Bug Report - Bug Finding Challenge

Project Name: Paylocity Benefits Dashboard

Report Date: October 7, 2024

Prepared by:
Pablo Eduardo López Ontiveros

Position: QA Engineer

Email: pablo_ontiveros1@outlook.com

Phone: +52 331 769 2227

Project Description:

This document outlines the bugs and defects identified in the Paylocity **Benefits Dashboard** application, focusing on both the User Interface (UI) and the API. The purpose of this report is to provide detailed information about the issues found to help developers improve the product's quality before the end-users interact with it.

Scope of the Report:

The scope of this document includes:

- **API:** Validation of the provided endpoints for managing employees and dependents, authentication review, and business logic testing.
- **UI:** Review of the graphical user interface, usability testing, and verification of benefits calculations.

2. API Bugs Report

API Bugs Report
Paylocity Benefits Dashboard
Tester: Pablo Ontiveros
Date: October 7, 2024

Bug #1: Wrong Salary and Benefits Cost When Adding New Employee

- **Priority:** Medium
- **Severity:** Major

Description:

When I add a new employee using the POST request, the salary and benefits cost values returned are incorrect.

Request Payload:

```
{
    "firstName": "New",
    "lastName": "Employee",
    "dependants": 2
}
```

API Response:

```
{
    "salary": 52000,
    "gross": 2000,
    "benefitsCost": 76.92308,
    "net": 1923.0769
}
```

Expected Result:

- The benefits cost should be based on the number of dependents. It should be $1000/year for the employee and $500/year for each dependent, which means it should total **$2000/year**.

Actual Result:

- The API returns benefits cost of **$76.92**, which is wrong.

Bug #2: Long Employee ID Required for GET Employee Function

- **Priority:** Medium
- **Severity:** Major

Description:

The **GET Employee** function requires a very long ID to retrieve employee details, which is not user-friendly and could lead to potential errors when copying or pasting the ID.

Steps to Reproduce:

1. Use the following API endpoint to retrieve employee details:

   ```
   GET https://wmxrwq14uc.execute-api.us-east-1.amazonaws.com/Prod/api/employees/2bc9cebc-eade-4787-a26d-11d7f6edcd29
   ```

2. Observe that a lengthy ID must be provided to successfully retrieve the employee details.

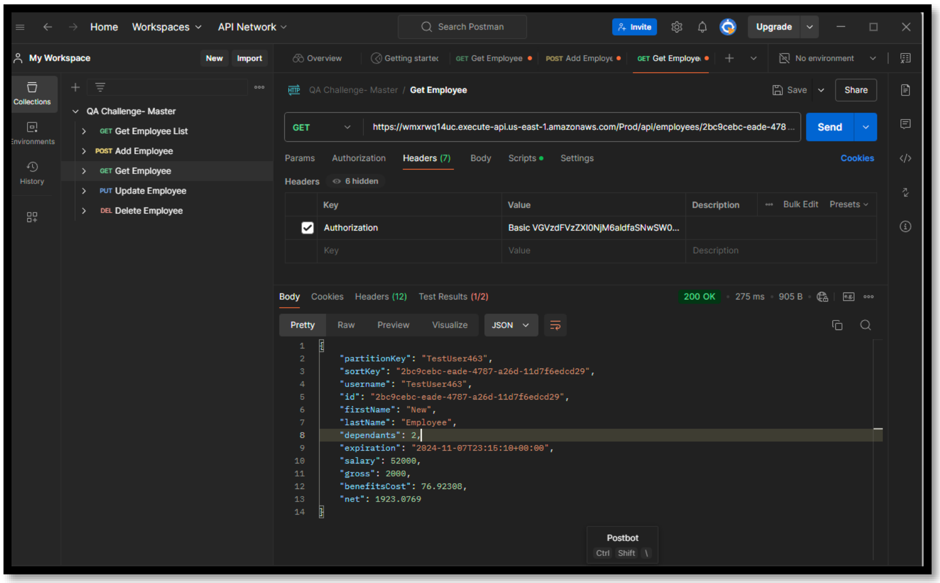
Expected Result:

- The system should allow for more user-friendly identifiers, potentially using shorter or more manageable IDs for ease of use.

Actual Result:

- The user must use a lengthy ID (e.g., `2bc9cebc-eade-4787-a26d-11d7f6edcd29`) to access the employee data, which can be cumbersome.

Screenshot:



GET https://wmxrwq14uc.execute-api.us-east-1.amazonaws.com/Prod/api/employees/2bc9cebc-eade-478 ...

