Task 9:React Performance Optimization for a Profile Website

Objective:

Learn to optimize the performance of a React application by using `useMemo` and `useCallback`, and by leveraging the React Developer Tools Profiler for performance analysis. This task focuses on implementing these techniques in the context of a profile website.

Pre-requisites:

- Basic understanding of HTML, CSS, and JavaScript
- Familiarity with a code editor like Visual Studio Code
- Basic knowledge of React

Concepts Covered:

- Implementing useMemo and useCallback
- React Profiler for Performance Analysis

Concepts:

1. **Implementing useMemo and useCallback:**

   Use `useMemo` to memoize the profile list and `useCallback` for click handlers to optimize performance.

```
// src/components/ProfileList.js
import React, { useMemo, useCallback, useState } from 'react';

const ProfileList = () => {
    const profiles = [{...}, {...}, {...}]; // Profile data
    const memoizedProfiles = useMemo(() => profiles, []);
    const handleClick = useCallback(profile => { /* Handle click */ }, []);
    // Render logic
};
```

   Hints:

   - Use `useMemo` to memoize the array of profiles.
   - Use `useCallback` to memoize the click handler function.

2. **React Profiler for Performance Analysis:**

   Use the React DevTools Profiler to identify and optimize a performance issue in your application.

```
// src/components/ProfileList.js
import { Profiler } from 'react';

const onRenderCallback = (id, phase, actualDuration) => {
    console.log(id, phase, actualDuration);
};

return (
    <Profiler id="ProfileList" onRender={onRenderCallback}>
        {/* Render profile list */}
    </Profiler>
);
```

Hints:

- Wrap the component with the `Profiler` component.
- Use the `onRender` callback to log performance metrics.

Setup:

1. **Install Visual Studio Code (VS Code):**
   Download and install VS Code from Visual Studio Code.
2. **React Developer Tools:**
   Install the React Developer Tools browser extension for Chrome or Firefox.
3. **Git:**
   Install Git for version control. You can download it from Git.
4. **Node.js and npm:**
   Ensure Node.js and npm are installed on your machine. You can download and install them from Node.js.
5. **Setting Up the React Project:**
   - Open your terminal.
   - Navigate to the directory where you want to create your project.
   - Run the following command to create a new React project:

```
npx create-react-app profile-website
```

Tasks:

Part 1: Implementing useMemo and useCallback (30 minutes)

1. **Use `useMemo` to memoize the profile list and `useCallback` for click handlers to optimize performance:**
   - Example hint:

```
const memoizedProfiles = useMemo(() => profiles, []);
const handleClick = useCallback(profile => { /* Handle click */ }, []);
```

- Goal: Learn to optimize performance by memoizing expensive computations and functions.

Part 2: React Profiler for Performance Analysis (30 minutes)

1. **Use the React DevTools Profiler to identify and optimize a performance issue in your application:**
   - Example hint:

```
<Profiler id="ProfileList" onRender={(id, phase, actualDuration) =>
{ console.log(id, phase, actualDuration); }}>
    {/* Render profile list */}
</Profiler>
```

   - Goal: Gain hands-on experience with performance profiling and optimization in React.

Instructions:

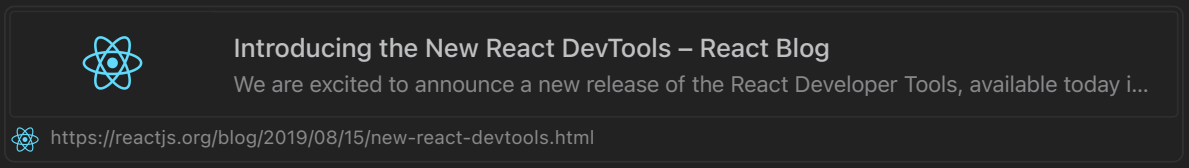
1. Write the required code in the appropriate files in the `src/components` directory.
2. Open the terminal and navigate to the project directory.
3. Run the React project using:

