SE3040 -Application Frameworks

Lab Exercise 5

Software Engineering - Year - 3 **Semester 1, 2018**

Lab Session: ReactJS

Objective: Teach main features of ReactJS

1. Create a node project.
npm init
2. Install webpack and babel related dependencies.
npm install webpack webpack-dev-server webpack-cli babel-loader babel-core babel-preset-env babel-preset-react --save-dev
3. Install React JS dependencies.
npm install react react-dom prop-types --save-dev
4. Create webpack config file named webpack.config.js

```
'use strict';

const path = require('path');

module.exports = {
   entry: path.resolve(__dirname, "app.jsx"),
   output: {
       path: path.resolve(__dirname, "dist"),
       filename: 'bundle.js'
   },
   module: {
       rules: [
           {
               test: /\.jsx?$/,
               use: {
                   loader: "babel-loader",
                   options: {
                       presets: ["env", "react"]
                   }
               }
           }
       ]
   },
   resolve: {
       extensions: [".js", ".jsx"]
   },
    devServer: {
        contentBase: path.join(__dirname, "/"),
        compress: true
    },
    devtool: "source-map"
};
```

5. Create index.html file.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>React JS</title>
</head>
<body>
<div id="app"></div>
<script src="bundle.js"></script>
</body>
</html>
```

6. Create main application container file as AppContainer.jsx

```
'use strict';

import React, {Component} from 'react';

export default class AppContainer extends Component {
    constructor(props) {
        super(props);
    }

    render() {
        return <div>
            <h2>Hello World</h2>
        </div>;
    }
}
```

7. Add new JSX file as the entry point for the React application.

```
'use strict';

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

import AppContainer from './AppContainer.jsx';

render(<AppContainer/>, document.getElementById('app'));
```

8. Add start script into the scripts block in package.json file.

   ```
   "start": "webpack-dev-server --mode development"
   ```

9. Run the application.

   npm start

10. Create a file called User.jsx to display information belong to a single user in a table row.

```
'use strict';

import React from 'react';

const User = props => {
   const {user} = props;
   return <tr>
       <td>{user.id}</td>
       <td>{user.name}</td>
   </tr>
};

export default User;
```

11. Create a file called Users.jsx to handle displaying user list. Use the previously created User component in Users component.

```
'use strict';

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

import User from './User.jsx';

export default class Users extends Component {
   static get propTypes() {
       return {
           users: PropTypes.array
       }
   }

   constructor(props) {
       super(props);
   }

   render() {
       const {users} = this.props;
       return <div>
           <table>
               <thead>
               <tr>
```

```
                        <th>ID</th>
                        <th>Name</th>
                    </tr>
                    </thead>
                    <tbody>
                    {
                        users.map(user => {
                            return <User key={user.id} user={user}/>
                        })

                    }
                    </tbody>
                </table>
            </div>;
    }

}
```

12. Add Users component to AppContainer component.

```
'use strict';

import React, {Component} from 'react';

import Users from './Users';

export default class AppContainer extends Component {
    constructor(props) {
        super(props);
        this.state = {
            users: [{
                id: Date.now(),
                name: 'John'
            }]
        }
    }

    addUser(user) {
        this.setState({
            users: this.state.users.concat({id: Date.now(), name:
    user.name})
        })
    }

    render() {
        return <div>
            <h2>Users App</h2>
            <Users users={this.state.users}/>
        </div>;
    }
}
```

13. Create another component AddUser to add new users.

```
'use strict';

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

export default class AddUser extends Component {
   static get propTypes() {
       return {
           addUser: PropTypes.func
       }
   }

   constructor(props) {
       super(props);
   }

   onNameChange(event) {
       event.preventDefault();
       event.stopPropagation();
       this.name = event.target.value;
   }

   onSubmit(event) {
       event.preventDefault();
       event.stopPropagation();
       if (this.name) {
           this.props.addUser({name: this.name});
           this.name = '';
       }
   }

   render() {
       return <div>
           <form onSubmit={event => this.onSubmit(event)}>
               <label>Name:</label>
               <input type="text" onChange={event =>
   this.onNameChange(event)}/>
               <button type="submit">Add</button>
           </form>
       </div>;
   }
}
```

14. Update AppContainer component to cater user adding.

```
'use strict';

import React, {Component} from 'react';

import AddUser from './AddUser';
import Users from './Users';

export default class AppContainer extends Component {
   constructor(props) {
        super(props);
        this.state = {
            users: [{
                id: Date.now(),
                name: 'John'
            }]
        }
   }

   addUser(user) {
        this.setState({
            users: this.state.users.concat({id: Date.now(), name:
   user.name})
        })
   }

   render() {
        return <div>
            <h2>Users App</h2>
            <AddUser addUser={user => this.addUser(user)}/>
            <Users users={this.state.users}/>
        </div>;
   }
}
```

15. Run the application can check the output.

    npm start