BSc (Hons) in Information Technology
Specializing in Software Engineering
Year 3 - 2021
SE3040 – Application Frameworks
Lab 08 – ReactJS [Part 02 – Hooks and Context API]

1. Covert "Posts" component to a functional component.
2. Declare State in the "Posts" functional component. Using "useState" hook inside the component function.

```
const [post, setPost] = useState(null);
```

3. Use "setPost" method from hook when passing "selectPost" props to "PostListItem".

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

4. Use "Post" state from hook when passing props to "Post" component.

```
<Post post={post} />
```

Extra: Add "useEffect" hook to "Posts" functional component and observe it's behavior.
Ex:

```
useEffect(() => {
    console.log('Mount and update from the effect hook');
    return () => console.log("Unmount from the effect hook");
});
```

5. Create a file named "UserContext.jsx" and declare new context.

```
import React from 'react';

const user = {
    name: 'app_user'
}

const UserContext = React.createContext(user);

export default UserContext;
```

6. Introduce "user" state to "App" component. Set user state to "admin" in the "componentDidMount" life-cycle method.
7. In the "App" component enclose "PostsHolder" component inside the "UserContext".

```
<UserContext.Provider value={this.state.user}>
    <PostsHolder/>
</UserContext.Provider>
```

8. Use this context inside the AddPost component.

```
<p>{this.context.name}</p>
```