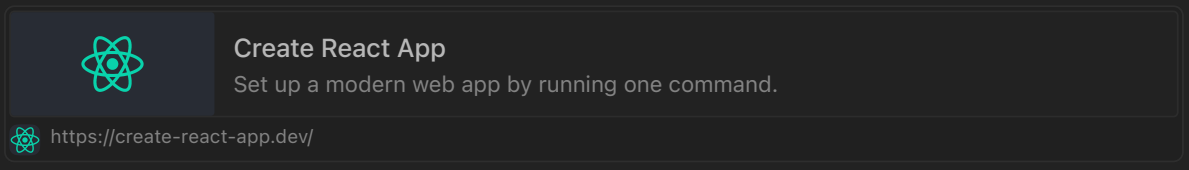
```
npm start
```

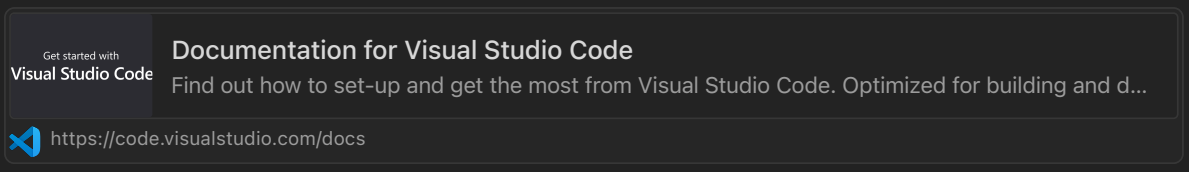
4. Open the project in a web browser to ensure the code displays correctly.
5. Use the browser's developer tools and React Developer Tools to debug and inspect the elements.

Resources:

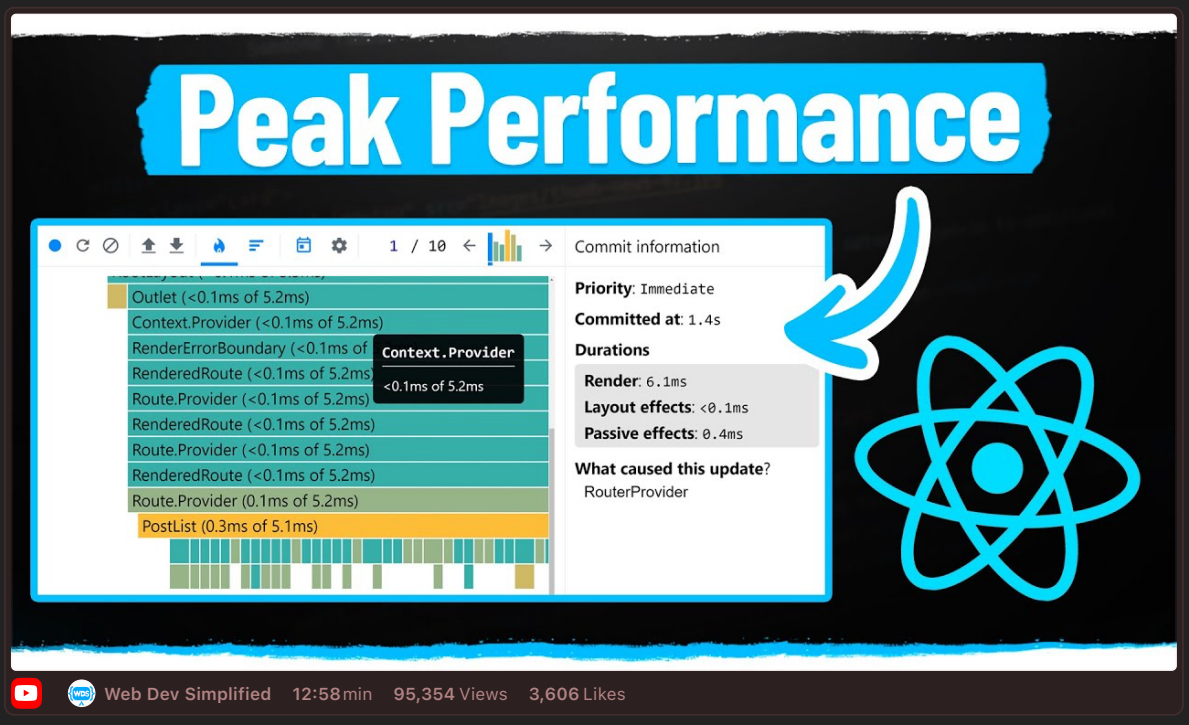
- 
  Hooks API Reference – React
  A JavaScript library for building user interfaces
  https://reactjs.org/docs/hooks-reference.html#usememo
- 
  Hooks API Reference – React
  A JavaScript library for building user interfaces
  https://reactjs.org/docs/hooks-reference.html#usecallback
- 
  Profiler API – React
  A JavaScript library for building user interfaces
  https://reactjs.org/docs/profiler.html

