**Topics Learned During the Fellowship**

**Recap: Introduction to HTML, CSS, and JavaScript**

Revisited the basic building blocks of web development, including HTML for structuring web content, CSS for styling and layout, and JavaScript for adding interactivity and functionality to web pages.

**Recap: JavaScript before Starting React**

Revisited JavaScript concepts such as variables, data types, operators, conditionals, loops, and functions. Understanding these foundational concepts is essential for working with React.

**Git and GitHub**

Gained knowledge about version control using Git and how to collaborate on projects with others using GitHub. Git allows for efficient tracking of code changes, while GitHub provides a platform for hosting and sharing repositories.

**React Fundamentals: Introduction to React**

Introduced to React, a popular JavaScript library for building user interfaces. Learned about React's component-based architecture and the advantages it offers for creating reusable and modular UI components.

**React Fundamentals: Components**

Explored the concept of React components, which are self-contained, reusable building blocks that encapsulate UI elements and their logic. Components allow for a structured and scalable approach to building UIs.

**React Fundamentals: JSX Syntax**

Learned about JSX, a syntax extension for JavaScript that allows you to write HTML-like code within your JavaScript files. JSX simplifies the process of defining React components and rendering them to the DOM.

**React Fundamentals: Virtual DOM**

Gained an understanding of React's virtual DOM (Document Object Model) and how it optimizes rendering performance. The virtual DOM allows React to efficiently update only the necessary parts of the UI when state or props change.

**React Fundamentals: States and Props**

Explored React's state and props systems. State represents the internal data of a component, while props are used to pass data from a parent component to its child components, crucial for managing and updating component data.

**Made a Project of an Expense Tracker**

Developed an expense tracker application using React, where users can add, edit, and delete their expenses. This project allowed us to practice applying React concepts and managing component state.

**Handling Events**

Learned how to handle user events such as button clicks, form submissions, and keyboard interactions in React. Event handling enables interactivity and dynamic behavior in applications.

**Conditional Rendering**

Explored techniques for conditionally rendering components based on certain conditions or states. Conditional rendering allows to show or hide elements dynamically based on user input or other factors.

**Rendering Lists**

Learned how to dynamically render lists of data in React. Gained knowledge about iterating over arrays and rendering components for each item in the array, enabling to display dynamic content.

**Routing in React**

Explored routing in React using libraries like React Router. Routing enables the creation of multiple pages or views within a single-page application, allowing users to navigate between different sections of an application.

**React Hooks**

Learned about React Hooks, which are functions that allow to use state and other React features in functional components. Hooks provide a simpler and more concise way to manage component state and lifecycle events.

**API Integration**

Gained experience integrating external APIs into React applications. Learned how to make HTTP requests to fetch data from APIs and display the retrieved data in your UI.

**Custom Hooks**

Explored the concept of custom hooks, which are reusable functions that encapsulate logic and stateful behavior in React. Custom hooks allow for code reuse and abstraction, making code more modular and maintainable.

**Made a Pokemon Project**

Developed a project that utilized the Poki API to fetch and display data about Pokémon based on user search. This project allowed us to practice API integration and dynamic data rendering in React.

**Styling in React**

Learned various approaches for styling React components, including using styled components, CSS modules, Tailwind CSS, and Framer Motion. These tools and techniques enabled us to create visually appealing and interactive UIs.

**State Management Using Recoil**

Explored Recoil, a state management library for React. Recoil provides a simple way to manage and share state across components, making it easier to build complex and scalable applications.

**Forms in React: Validation and React Hook Form**

Gained knowledge about handling forms in React, including form validation techniques and using libraries like React Hook Form. Also, learned how to validate user input and handle form submissions effectively.

**Made a Project of Personal Portfolio**

Developed a personal portfolio project using React, showcasing skills, projects, and experience. This project allowed us to demonstrate our understanding of React concepts while creating a professional online presence.

**Firebase Integration**

Learned how to integrate Firebase, a backend-as-a-service platform, with React applications. Firebase provides features such as real-time database, authentication, and cloud functions, enabling us to build powerful and dynamic applications.

**Formal Introduction to Next.js**

We were formally introduced to Next.js, a popular framework for server-rendered React applications. Next.js simplifies the process of building server-side rendered and statically generated React apps, providing benefits like improved performance and SEO.