	Key	Value	Description
☑	Authorization	Basic VGVzdFVzZXI0NjM6aldfaSNwSW0...	
	Key	Value	Description

```
{
    "partitionKey": "TestUser463",
    "sortKey": "2bc9cebc-eade-4787-a26d-11d7f6edcd29",
    "username": "TestUser463",
    "id": "2bc9cebc-eade-4787-a26d-11d7f6edcd29",
    "firstName": "New",
    "lastName": "Employee",
    "dependants": 2,
    "expiration": "2024-11-07T23:15:10+00:00",
    "salary": 52000,
    "gross": 2000,
    "benefitsCost": 76.92308,
    "net": 1923.0769
}
```

Bug #3: Inconsistent Employee ID Format

- **Priority:** Medium
- **Severity:** Minor

Description:

The employee ID format is inconsistent with other identifiers in the application, which can lead to confusion for users and developers.

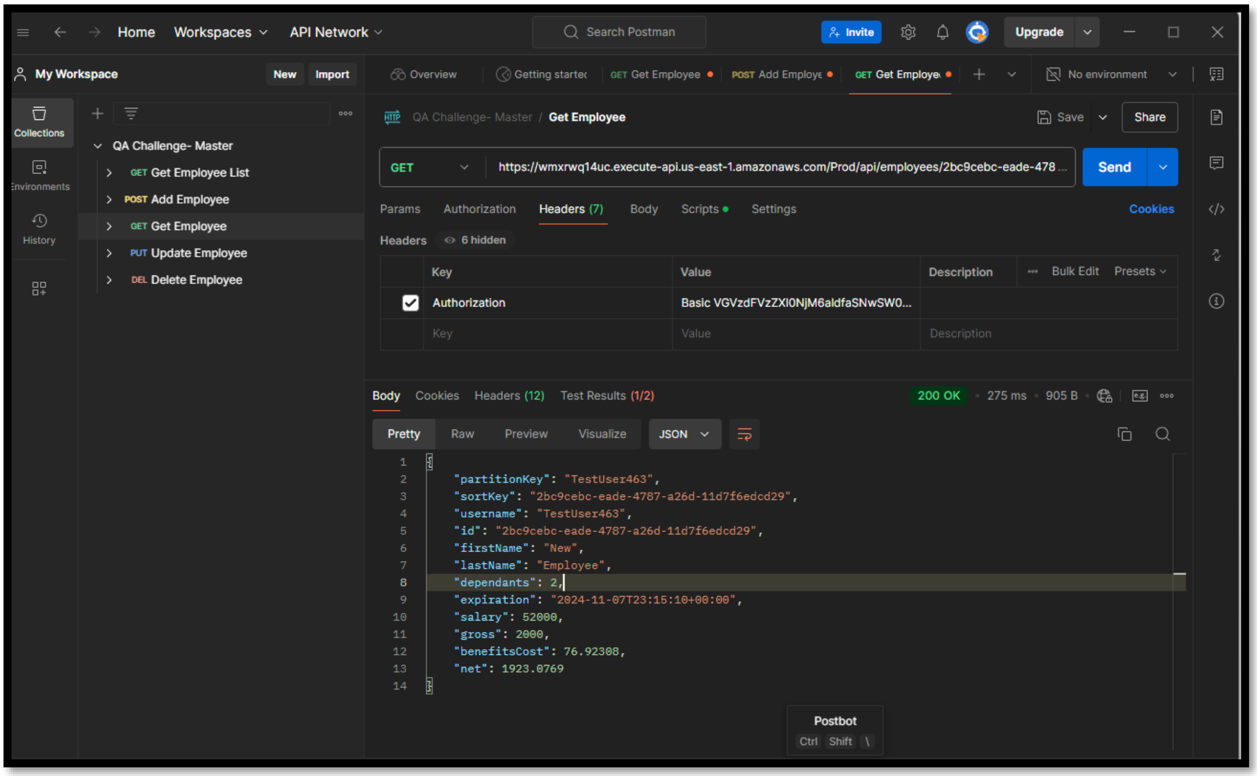
Expected Result:

- Employee IDs should follow a consistent format that is clear and easy to understand.

Actual Result:

- The employee ID is a long UUID format that may not align with other identifiers used within the application, making it harder to remember or manually input.

Screenshot:



QA Challenge- Master / Get Employee

GET https://wmxrwq14uc.execute-api.us-east-1.amazonaws.com/Prod/api/employees/2bc9cebc-eade-478 ...

