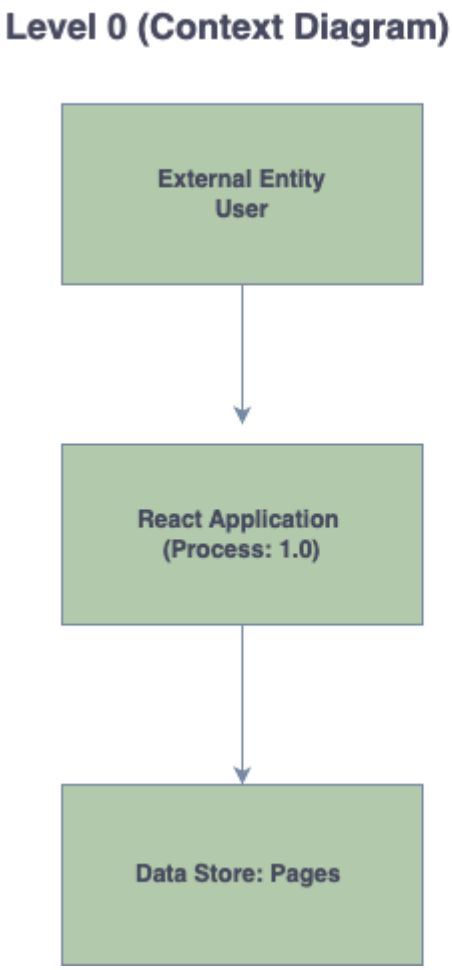
React App with ThemeProvider and Lazy Loading

Level 0 (Context Diagram)

At the highest level, the application acts as a single process handling theme management and lazy loading components.



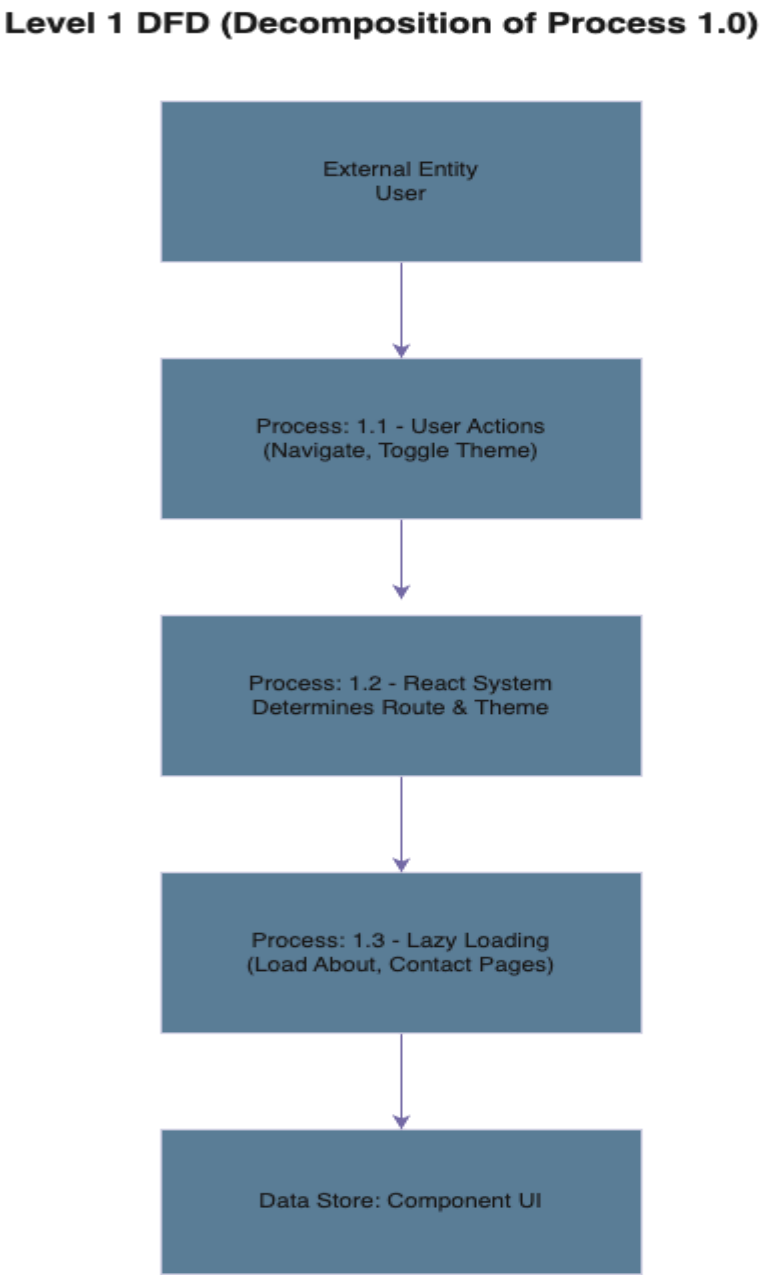
Explanation:

- **User**: The user interacts with the application by navigating between pages and toggling the theme.
- **React Application**: The core system that manages themes and lazy loads components.

