Profiling Tool Documentation and Plan of Use

React Devtools

React devtools has two main functionalities that help with the development of a React application. Component tracking and performance analysis.

Component Tracking

- A tab for verifying a component's props and state. This tab displays a map of the components in the React app. It helps you to visualize the props and the current state each component in real time as you interact with them in your app.
- You can edit and test various props and edit the states in the React app straight from the component tab.
- It can show where the component is displayed when you hover overtop of the component in the component tab.
- The mapping of the components and their states/props can be logged onto the console.

Performance Analysis

- A tab for observing the rendering of components or large amounts of data. It exposes the parts of the application that might be slow or under-performing.
- The statistics are displayed in "commits" where there is an interaction with the app or when a new component is being rendered. Further details on what was rendered and how long it took to render can be found on the right side of the profiler. Details include: when the commit was made, what was being rendered during that commit, and how long it took for that component to render.

The plan to use React Devtools

1. Identify and document which components of the program are slowing down the performance of the app.
2. Use the results of the performance analysis to pinpoint and remedy/optimize the slow-performing components
3. Use the component tracker as a helper in fixing code that is not working as intended.
4. All documented statistics will be used as benchmarkers for future reference.

Example logging/documenting can look like:

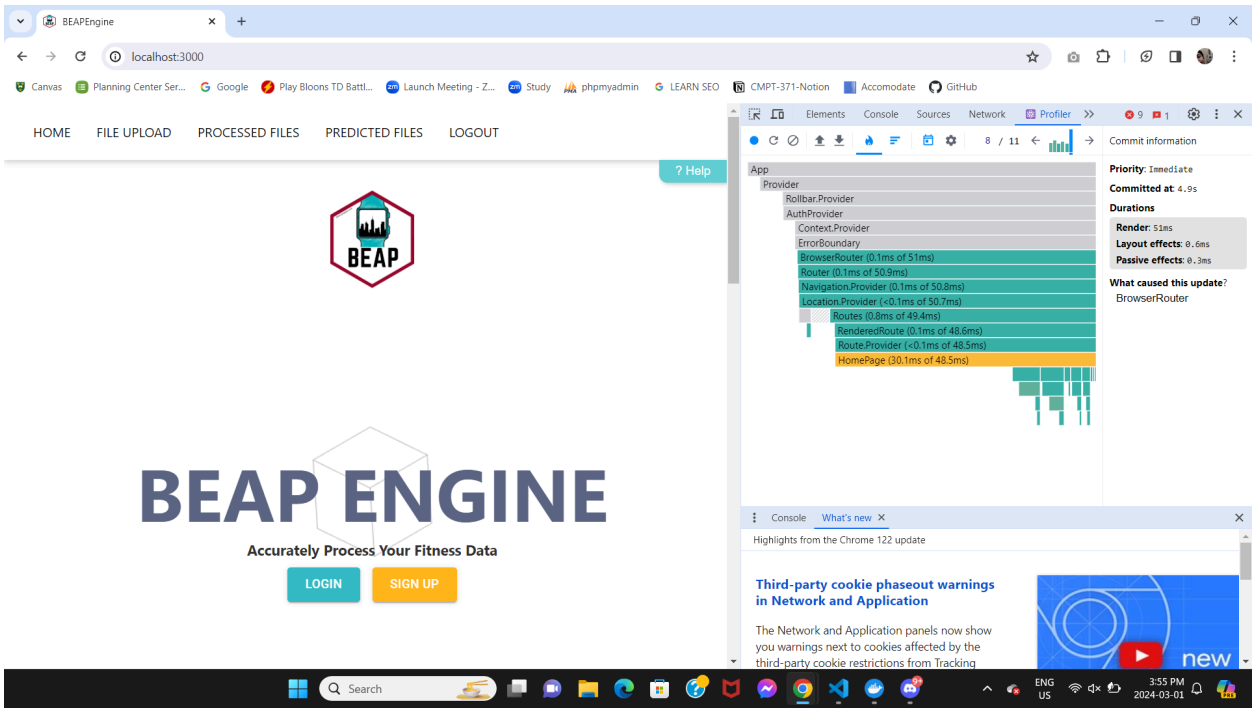
Slow rendering components: *Component 1, Component 2*
Component 1: [Description of when the component is rendered]

Initial solution: [A general direction on how to fix the problem]

Component 2: [Description of when the component is rendered]
Initial solution: [A general direction on how to fix the problem]

Steps on implementing React Devtools
1. Add the React Devtools extension on your browser
2. Run your React app and open your localhost
3. Right-click the app on the browser and open the "inspect" page
4. In the tabs, the new "Components" and "Profiler" tabs should appear.

Figure 1: Example of profiler results.



Other options considered:

AQTime 8
AQTime is a multi-purpose profiling toolkit that can be used to observe application performance, and detect unwanted behavior within your application.

Heavily integrated into Microsoft Visual Studio, where the optimal use of AQTime can be found.

-Will not be using AQTime because our primary coding environment is Visual Studie Code, not the Microsoft Visual Studio. The learning curve of using AQTime would not be worth it if we can just use React Devtools to satisfy our requirements.

References

Borrelli, Piero. *Getting Started with React DevTools in Chrome.* Debugbear, https://www.debugbear.com/blog/react-devtools

AQTime Documentation Getting Started. Smartbear, https://support.smartbear.com/aqtime/docs/tutorials/getting-started/index.html