**Root Cause Analysis**

1. Customer Details:

Name: [Insert customer name]

Email Address: [Insert customer email address]

Product: [Insert name of product customer was trying to purchase]

Plan: [Insert name of plan customer had selected for the product]

Issue:

The issue that the customer faced while adding the product to the cart was caused by a conflict between two JavaScript codes triggering at the same time.

1. Date of occurrence:

The issue was reported on [Insert date of occurrence].

1. How the issue occurred:

The conflict occurred because the website's jQuery version or other JavaScripts were using similar function libraries, causing the two JavaScript codes to modify the same resource at the same time.

1. Reason:

The issue was caused by a race condition in the JavaScript code. A race condition occurs when two or more processes are trying to access or modify a shared resource at the same time. In this case, the two JavaScript codes were trying to modify the same resource, causing a conflict.

1. Solution:

To resolve the issue, a patch was added to the patc to prevent any future conflicts. The patch was designed to use a debouncing technique methods to ensure that the JavaScript code triggered only when the object needed and if already other object used this than wait and take their results, even if the user clicked the button multiple times or other javascript triggered/using the resource. The debouncing technique is a common solution used to prevent conflicts between JavaScript codes. The patch was implemented to work as a standalone solution to avoid any conflicts with existing JavaScript and future modifications.

1. Preventive measures:

To prevent this issue from occurring in the future, the patch was made standalone on the page. Additionally, the development team is working to remove the Piwik dependency, a web analytics platform previously used on the website, that was causing conflicts with the JavaScript code from existing piwik or similar code.

1. Additional comments:

The solution implemented was specific to the Lapp team and required multiple testing to ensure its efficacy. It's important to note that race conditions can be tricky to identify and resolve, especially in complex JavaScript applications. Therefore, it's essential to test thoroughly and ensure that the solution implemented does not cause any unintended consequences.

~