Mismatched First and Last Names in API and UI

Steps to reproduce:

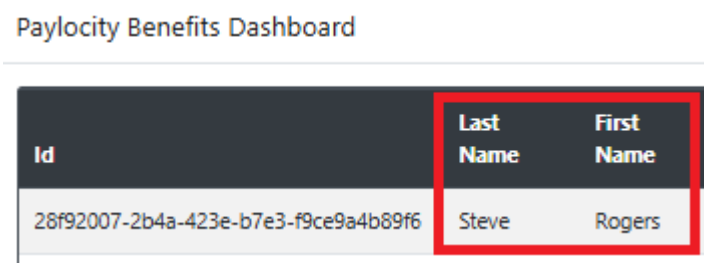
1. Go to Benefits Dashboard page
2. Look at the LastName and FirstName (i.e. LastName: Steve and FirstName: Rogers)
3. Open the browser developer tools usually accessible via F12 or right-click > Inspect
4. Go to the Network tab within the developer tools
5. Refresh the application page to capture all network requests
6. Go to employees request and go to Response tab, look for same name you was looking at UI

Actual Results:

In the UI table, the names are displayed as:

Last Name: "Steve"

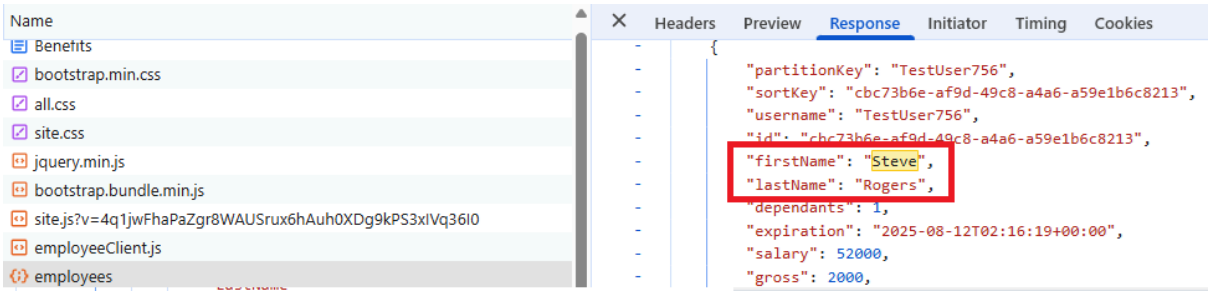
First Name: "Rogers"



In the API response, the names are:

First Name: "Steve"

Last Name: "Rogers"



Expected:

The first and last names should be consistent between the API response and the UI table.

Duplicate Employee Entries Allowed in Benefits Dashboard Page

Steps to reproduce:

1. Go to Benefits Dashboard page

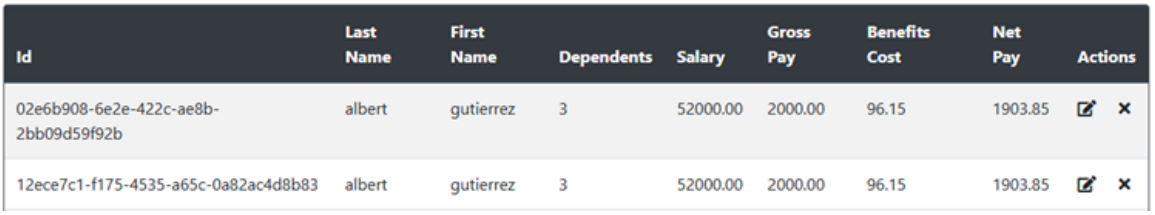
2. Click on “Add Employee” button

3. Enter the details of an employee, including Last Name, First Name and Dependents that match an existing employee

4. Click on Add button

Actual Results:

The system successfully adds the employee, resulting in duplicate entries with identical "First Name," "Last Name," and "Dependents" in the employee list



Id	Last Name	First Name	Dependents	Salary	Gross Pay	Benefits Cost	Net Pay	Actions
02e6b908-6e2e-422c-ae8b-2bb09d59f92b	albert	gutierrez	3	52000.00	2000.00	96.15	1903.85	
12ece7c1-f175-4535-a65c-0a82ac4d8b83	albert	gutierrez	3	52000.00	2000.00	96.15	1903.85	

Expected Results:

Checking for existing entries with the same "First Name," "Last Name," and "Dependents" before allowing a new entry.

Displaying an error message to the user indicating that an employee with the same details already exists.

