**Toast Notification Messages**

In web development, a "toast" refers to a small notification message that appears on the screen to provide feedback or information to the user. These notifications are usually displayed for a short duration and then disappear automatically, although they can also be dismissed by the user. Here are some key characteristics and usage scenarios for toasts in web development:

1. **Temporary Display:** Toast notifications are typically displayed for a few seconds before they fade out or slide away, although the duration can be customized based on the use case.
2. **Non-blocking:** Unlike modal dialogs or alert boxes, toast notifications are non-blocking. They allow the user to continue interacting with the application while the notification is displayed.
3. **Informative Feedback:** Toasts are often used to provide feedback on an operation, like confirming that an action has been completed successfully (e.g., "Your settings have been saved.").
4. **Error Messages:** They can also be used to display error or warning messages, although critical errors might warrant more attention-grabbing UI elements.
5. **Positioning:** Toast notifications are often positioned at the top or bottom of the screen, either in the center or at the corners, so they’re noticeable without obstructing the main content.
6. **Styling:** The styling of toast notifications can be customized to match the branding and design of the application. They can also contain icons, links, or buttons for additional interactivity.
7. **Accessibility:** Good toast implementations should be accessible and provide the necessary ARIA roles and attributes to ensure they're perceivable by screen readers and other assistive technologies.

Several libraries and frameworks offer pre-built toast components, such as Bootstrap's Toast component, or dedicated libraries like react-toastify for React applications.

In this example, I'll demonstrate how you can create a simple toast notification in React without using any external libraries. We'll create a component called **Toast** and a component called **App** to trigger the toast notification.

1. **Create a new CSS file (Toast.css) for styling:**

.toast {

position: fixed;

bottom: 20px;

right: 20px;

background-color: #333;

color: white;

padding: 10px 20px;

border-radius: 4px;

transition: opacity 0.5s, transform 0.5s;

opacity: 0;

transform: translateY(100%);

}

.toast.show {

opacity: 1;

transform: translateY(0);

}

1. **Create a new file (Toast.js) for the Toast component:**

// Toast.js

import React from 'react';

import './Toast.css';

function Toast({ message, visible }) {

return (

<div className={`toast ${visible ? 'show' : ''}`}>

{message}

</div>

);

}

export default Toast;

1. **Create a new file (App.js) for the App component:**

// App.js

import React, { useState } from 'react';

import Toast from './Toast';

function App() {

const [toastVisible, setToastVisible] = useState(false);

const showToast = () => {

setToastVisible(true);

setTimeout(() => {

setToastVisible(false);

}, 3000); // Hide toast after 3 seconds

};

return (

<div>

<button onClick={showToast}>Show Toast</button>

<Toast message="This is a toast message!" visible={toastVisible} />

</div>

);

}

export default App;

In this setup:

* Clicking the "Show Toast" button sets **toastVisible** to **true**, which triggers the **Toast** component to display.
* A **setTimeout** function is used to hide the toast after 3 seconds by setting **toastVisible** back to **false**.
* The **Toast** component receives **message** and **visible** props to control the displayed message and visibility.
* The CSS transition property provides a simple slide-and-fade animation when the toast is shown and hidden.Top of Form