

Project Documentation

Route Setup

Page routing was initiated by using the React Router (react-router-dom v6). Each of the websites major page in the application is mapped to its own route using `<Routes>` and `<Route>` components. A layout structure is used for the navigation and footer bar to remain visible across all pages. A reusable component called, `PartPage`, is used to display all individual part pages and receives its specific content through props that then loads content to the specific page.

Passing Data via Props

Data was passed through components by using props, one of the best examples for this is the `partsData.js` file as it stores all information related to each JoJo character, images, information, and part. This data is passed into the `PartPage` component via the `partId` prop, which is then used to select the appropriate object from `partsData`. These were also used in components for cards, modals, carousels, and accordions to pass descriptions and content information.

React Hooks

The `useState` hook was primarily used to manage UI state like opening and closing modals and switching carousel. Looking in the About page, clear usage of this hook can be seen with clicking on card icons that pop up character information. Though these hooks can also be seen to be used for carousel's as it tracks the image in current use or rotation.

Hamburger Menu

A hamburger menu was implemented in the `Navbar.js` file for a more responsive design for mobile users or smaller screens in general. To track whether the menu was open or not, a state variable is used and toggled to display the navigation links when clicked.

Component Relations

The application was structured around many reusable components to maintain readability, performance, and organization. Some core components seen is the BackgroundLayout, Navbar, Header, and Footer which wraps around the page content. These components are smaller and are mainly used for certain pages only such as cards, accordions, carousels, and modal manage the UI functionality. Separation of logic and presentation can be seen within the data files as it feeds content stored into components, making it much easier to change topics.

Styling Approach

Both types of styling, Tailwind and custom CSS files were used throughout the project. Tailwind can be seen more within the spacing, typography, layout and implementing responsiveness with grids and flex. Meanwhile custom CSS files were used for background/wallpaper design, overlays, and navigation styling which contains far more content needed to be applied.

Difficulties, Errors, and Solutions

Several challenges were encountered during the developmental stage. Issues such as background images cutting off due to height limitations, though that was solved with CSS styling. Though struggle came when routing issues occurred as pages failed to load properly, which was solved by installing a specific version of react router (`react-router-dom@6`) and then restarting the server. Another issue came as App.js was rendered twice, but that was a minor issue that was easily resolved by removing an unneeded import from index.js.