Headers 6 hidden

	Key	Value	Description
☑	Authorization	Basic VGVzdFVzZXI0NjM6aldfaSNwSW0...	
	Key	Value	Description

200 OK · 275 ms · 905 B

```
{
    "partitionKey": "TestUser463",
    "sortKey": "2bc9cebc-eade-4787-a26d-11d7f6edcd29",
    "username": "TestUser463",
    "id": "2bc9cebc-eade-4787-a26d-11d7f6edcd29",
    "firstName": "New",
    "lastName": "Employee",
    "dependants": 2,
    "expiration": "2024-11-07T23:15:10+00:00",
    "salary": 52000,
    "gross": 2000,
    "benefitsCost": 76.92308,
    "net": 1923.0769
}
```

Bug #4: Inadequate Error Handling for Invalid Employee IDs

- **Priority:** High
- **Severity:** Major

Description:

When an invalid employee ID (e.g., `1`) is requested using the **GET Employee** API, the response does not provide a clear error message. Instead, it returns an HTML error page, which is not user-friendly and lacks useful information for troubleshooting.

Steps to Reproduce:

1. Use the following API endpoint with an invalid employee ID:

   `GET https://wmxrwq14uc.execute-api.us-east-1.amazonaws.com/Prod/api/employees/1`

2. Observe the HTML response indicating an error occurred.

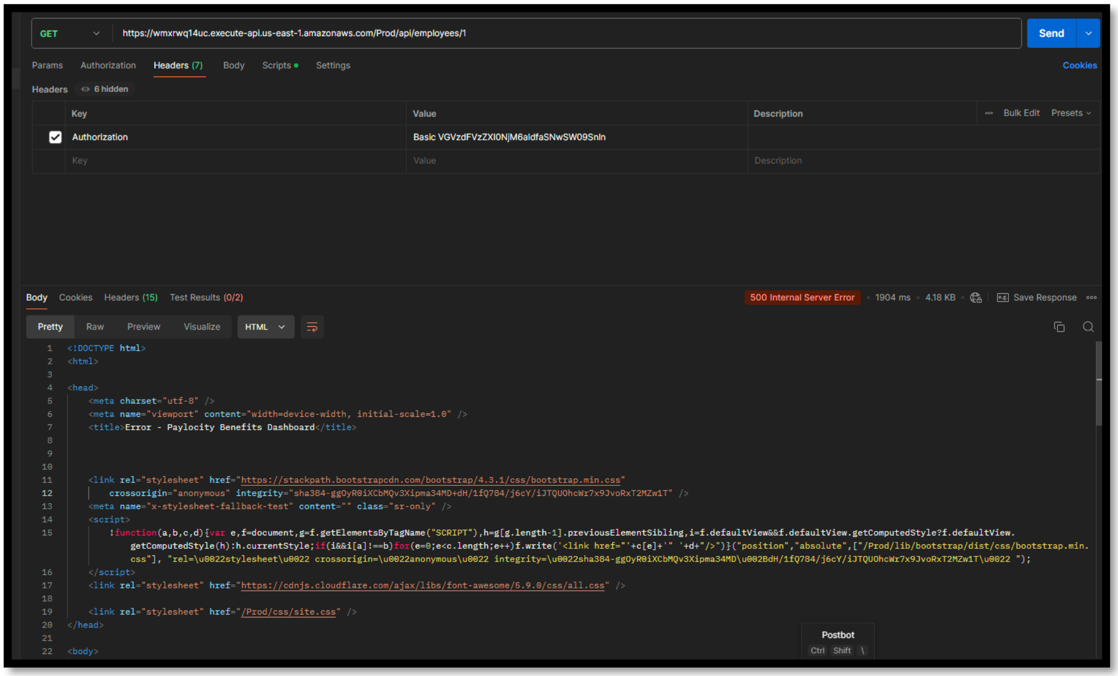
Expected Result:

- The API should return a `404 Not Found` status with a clear error message indicating that the employee ID is invalid.

Actual Result:

- The response contains an HTML error page that does not provide useful information for the user, such as the reason for the error or how to correct it.

Screenshot:



https://wmxrwq14uc.execute-api.us-east-1.amazonaws.com/Prod/api/employees/1

	Key	Value	Description
☑	Authorization	Basic VGVzdFVzZXI0NjM6aldfaSNwSW09Snln	
	Key	Value	Description

500 Internal Server Error · 1904 ms · 4.18 KB

```html
<!DOCTYPE html>
<html>

<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Error - Paylocity Benefits Dashboard</title>



    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
        crossorigin="anonymous" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" />
    <meta name="x-stylesheet-fallback-test" content="" class="sr-only" />
    <script>
        !function(a,b,c,d){var e,f=document,g=f.getElementsByTagName("SCRIPT"),h=g[g.length-1].previousElementSibling,i=f.defaultView&&f.defaultView.getComputedStyle?f.defaultView.
            getComputedStyle(h):h.currentStyle;if(i&&i[a]!==b)for(e=0;e<c.length;e++)f.write('<link href="'+c[e]+'" '+d+"/>")}("position","absolute",["/Prod/lib/bootstrap/dist/css/bootstrap.min.
            css"], "rel=\u0022stylesheet\u0022 crossorigin=\u0022anonymous\u0022 integrity=\u0022sha384-ggOyR0iXCbMQv3Xipma34MD\u002BdH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T\u0022 ");
    </script>
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.9.0/css/all.css" />

    <link rel="stylesheet" href="/Prod/css/site.css" />
</head>

<body>
```