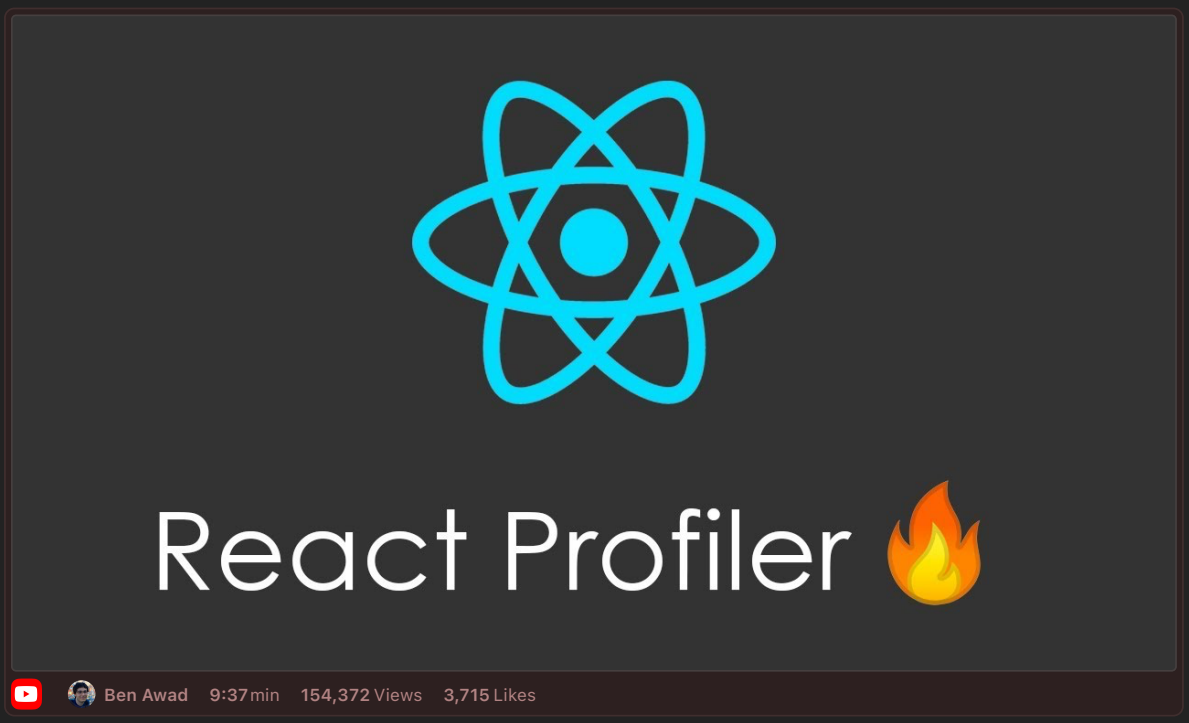
- 

- 

- 

Videos:

- 

- 

GitHub Instructions:

1. **Open in Visual Studio Code:**

   After clicking on the "Open in Visual Studio Code" button from the GitHub Classroom confirmation page, VSCode will open the repository directly. If prompted, select "Open" or "Allow" to open the repository in VSCode.

2. **Open the Terminal in VSCode:**

   In VSCode, open a terminal by selecting Terminal > New Terminal from the top menu.

3. **Complete the Task:**

   In VSCode, write your solution in the appropriate files in the `src/components` directory.

4. **Run and Test Your Code:**

   Open your terminal, navigate to your project directory, and run:

   ```
   npm start
   ```

5. **Commit Your Changes:**

   In the VSCode terminal, add your changes to git:

   ```
   git add src/components/*
   ```

   Commit your changes with a meaningful message:

```
git commit –m "Completed task 30"
```

6. **Push Your Changes to Your Repository:**

   Push your changes to your forked repository:

```
git push origin main
```

7. **Create a Pull Request:**

   Go to your repository on GitHub.

   Click on the "Pull Requests" tab.

   Click the "New Pull Request" button.

   Ensure the base repository is the original template repository and the base branch is `main`.

   Ensure the head repository is your forked repository and the compare branch is `main`.

   Click "Create Pull Request".

   Add a title and description for your pull request and submit it.

Summary of Commands:

```
# Open in Visual Studio Code

# Open terminal in VSCode

# Complete the task by editing src/components/*

# Navigate to the project directory
cd profile-website

# Run your code
npm start

# Add, commit, and push your changes
git add src/components/*
git commit –m "Completed task 9"
git push origin main

# Create a pull request on GitHub
```