Missing Indicators for Required Fields: First Name, Last Name, and Dependents

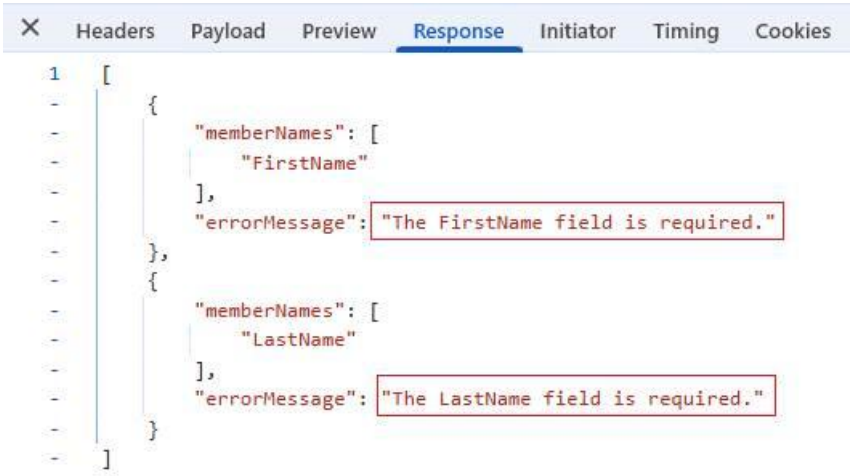
Steps to reproduce:

1. Go to Benefits Dashboard page
2. Click on Add Employee button
3. Enter dependents between 0 and 32
4. Click on Add

Actual Results:

After clicking the Add button, the form does not submit, and no feedback is provided to the user.

A review of the network activity reveals an error message indicating that the "First Name" and "Last Name" fields are required.



Expected Results:

The application should clearly indicate that First Name, Last Name, and Dependents are required fields.

Required fields should be marked with an asterisk (*) or labeled with the word Required next to the field label.

If a user attempts to submit the form without filling in required fields, the application should display an error message near the fields, informing the user of the missing information.

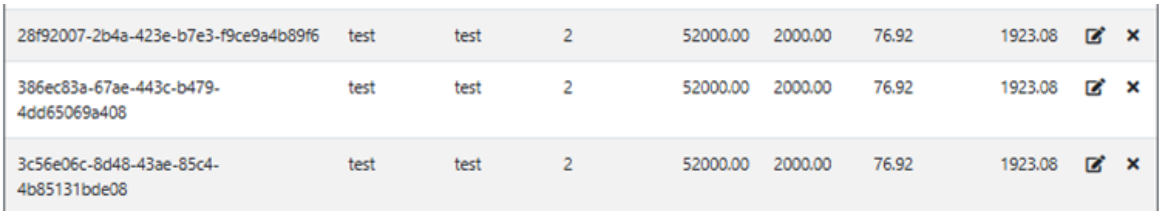
Duplicate Employee Entries Created by Repeated Clicking of Add Button

Steps to reproduce:

1. Go to Benefits Dashboard page
2. Click on Add Employee button
3. Fill all the fields
4. Click on Add button many times quickly

Actual Results:

Each click on the Add button results in the employee being added to the system multiple times, creating duplicate entries in the employee list



28f92007-2b4a-423e-b7e3-f9ce9a4b89f6	test	test	2	52000.00	2000.00	76.92	1923.08	
386ec83a-67ae-443c-b479-4dd65069a408	test	test	2	52000.00	2000.00	76.92	1923.08	
3c56e06c-8d48-43ae-85c4-4b85131bde08	test	test	2	52000.00	2000.00	76.92	1923.08	

Expected Results:

The system should add the employee only once, regardless of how many times the Add button is clicked.

After the first successful addition, subsequent clicks should be ignored or the button should be disabled until the operation is complete

Session Timeout: Unauthorized Error After 10 Minutes of Inactivity

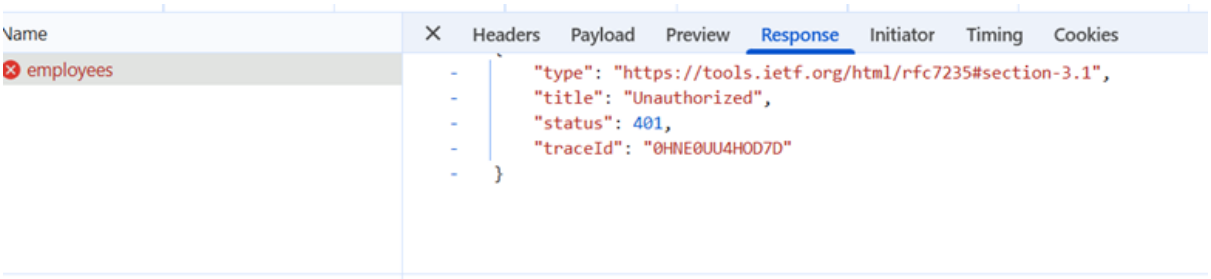
Steps to reproduce:

1. Go to Benefits Dashboard page
2. Wait 10 minutes without performing any actions
3. Click on Add Employee button

Actual Results:

Upon clicking the "Add Employee" button, the application does not respond or perform the expected action.

A review of the network activity reveals an unauthorized error message, indicating that the session has expired.



Expected Results:

The application should handle session timeouts gracefully by:

Displaying a clear message to the user indicating that the session has expired due to inactivity.

Prompting the user to log in again to continue using the application.

