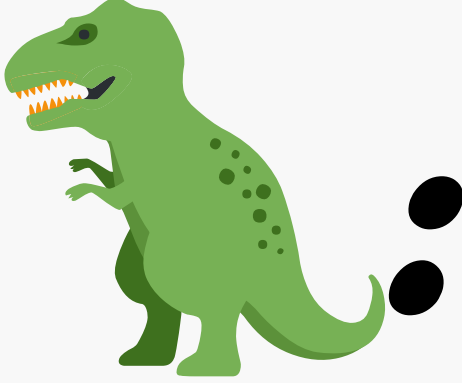
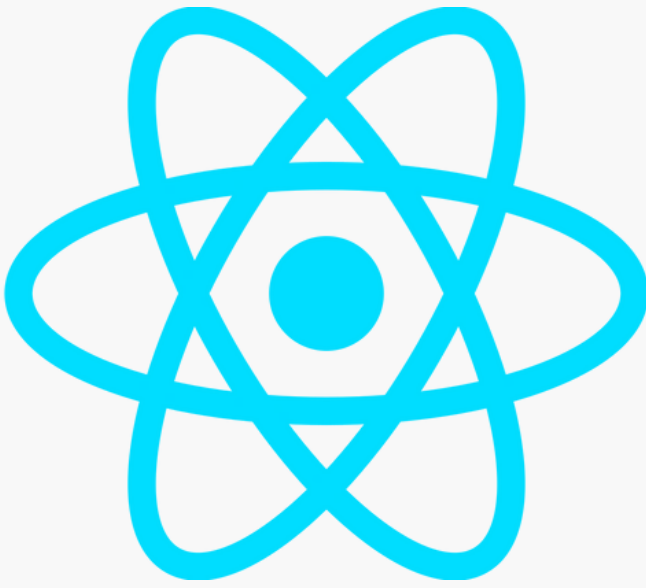

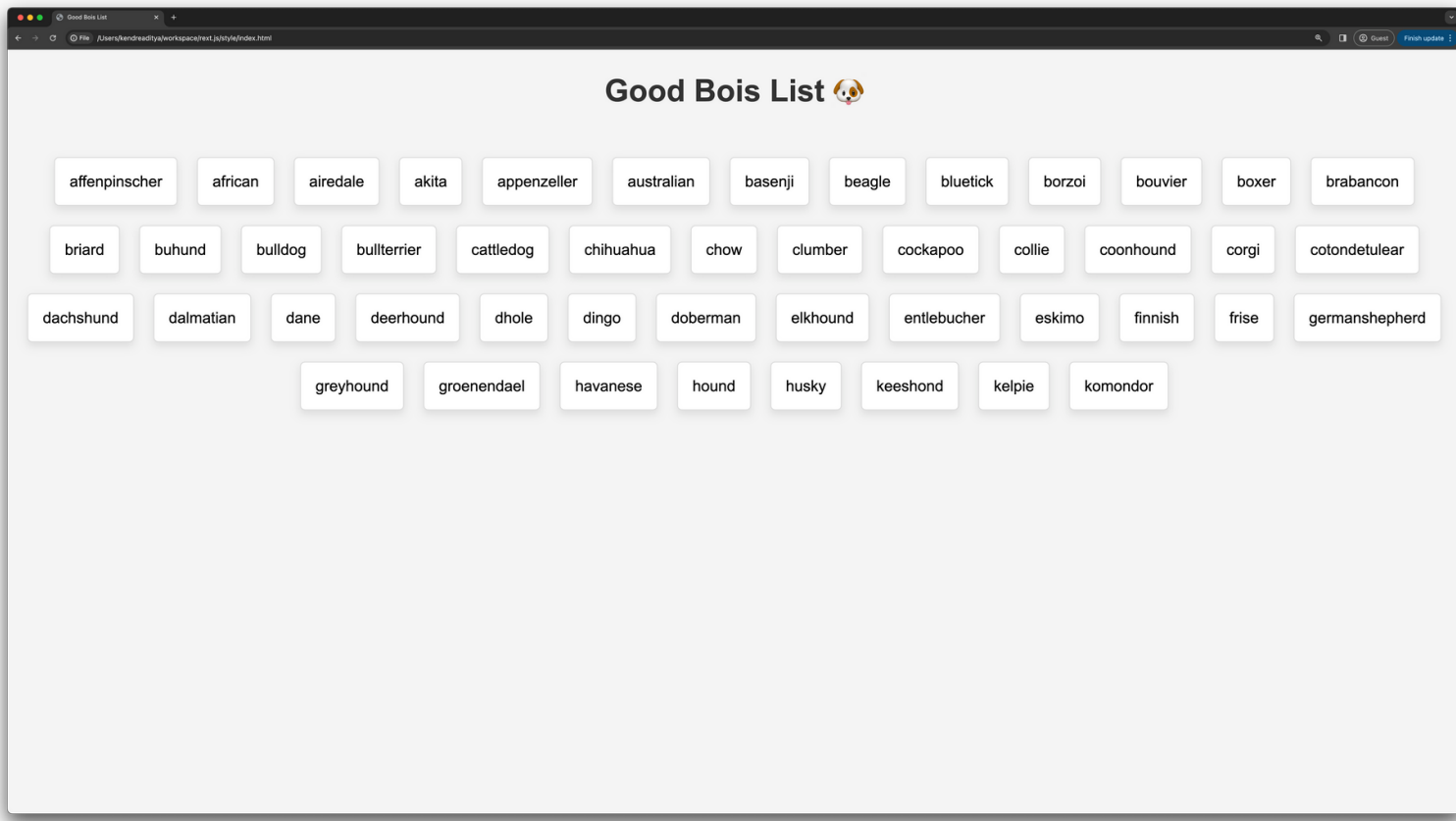
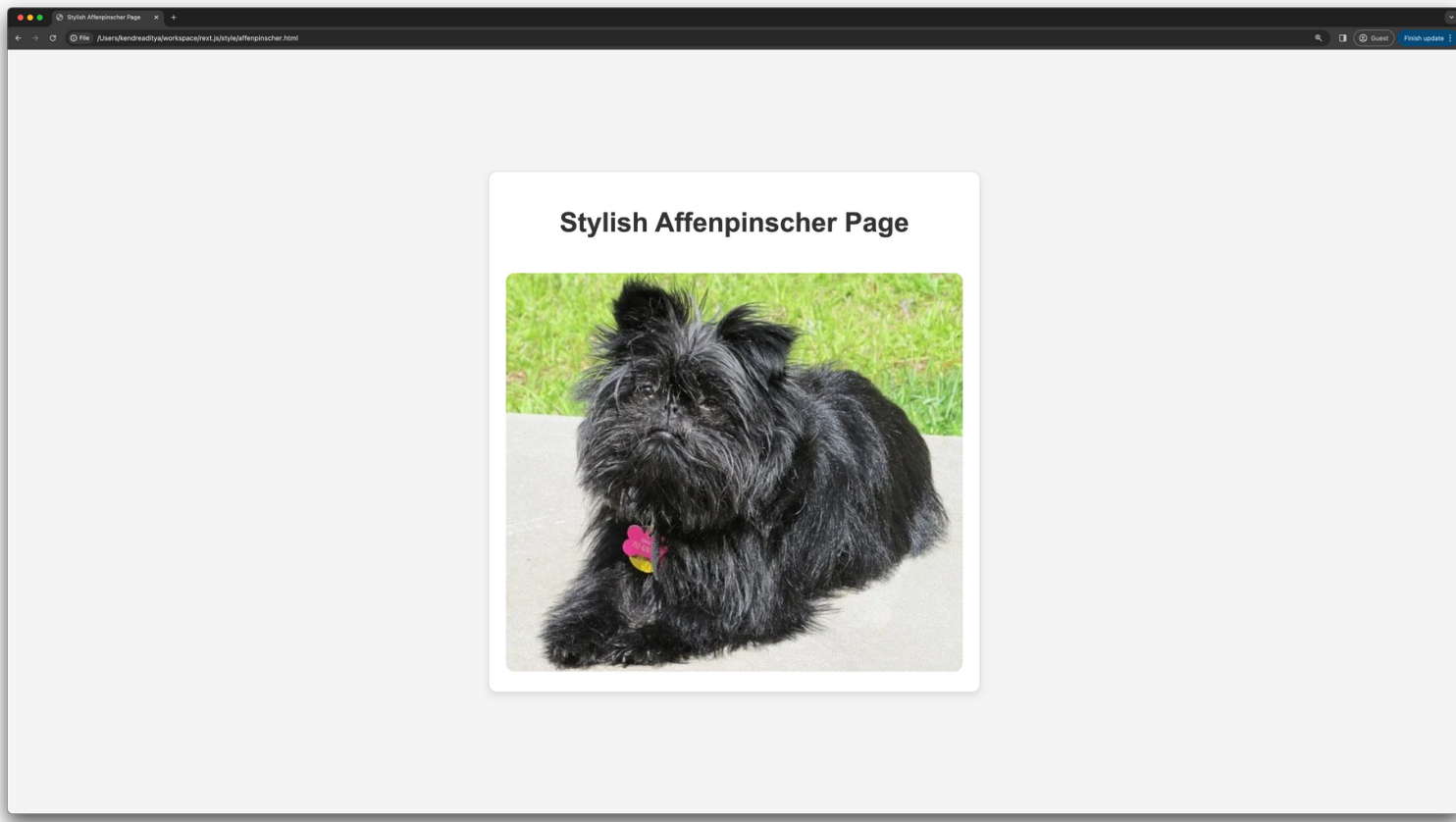


<div><div><div><div><div><div></div><div><h1>Rext.js </h1></div></div><div><h2>A Frontend Javascript Framework</h2></div></div></div></div></div>		Methodology	<p>React utilizes Virtual DOM, Hooks and Concurrent Mode are three fundamental concepts in the React framework that enable efficient and performant rendering of user interfaces. The Virtual DOM is a key concept in React that allows for faster and more efficient updates to the UI by creating a lightweight representation of the actual DOM. Hooks allow for better modularity and reusability of code by providing a way to manage state and side effects without using class components. Concurrent Mode is a new feature in React that allows for better performance by rendering components in parallel instead of sequentially, but it requires careful consideration when updating state and managing side effects.</p>
Introduction	<p>In the realm of web development, React.js and Next.js stand out as popular frameworks with distinct capabilities. This research project aims to deepen our understanding of these technologies by combining React and Next.js to create a makeshift framework, named Rext.js. The objective is to explore the integration of React's virtual DOM and reconciliation algorithm with Next.js' server-side rendering (SSR) and data fetching capabilities. This research seeks to provide insights into the inner workings of these frameworks and how they can be harmoniously employed to enhance web development practices.