React StreamList App Setup: Part 2

Pantea Namiranian

The University of Arizona Global Campus

INT499: Capstone for Information Technology (INP2520A)

John Russell

5/19/2025

React StreamList App Setup

Completing the first part of the StreamList React application assignment was a great learning experience that helped to improve understanding of React, component-based design, and user interface creation. Here you would find the development process, focusing on building the navigation bar with React Router, creating the form components. The goal of this paper is to give a detailed overview of the developement steps, the challenges and the final outcome of the StreamList application.

Navigation Bar and React Router

The first important step in developing the StreamList application was making the navigation bar and here we use React Router to enable smooth navigation between different components. Setting up the router required importing essential components from the react-router-dom library, such as BrowserRouter, Route, and Link. The navigation bar has been organized as an unordered list, with each list item containing a Link component that directed users to the relevant pages (StreamList, Movies, Cart, About).

The React Router provided a straightforward way to manage navigation without reloading the entire page. However, set up routes and link them to the components was challenging but by using React Router, the application can provide a smooth user experience, enabling users to access various features without reloading the page. This structure is essential for building dynamic web applications that require multiple views or components. Below is a breakdown of the Routing Implementation:

1. Importing Required Components:
    - The script begins by importing React and necessary components from the react-router-dom library, including BrowserRouter, Route, Switch, and Link.

- It also imports the individual components for each page: StreamList, Movies, Cart, and About.

2. Setting Up the Router:
    - The entire application is wrapped in the component. This enables routing functionality throughout the app.
3. Navigation Bar:
    - A navigation bar is created using an unordered list (<ul>). Each list item (<li>) contains a component that navigates to different routes: StreamList (homepage), Movies, Cart, About
    - The to attribute of each <Link> specifies the path to navigate to when the link is clicked.
4. Defining Routes:
    - The <Switch> component is used to render only the first matching <Route>. This means that when the URL matches a specific path, the corresponding component will be displayed.
    - The routes are defined as follows:
        - path="/" with exact prop to render the StreamList component when the user is on the homepage.
        - path="/movies" to render the Movies component.
        - path="/cart" to render the Cart component.
        - path="/about" to render the About component.

Form Component Implementation

The next step involved completing the StreamList component, which allows users to input their desired movies or programs, to achieve that, there would be a form that captures user input and logs it to the console. To manage the input state, the useState hook had been utilized, which is a fundamental aspect of React for handling component state.

In the form, there is an input field and a submit button. The handleInputChange function updates the state as the user types, while the handleSubmit function prevents the default form submission behavior and logs the input value to the console. This approach ensures that the application can effectively capture user input and respond accordingly. Below is a breakdown of the complete the Form Component:

1. Component Creation:
    - The StreamList component is used to serve as the homepage where users can input the titles of movies or programs they want to watch.
2. Setting Up State Management:
    - The useState hook from React is to manage the input value. This hook allows us to create a state variable that holds the current value of the input field:

```
const [inputValue, setInputValue] = useState('');
```

    - Here, inputValue is the state variable that stores the user's input, and setInputValue is the function used to update this state.
3. Creating the Input Field:
    - An element is added to the form where users can type in the name of a movie or program:

```
<input
                    type="text"
```

```
    value={inputValue}
    onChange={handleInputChange}
    placeholder="Add a movie or program"
/>
```

- The value prop of the input is set to inputValue, which means the input field will display whatever is stored in the state.
- The onChange event handler is set to handleInputChange, which updates the state whenever the user types in the input field.

4. Handling Input Changes:
    - The handleInputChange function is defined to handle changes in the input field.

```
const handleInputChange = (e) => {
        setInputValue(e.target.value);
};
```

- This function takes the event object e as a parameter and uses e.target.value to get the current value of the input field. It then updates the inputValue state with this new value.

5. Creating the Form Submission Handler:
    - I implemented a handleSubmit function to manage the form submission:

```
const handleSubmit = (e) => {
    e.preventDefault(); // Prevents the default form submission behavior
    console.log(inputValue); // Logs the input value to the console
    setInputValue(''); // Resets the input field
};
```

- The e.preventDefault() method is used to stop the default behavior of the form, which would normally cause a page to reload.
- After logging the input value to the console for debugging purposes, I reset the input field by calling setInputValue('').

6. Form Structure:

    - I wrapped the input field and the submit button in a element:

```
<form onSubmit={handleSubmit}>
                    // Input and button here
                </form>
```

    - The onSubmit event of the form is set to handleSubmit, which ensures that the form submission is handled correctly.

