**Data Flow Diagram (DFD) for Theme Management App**

**Theme Management App** allows users to toggle between **light mode** and **dark mode**, and it also includes a shopping cart system. Below is the **DFD representation**:

**Level 0 (Context Diagram)**

At the highest level, the system consists of:

1. **External Entity: User** → Clicks the theme toggle button to switch themes.
2. **Process: Theme Management System** → Manages and updates the theme across the app.
3. **Data Store: Local Storage & React State (useState Hook)** → Stores the selected theme and persists it.

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

| External Entity: |

| User |

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

|

v

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

| Theme Management | (Process 1.0)

| System |

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

|

v

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

| Local Storage & | (Stores theme state)

| useState Hook |

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

**Level 1 DFD (Decomposed Processes)**

Breaking down **Process 1.0 (Theme Management System)** into detailed steps:

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

| External Entity: | | Process 1.1 - User |

| User | -----> | Clicks Theme Button |

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

| |

v v

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

| Process 1.2 - | | Process 1.3 - Update |

| Capture Event | -----> | (Change Theme State) |

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

| |

v v

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

| Process 1.4 - | | Process 1.5 - Persist |

| Store Theme | -----> | (Save to LocalStorage)|

| (useState Hook) | | (Light/Dark Mode) |

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

|

v

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

| Process 1.6 - |

| Apply Theme |

| (Change UI Style)|

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

**Explanation of Level 1 DFD Processes**

1. **Process 1.1 - User Clicks Theme Button**
   * The user clicks the **"Toggle Theme"** button.
2. **Process 1.2 - Capture Event**
   * The button click triggers an event handled by the ThemeToggle component.
3. **Process 1.3 - Update Theme State (useState)**
   * The setDarkMode function updates the theme between "light" and "dark".
4. **Process 1.4 - Store Theme in State (useState)**
   * The updated theme is stored in React state to reflect changes instantly.
5. **Process 1.5 - Persist Theme in Local Storage**
   * The selected theme is saved in localStorage so that it persists across page reloads.
6. **Process 1.6 - Apply Theme to UI**
   * The theme state is used to apply **light mode** or **dark mode** styles dynamically.

**Data Flow Summary**

✔ **User clicks a button → System captures event → Updates theme state → Saves to local storage → Applies new theme**  
✔ Uses **React state (useState) and localStorage** for real-time updates and persistence.  
✔ **No backend or database**—only front-end state management with Context API and localStorage.