The error message "TypeError: Cannot read properties of null (reading 'label')" suggests there's an issue with attempting to access the `label` property of an object that is, at some point, `null`. This kind of error typically occurs when the code expects an object to be present but doesn't explicitly check for its existence before trying to access its properties.

The primary focus should be on areas where the `.label` property is being accessed. Given the context of the error happening when deleting text, this issue is most likely related to how the autocompletion and tooltip functionalities are implemented in `page.js` and `tooltip.js`.

In `page.js`, there's logic that dynamically updates the options for autocompletion based on the current input. When text is deleted, the code attempts to find a matching completion object based on the reduced input and access its `.label` property. If no matching object is found due to the deletion of text, attempting to access the `.label` of `undefined` (or `null`) will throw the error you've encountered.

Similarly, in `tooltip.js`, there's logic to display tooltips based on cursor position and input text. If the code assumes it will always find a matching completion and then tries to access its `.label` property without verifying that the completion exists, it could also result in the error.

Here are some specific points in your code that could lead to such errors:

1. page.js onChange Handler:

- The logic in the `onChange` handler, especially where it processes autocompletion and parameter entry, might attempt to access `.label` on a completion object that does not exist. This could happen in the `options.find()` call if no matching completion is found. Adding checks before accessing properties of the found object could prevent this error.

2. Accessing Previous Options:

- The code maintains a `prevOptions` state but seems to rely on the existence of a "best match" without confirming that such a match exists when deleting text. Ensuring that `bestMatchObject` is not null before trying to build a new options array with it could solve the issue.

3. ToolTips and Completions Handling:

- In `tooltip.js`, the `getCursorTooltips` function might not always find a matching completion. Ensuring that `find` and the eventual access to `find[0].label` are guarded against `null` values is crucial.

To address the issue, consider adding null checks or using optional chaining (`?.`) when accessing properties that might not exist on an object. For example, instead of directly using `bestMatchObject.label`, you could use `bestMatchObject?.label` and then check if the value is `undefined` before proceeding. This approach should help prevent the error when the expected object is not found due to deletions or otherwise.