</p> <div><div></div><div></div></div> <div><div>Image 1. React - an open-source front-end JavaScript library for building user interfaces based on components maintained by Meta</div><div>Image 2. Next.js - an open-source web development framework created by Vercel.</div></div>		<p>Next.js employs side, server-side rendering to improve performance by pre-rendering the initial page on the server and reducing time to first byte (TTFB). Additionally, data is fetched on the server side before rendering to ensure that it's readily available when needed, minimizing reliance on client-side fetching and rendering. To further optimize performance, asynchronous data fetching using Suspense allows for precise control over when data is retrieved and processed, resulting in a more efficient user experience with fewer interruptions or delays caused by slow network requests or resource-intensive processing tasks running afoul of one another inside an individual user's session space within any given web app instance at any particular moment in time during its lifecycle from load up through shutdown or periodical refreshes as applicable across various touchpoint.</p>
Objective	<p>By combining the strengths of React and Next.js, the aim is to develop a unified framework that harnesses the advantages of both technologies. The specific goals encompass the implementation of server-side rendering, the incorporation of a virtual DOM with reconciliation capabilities, the utilization of functional components, and the implementation of robust server and client-side routing mechanisms. This synthesis aims to create a powerful framework that excels in web development by seamlessly blending the key features of React and Next.js.</p> <div>SSR + VIRTUAL DOM</div>	Findings	<p>Utilizing a Virtual DOM, proved instrumental in optimizing UI updates and rendering efficiency, by representing the UI as a virtual tree structure and employing a reconciliation algorithm, Rext.js successfully achieved efficient updates and minimized DOM manipulations. Additionally, the component-based JSX architecture showcased its versatility, allowing for a modular and reusable structure. This not only enhanced code organization but also facilitated the seamless integration of components into the Virtual DOM. The SSR implementation in Rext.js brought significant performance improvements. In rendering pages on the server and sending HTML to the client, initial page load times were reduced.</p> <div><div></div><div>Image 3. Rext.js based server-side rendered list of dog breeds</div><div></div><div>Image 4. Rext.js based page of a dog with server-side rendered image with file-based routing</div></div>