## **BUG REPORT DOCUMENTATION FOR BROKEN LOGIN BUTTON**

**Bug ID:** Bug001

**Bug Title:** Login Form Not Responding After Clicking Submit

**Environment:**

* Operating System: Windows 10
* Browser/App Version: Chrome 115
* Device Type: Desktop

**Preconditions:** User is on login page and login page is fully loaded.

**Severity:** Critical — Users cannot log in to the system.

**Priority**: High — This issue prevents users from accessing their accounts.

**Steps to Reproduce:**

1. Open the login page.
2. Enter a valid email address and password.
3. Click the Login button.
4. Observe: Nothing happens — the page does not reload or redirect.

✅ **Expected Result:**

* The user should be successfully logged in and redirected to their account dashboard.

❌ **Actual Result:**

* Clicking the Login button does nothing.
* No error message is displayed.
* No indication of form submission, failure, or success.

**Impact on Users:**

* This affects 100% of users trying to log in on desktop Chrome, leading to frustration and potential churn, especially for time-sensitive services.

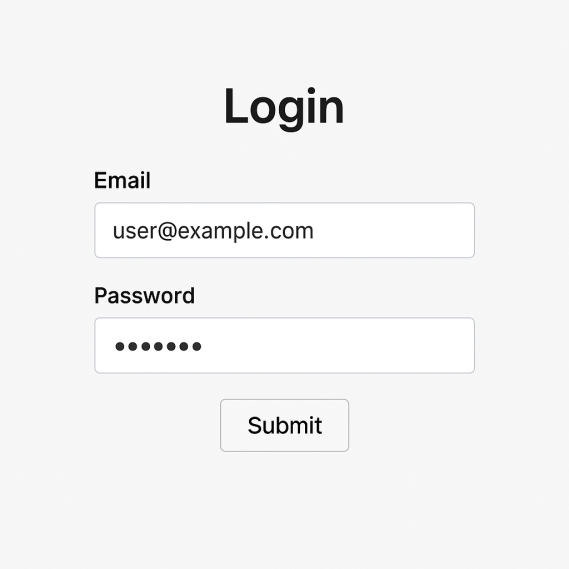
**Assignee:**

* Assign to: Frontend Development Team

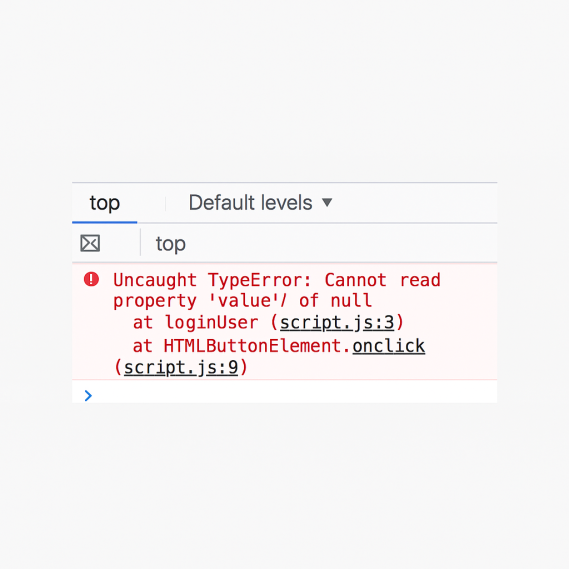
**Status:**

* To Do

**Screenshot of the form after clicking submit with no response.**

****

**Screenshot of Console errors.**

****

**Recommended Fix**

**1. Inspect DOM Element Bindings**

* Ensure that the Submit/Login button is properly bound to a JavaScript event handler (e.g., onclick, or using .addEventListener).
* If using a framework like React or Vue, confirm the event handler function (e.g., handleLogin) is properly passed and bound.
* Fix Example (Vanilla JS):

javascript

Copy

Edit

document.getElementById("loginBtn").addEventListener("click", function () {

loginUser();

**2. Check Input Element IDs/Selectors**

* Ensure the form fields (email, password) are correctly referenced in the script. A common error is querying elements that don't exist due to typo or missing id.
* Fix Example:

javascript

Copy

Edit

const email = document.getElementById("email").value;

const password = document.getElementById("password").value;

Validate that the form inputs have correct id attributes in the HTML.

**3. Validate API Call or Form Submission**

* Ensure the login request is being sent (via fetch, axios, or form.submit()).
* Fix Example (Using fetch):

javascript

Copy

Edit

fetch("/api/login", {

method: "POST",

headers: { "Content-Type": "application/json" },

body: JSON.stringify({ email, password })

})

.then(res => res.json())

.then(data => {

if (data.success) {

window.location.href = "/dashboard";

} else {

alert("Login failed.");

**4. Add Error Handling + Feedback**

* Always show user feedback when:
* Fields are empty.
* API fails.
* Credentials are incorrect.
* Fix Example:

javascript

Copy

Edit

if (!email || !password) {

alert("Please enter both email and password.");

return;

**5. Ensure Network Request Is Not Blocked**

* Check the browser dev tools to ensure the request isn’t blocked due to:
* CORS policy issues
* Invalid/missing CSRF token
* Incorrect API endpoint

**6. Test the Fix**

* After applying changes:
* Test with valid and invalid credentials.
* Check behavior across browsers.
* Use browser console and network tab to ensure request is firing and handled properly