Insights and Challenges

Throughout the development process, several insights and challenges emerged. One notable hurdle was managing the state effectively within the StreamList component, it was challenging to understand how to properly use the useState hook to manage form inputs. However, through practice and experimentation, a better grasp of state management in React had been gained which solved the issue.

Another insight was the importance of user experience in application design. By focusing on creating an intuitive interface, we can realize how crucial it is for users to have a seamless experience while interacting with the application. This understanding will guide the future projects as we continue to develop our skills in web development.

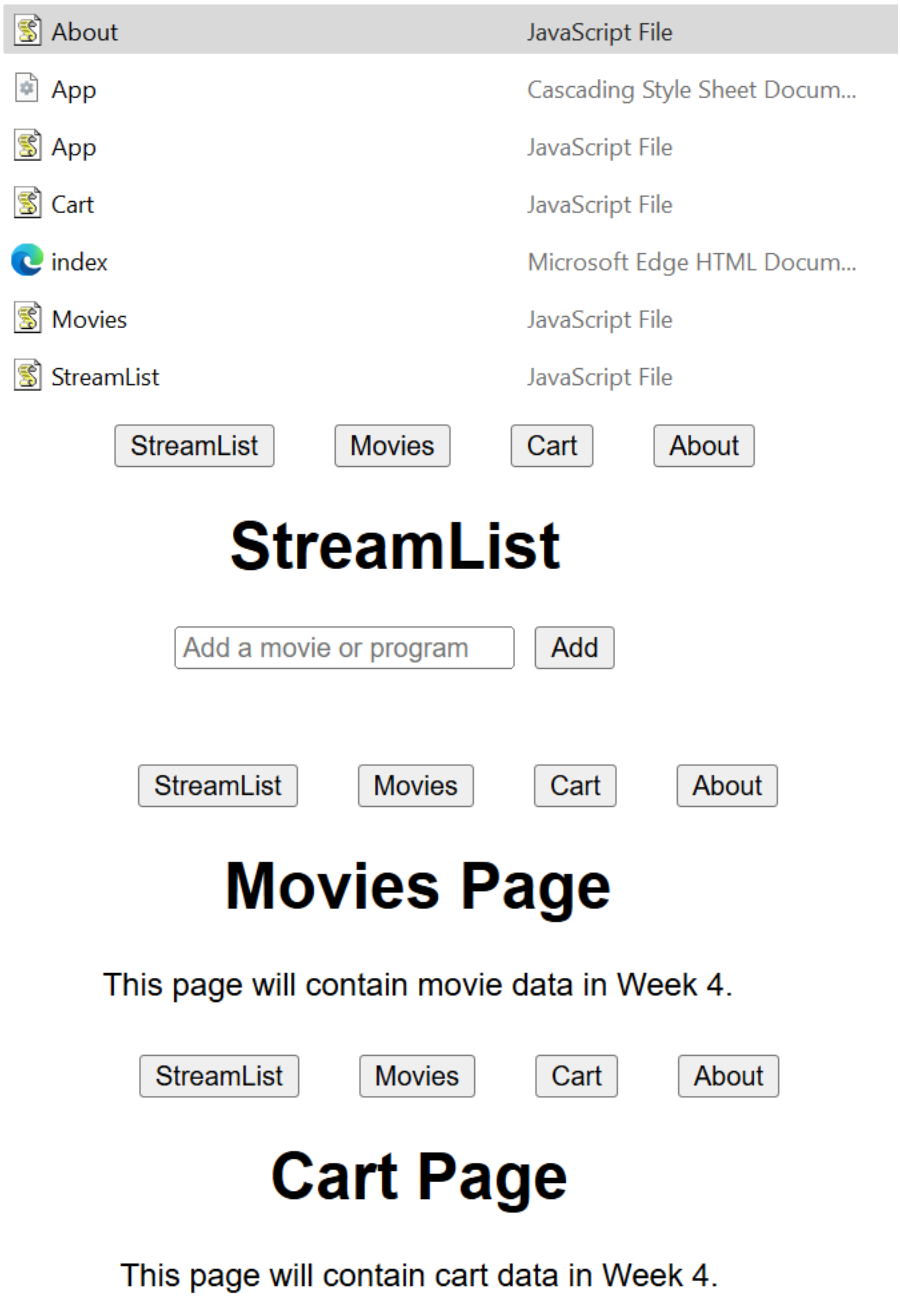
Conclusion

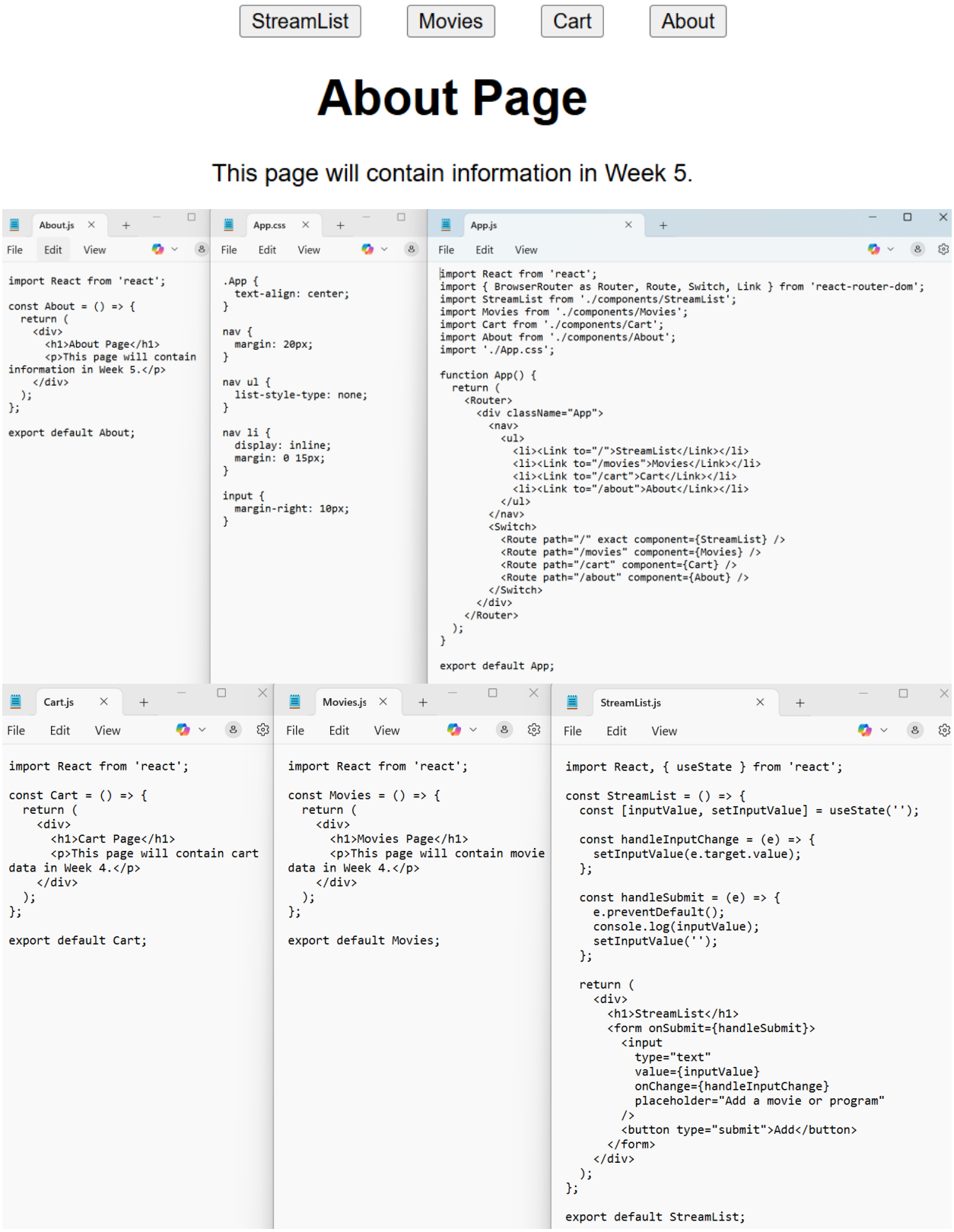
In summary, the development of the StreamList React application provided a comprehensive learning experience that reinforced understanding of React and web development concepts. By successfully implementing the navigation bar with React Router and creating a functional form component, we successfully provide the groundwork for a user-friendly application. The challenges faced and insights gained during this process will undoubtedly enhance our future endeavors in software development. For further review, the complete code for

the StreamList application can be found in the GitHub repository:

[https://github.com/panteanam/Pantea-Namiranian---INT499-Week-1-Assignment-Part-1.git]

and screenshots below:




StreamList
Movies
Cart
About
About Page
This page will contain information in Week 5.
About.js
File Edit View
import React from 'react';
const About = () => {
return (

<h1>About Page</h1>
This page will contain information in Week 5.

