',
      description: ''
    }
  }

  onChange(event) {
    const {name, value} = event.target;
    this.setState({[name]: value})
  }

  render() {
    const {save} = this.props;
    return <div>
      <form>
        <div>
          <label htmlFor="name">Name: </label>
          <input type="text" name="name" id="name"
value={this.state.name}
onChange={event =>
this.onChange(event)} />
        </div>
        <div>
          <label htmlFor="description">Description:
</label>
          <input type="text" name="description"
id="description" value={this.state.description}
onChange={event =>
this.onChange(event)} />
        </div>
      </form>
    </div>;
  }
}
```

30. Now add method to handle the adding a new Post into the component AddPost with a button (this should be inside the form as the last).

```
<div>
  <button onClick={event => {
    event.preventDefault();
    save({name: this.state.name, description:
this.state.description});
    this.setState({
      name: '',
      description: ''
    })
  }}>Add Post</button>
</div>
```

BSc (Hons) in Information Technology
Specializing in Software Engineering
Year 3 - 2022
SE3040 – Application Frameworks
Lab 08 – ReactJS

```
    })  
  }}>Save  
</button>  
</div>
```

31. Add the routing to AddPost component in the PostHolder.

- a. This should be a child element of the Switch.

```
<Route exact path="/add">  
  <AddPost save={this.addNewPost}/>  
</Route>
```

- b. This should be a child element of Router but not Switch.

```
<Link to="/add">Add</Link>
```

32. Add a link to Posts in the AddPost component (inside the parent div)

```
<Link to="/">Posts</Link>
```