React

Hooks

What are Hooks?

- Hooks are a new addition to React in version 16.8 that allows you use state and other React features, like lifecycle methods, without writing a class.
- Classes in TypeScript encourage multiple levels of inheritance that quickly increase overall complexity and potential for errors.
- Hooks are the functions which "hook into" React state and lifecycle features from functional components.
- Hooks are backward-compatible, which means it does not contain any breaking changes
- The benefits of hooks are:
  - Managing state has become easier
  - Our code is significantly simplified, and more readable
  - It’s easier to extract and share stateful logic in our apps

Convert Class Component to Functional Component with Hook

Steps to quickly change a class component to a functional component with hooks:

- Create a new type or interface for Props

```
type LoginProps = {
};
```

- Change class to function

```
class Login extends React.Component {
```

```
const Login: FC<LoginProps> = ({}): ReactElement => {
```

- Remove the constructor (if any)
- Remove the render() method, keep the return
- Add const before all methods & class variables

- Replace React.createRef() with React.useRef() (if any)

```
React.createRef<any>();
```

⇒

```
React.useRef<any>();
```

- Remove the State type (if any), then set initial state with useState()

```
const [message, setMessage] = useState("");
```

- Remove this.state throughout the component

```
this.state?.message
```

⇒

```
message
```

- Change this.setState() … instead, call the function that you named in the previous step to update the state…

```
this.setState({ message: "Good!" });
```

⇒

```
setMessage("Good!");
```

- Remove all references to 'this' throughout the component
- Replace componentDidMount with useEffect()

```
componentDidMount(): void {
  this.usernameRef.current?.focus();
}
```

```
useEffect(() => {
  usernameRef.current?.focus();
}, []);
```

- Use useNavigate() of "react-router-dom" for redirecting (if any)

```
let navigate = useNavigate();
```

```
navigate("/home");
```

- Use useParam() of "react-router-dom" for getting URL parameters values (if any)

Simple React Snippet Extension

- Use **imrse** to import React, useState and useEffect.

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

- Use **sfc** to create a stateless functional component.

```
const Student = () => {
    return (  );
}

export default Student;
```

- Use **usf** to declare a new state variable using State Hook

```
const [, set] = useState();
```

- Use **uef** to use useEffect.

```
useEffect(() => {


}, []);
```

- Use **cpf** to create Class Property Function.

```
 = (e) => {


};
```

Exercise

- Convert all class components in this project into functional components with hooks

THE END