- **Data Store (Pages)**: The different pages (Home, About, Contact) are dynamically loaded as React components.

Level 1 DFD (Decomposition of Process 1.0)

Breaking down the navigation and theme switching system further:

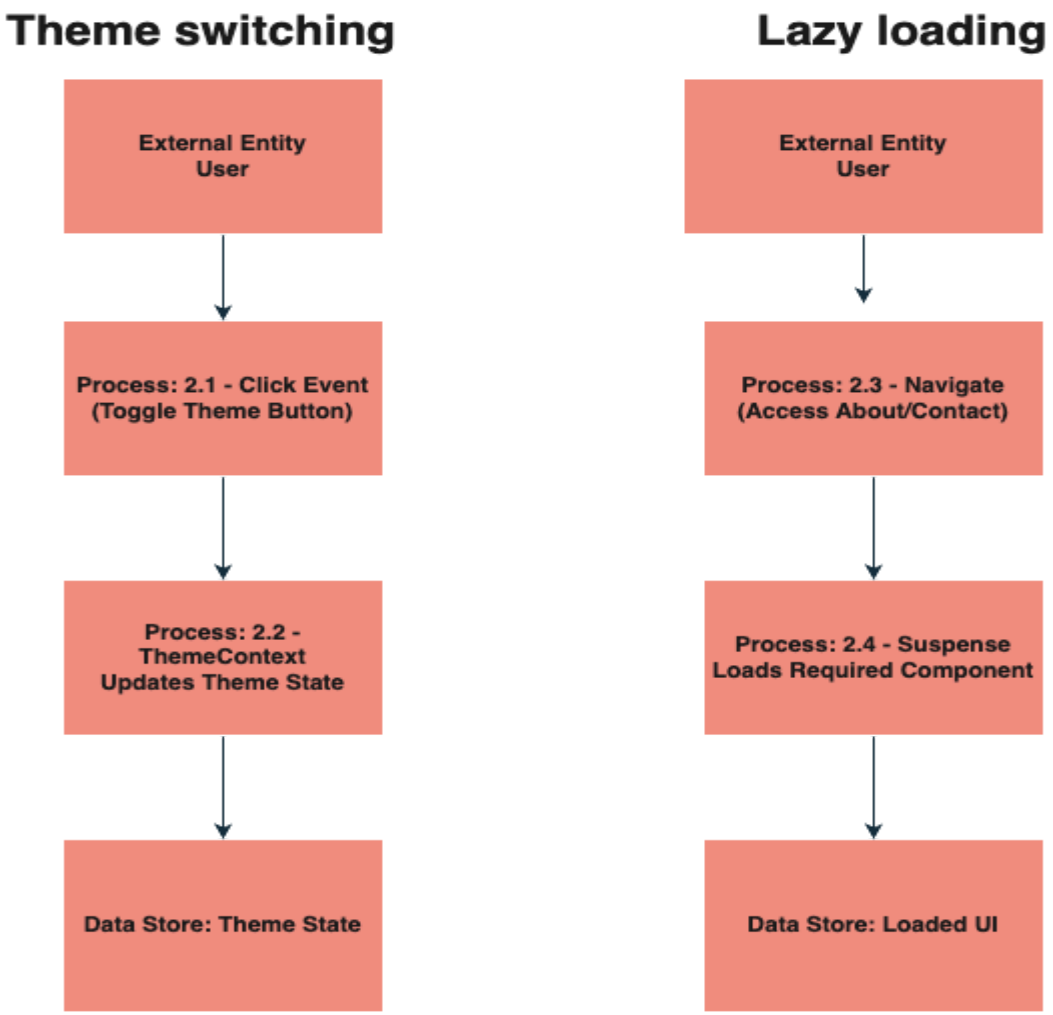


Explanation:

- **Process 1.1 (User Actions)**: The user interacts with the application by navigating and toggling themes.
- **Process 1.2 (React System Determines Route & Theme)**: React processes the user actions and updates the UI accordingly.
- **Process 1.3 (Lazy Loading About, Contact Pages)**: Suspense is used to dynamically load About and Contact components when required.
- **Data Store (Component UI)**: The components are rendered based on the theme and navigation.

Level 2 DFD (Detailed Breakdown of Theme Switching & Lazy Loading)

If we further break down theme switching and lazy loading:



Explanation:

- **Process 2.1 (Click Event - Toggle Theme)**: User clicks the toggle theme button.
- **Process 2.2 (ThemeContext Updates Theme State)**: The theme state is updated and applied across components.
- **Process 2.3 (Navigate to Lazy Loaded Component)**: User requests About or Contact component.
- **Process 2.4 (React Suspense Loads Component)**: Suspense loads the required component dynamically.
- **Data Stores**:
    - **Theme State**: Stores the current theme.
    - **Loaded UI**: Stores rendered components after lazy loading.

This DFD represents the key data flow elements of your React application, focusing on:

- User interactions with theme toggling and page navigation.
- How React Suspense manages lazy loading of components.
- How ThemeContext updates and maintains the theme across components.