**# Data Flow Diagram (DFD) for MyRouterApp**

## **Level 0: Context Diagram**

### **Overview:**

The **MyRouterApp System** is a web-based application that handles user navigation between multiple pages using React Router. It processes user requests, dynamically updates content, and ensures a smooth user experience.

**Entities & Processes:**

* **User:** Navigates between different routes (pages).
* **Router System (Process 1.0):** Manages URL changes and renders appropriate components.
* **Component Data Store (Data Store):** Stores the data used by different pages.

+-----------------------+

| User |

+-----------------------+

|

v

+-----------------------+

| Router System |

| (Process: 1.0) |

+-----------------------+

|

v

+-----------------------+

| Component Data Store |

+-----------------------+

## **Level 1: Detailed Breakdown**

### **Processes Breakdown:**

1. **Process 1.1 - User Requests a Page:** The user clicks a link or enters a URL.
2. **Process 1.2 - Router Matches Route:** The app checks the requested URL and determines the appropriate component to render.
3. **Process 1.3 - Load Component Data:** If necessary, the system fetches data required for the page.
4. **Process 1.4 - Render Page:** The requested page component is displayed to the user.

+-----------------------+

| User |

+-----------------------+

|

v

+----------------------------+

| Process: 1.1 - Request Page|

+----------------------------+

|

v

+----------------------------+

| Process: 1.2 - Match Route |

+----------------------------+

|

v

+----------------------------+

| Process: 1.3 - Load Data |

| (Component Store / API) |

+----------------------------+

|

v

+----------------------------+

| Process: 1.4 - Render Page |

+----------------------------+

### **Data Flow:**

1. The **User** navigates through the application (clicks links, types URL, etc.).
2. The **Router System** checks for a matching route and determines which page component should be displayed.
3. If needed, the system fetches relevant **data** from the **Component Data Store or API**.
4. The appropriate **page is rendered**, providing a seamless user experience.

**Benefits:** Enables dynamic routing without page reloads. Ensures efficient navigation and data handling. Provides a scalable architecture for React applications.