);
};
export default About;
App.css
File Edit View
.App {
text-align: center;
}
nav {
margin: 20px;
}
nav ul {
list-style-type: none;
}
nav li {
display: inline;
margin: 0 15px;
}
input {
margin-right: 10px;
}
App.js
File Edit View
import React from 'react';
import { BrowserRouter as Router, Route, Switch, Link } from 'react-router-dom';
import StreamList from './components/StreamList';
import Movies from './components/Movies';
import Cart from './components/Cart';
import About from './components/About';
import './App.css';
function App() {
return (
<Router>

<nav>

<Link to="/">StreamList</Link>
<Link to="/movies">Movies</Link>
<Link to="/cart">Cart</Link>
<Link to="/about">About</Link>

</nav>
<Switch>
<Route path="/" exact component={StreamList} />
<Route path="/movies" component={Movies} />
<Route path="/cart" component={Cart} />
<Route path="/about" component={About} />
</Switch>

</Router>
);
}
export default App;
Cart.js
File Edit View
import React from 'react';
const Cart = () => {
return (

<h1>Cart Page</h1>
This page will contain cart data in Week 4.

);
};
export default Cart;
Movies.js
File Edit View
import React from 'react';
const Movies = () => {
return (

<h1>Movies Page</h1>
This page will contain movie data in Week 4.

);
};
export default Movies;
StreamList.js
File Edit View
import React, { useState } from 'react';
const StreamList = () => {
const [inputValue, setInputValue] = useState('');
const handleInputChange = (e) => {
setInputValue(e.target.value);
};
const handleSubmit = (e) => {
e.preventDefault();
console.log(inputValue);
setInputValue('');
};
return (

<h1>StreamList</h1>
<form onSubmit={handleSubmit}>
<input
type="text"
value={inputValue}
onChange={handleInputChange}
placeholder="Add a movie or program"
/>
<button type="submit">Add</button>
</form>

);
};
export default StreamList;

index.html

File Edit View

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>StreamList App</title>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/react/17.0.2/umd/react.development.js" crossorigin></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/react-dom/17.0.2/umd/react-dom.development.js" crossorigin></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/babel-standalone/6.26.0/babel.min.js"></script>
    <style>
        body {
            font-family: Arial, sans-serif;
            text-align: center;
            margin: 20px;
        }
        nav {
            margin: 20px;
        }
        nav ul {
            list-style-type: none;
        }
        nav li {
            display: inline;
            margin: 0 15px;
        }
        input {
            margin-right: 10px;
        }
    </style>
</head>
<body>
    <div id="root"></div>
```

index.html

File Edit View

```
    <script type="text/babel">
        // StreamList Component
        const StreamList = () => {
            const [inputValue, setInputValue] = React.useState('');

            const handleInputChange = (e) => {
                setInputValue(e.target.value);
            };

            const handleSubmit = (e) => {
                e.preventDefault();
                console.log(inputValue);
                setInputValue('');
            };

            return (
                <div>
                    <h1>StreamList</h1>
                    <form onSubmit={handleSubmit}>
                        <input
                            type="text"
                            value={inputValue}
                            onChange={handleInputChange}
                            placeholder="Add a movie or program"
                        />
                        <button type="submit">Add</button>
                    </form>
                </div>
            );
        };
```

index.html

File Edit View

```
        // Movies Component
        const Movies = () => {
            return (
                <div>
                    <h1>Movies Page</h1>
                    <p>This page will contain movie data in Week 4.</p>
                </div>
            );
        };

        // Cart Component
        const Cart = () => {
            return (
                <div>
                    <h1>Cart Page</h1>
                    <p>This page will contain cart data in Week 4.</p>
                </div>
            );
        };

        // About Component
        const About = () => {
            return (
                <div>
                    <h1>About Page</h1>
                    <p>This page will contain information in Week 5.</p>
                </div>
            );
        };
```

index.html

File Edit View

```
        // Main App Component
        const App = () => {
            const [currentPage, setCurrentPage] = React.useState('streamlist');

            const renderPage = () => {
                switch (currentPage) {
                    case 'movies':
                        return <Movies />;
                    case 'cart':
                        return <Cart />;
                    case 'about':
                        return <About />;
                    default:
                        return <StreamList />;
                }
            };

            return (
                <div>
                    <nav>
                        <ul>
                            <li><button onClick={() => setCurrentPage('streamlist')}>StreamList</button></li>
                            <li><button onClick={() => setCurrentPage('movies')}>Movies</button></li>
                            <li><button onClick={() => setCurrentPage('cart')}>Cart</button></li>
                            <li><button onClick={() => setCurrentPage('about')}>About</button></li>
                        </ul>
                    </nav>
                    {renderPage()}
                </div>
            );
        };

        // Render the App component
        ReactDOM.render(<App />, document.getElementById('root'));
    </script>
</body>
</html>
```

References:

React Router Documentation. (n.d.). https://reactrouter.com/