The provided code is a React functional component named `Dashboard`, which serves as the parent component for a dashboard interface. Here's a breakdown of its functionality:

1. \*\*Imports\*\*: The component imports various modules and components necessary for its functionality, including React, useState, useEffect, useCallback, axios, and several others.

2. \*\*State Management\*\*: The component manages several state variables using the `useState` hook:

- `dataEx`: Holds data retrieved from an API.

- `user`: Holds user data.

- `dataType`: Holds data type information.

- `windowWidth`: Holds the width of the browser window.

3. \*\*Side Effects with useEffect\*\*:

- The component utilizes the `useEffect` hook to perform side effects:

- The first `useEffect` is called when the component mounts and checks if the user is logged in by calling the `isLoggedIn` function.

- The second `useEffect` is responsible for updating the `windowWidth` state variable and adding a resize event listener to adjust the width when the window is resized. It cleans up the event listener on unmount.

4. \*\*Event Handlers\*\*:

- `handleResize`: Updates the `windowWidth` state variable when the window is resized.

5. \*\*Functions\*\*:

- `isLoggedIn`: An asynchronous function responsible for checking if the user is logged in. It makes API calls to retrieve user and payment details. If successful, it updates the `user` state with the received data. If the user is not found or there is an error, it displays a toast message and navigates the user to the appropriate page.

6. \*\*Conditional Rendering\*\*:

- The component conditionally renders different content based on the `windowWidth` and `user` state variables. If the window width is less than 1050 pixels, it displays a message prompting the user to view on a bigger screen. If the user data is not available, it displays a loading spinner.

7. \*\*Context Providers\*\*:

- The component wraps its children components with `userContext.Provider` and `DataContext.Provider`, passing down `user` and data-related values as context.

8. \*\*Return Statement\*\*:

- The component returns JSX representing the dashboard interface, including a sidebar, the main content (`Outlet`), and a toast container for displaying notifications.

Overall, this component manages user authentication, window resizing, API data fetching, and conditional rendering of dashboard content. It utilizes hooks like `useState`, `useEffect`, and `useCallback` for state management and side effects. Additionally, it leverages React context for passing down user and data-related information to its children components.