Optionally, providing a countdown or warning before the session expires to allow the user to extend the session.

Enhancements

Disable Update Button Until Changes Are Made

Steps to reproduce:

1. Go to Benefits Dashboard page
2. Click on the Action Edit
3. Observe the "Update" button without making any changes to the employee details

Actual Results:

Update button remains enabled even when no changes have been made to the employee details.

Expected Results:

The "Update" button should be disabled by default and only become enabled when a change is detected.

Additional information:

Disabling the Update button until a change is made helps users understand that no updates are necessary unless they modify a field. This reduces confusion and enhances the clarity of the interface. It provides immediate visual feedback to the user that an action is required before they can proceed, which aligns with intuitive design principles.

Order uin how elements are being added

Api Bugs

API Bug: Restriction on Dependents Not Communicated to Frontend

API Bug: Restriction on Dependents Not Communicated to Frontend

Steps to Reproduce:

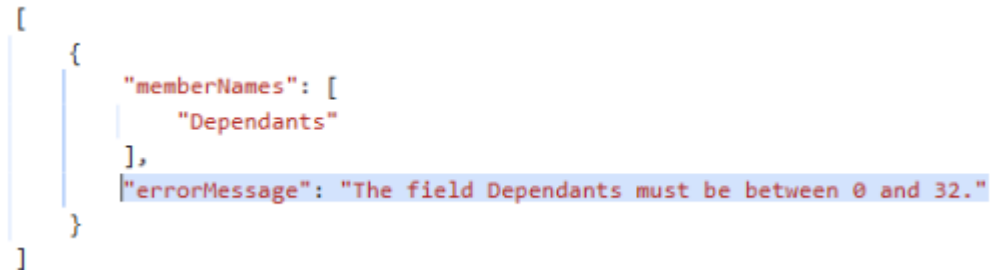
1. Open a tool like Postman or any API testing tool
2. Set the request method to POST
3. Enter the URL: https://wmxrwq14uc.execute-api.us-east-1.amazonaws.com/Prod/api/employees
4. Set the request body to JSON format with the following content:

```
{
   "firstName": "test3",
   "lastName": "test",
   "dependants": 1000
}
```

5. Send the request

Actual Results

The API returns an error message indicating that the field "dependants" must be between 0 and 32.



The employee is not added to the system.

Expected Results

The API should either:

Allow the addition of an employee with 1000 dependents if there are no communicated restrictions. Or, if there are restrictions, the frontend should display these constraints clearly to the user before submission.

The system should provide a user-friendly error message or UI indication if such restrictions exist.

Additional information:

Impact: This issue can lead to user confusion as the frontend does not indicate any restriction on the number of dependents.

Recommendation: Ensure that any backend restrictions are communicated to the frontend so that users are aware of input limits before submission. This can be done through validation messages or UI constraints.

Priority: High, as it affects the ability to add employees correctly and aligns with user expectations.

API Bug: Invalid Dependents Input Not Properly Handled

Steps to reproduce:

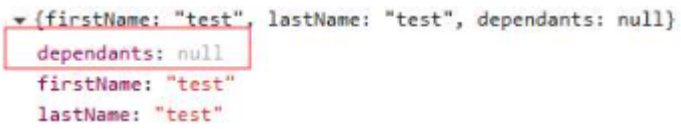
1. Open a tool like Postman or any API testing tool
2. Set the request method to POST.
3. Enter the URL for adding an employee: https://wmxrwq14uc.execute-api.us-east-1.amazonaws.com/Prod/api/employees.
4. Set the request body to JSON format with the following content, where dependants is a string or left empty:

```
{

   "firstName": "John",

   "lastName": "Doe",

   "dependants": "text"  // or leave this field empty

}
```

Actual Results:

The API accepts the request but sends the dependents field as null in the payload.

No error message or validation feedback is provided to indicate that the input is invalid.



Expected Results:

If the input is invalid (text or empty), the API should return a clear error message indicating that the dependants field must be a valid integer within the specified range.

Recommendations

Title 403 Forbidden Error for Favicon.ico Request

Steps to reproduce:

7. Go to Benefits Dashboard page
8. Open the browser developer tools usually accessible via F12 or right-click > Inspect
9. Go to the Network tab within the developer tools
10. Refresh the application page to capture all network requests
11. Observe the network requests and locate the request for favicon.ico
12. Note the status code returned for the favicon.ico request

Actual Result:

The request for favicon.ico returns a 403 Forbidden status code. The favicon is not displayed in the browser tab.

Expected Result:

Since the favicon is not part of the requirements, no specific behavior is expected. However, ideally:
The server should not return a 403 error for non-required resources.
If a favicon is not needed, the request should either not occur or return a 404 Not Found if the resource does not exist.

Additional information:

This issue does not impact the core functionality of the application but may affect user experience by not displaying a favicon in the browser tab.

Consider configuring the server to handle requests for favicon.ico gracefully, either by providing a default favicon or ensuring the request does not result in a 403 error.