• Bug #5: Validations in PUT Employees

No Field Validation on Update:

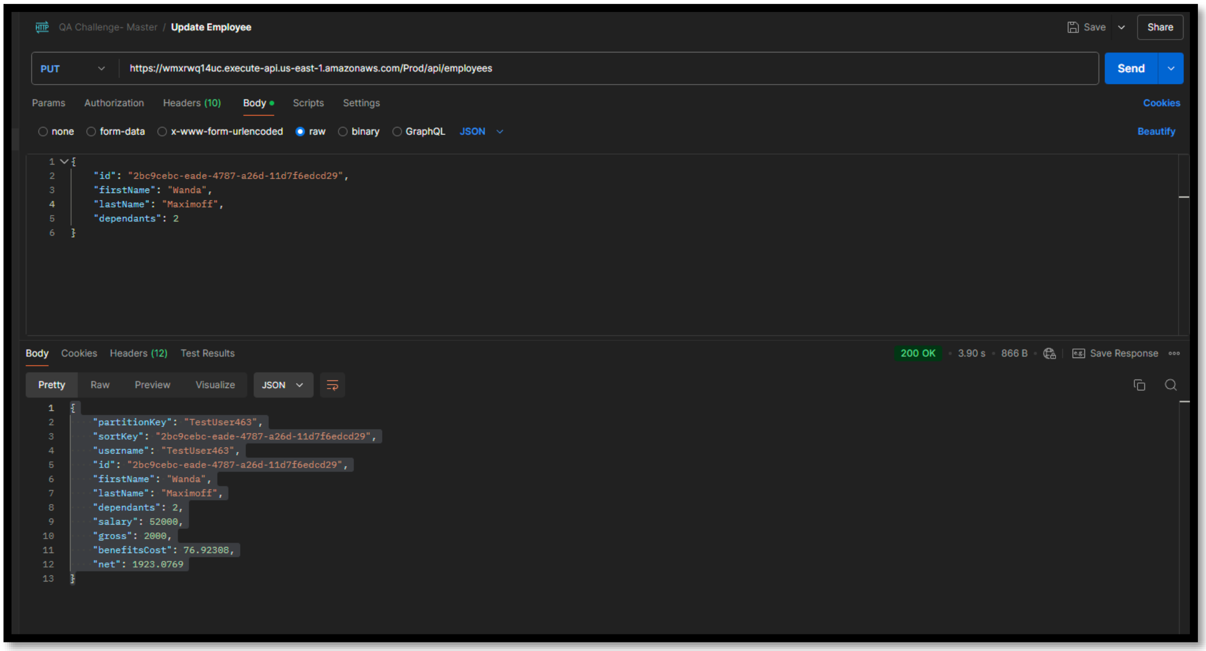
- **Description:** The API allows updating an employee without validating the fields properly. For instance, if the `firstName` or `lastName` includes special characters (like `#`), it should ideally return an error or handle them appropriately instead of accepting them.
- **Steps to Reproduce:** Attempt to update an employee with a first name containing special characters.

• No **Clear Response on Successful Update:**

- **Description:** When successfully updating an employee, the API returns the updated employee object, which is fine, but it might be helpful to also return a success message or status code in the response body to indicate that the operation was successful.
- **Steps to Reproduce:** Perform a valid PUT request and analyze the response for clarity.

• Potential **Overwriting of Existing Employee Data:**

- **Description:** When sending a PUT request with only some fields (e.g., only `firstName` and `dependants`), the API does not clarify if any other fields are overwritten with null or default values. This could lead to unintentional data loss.
- **Steps to Reproduce:** Send a PUT request with only partial data for an employee.



PUT https://wmxrwq14uc.execute-api.us-east-1.amazonaws.com/Prod/api/employees

○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON

```
{
    "id": "2bc9cebc-eade-4787-a26d-11d7f6edcd29",
    "firstName": "Wanda",
    "lastName": "Maximoff",
    "dependants": 2
}
```

Body · 200 OK · 3.90 s · 866 B · JSON

```
{
    "partitionKey": "TestUser463",
    "sortKey": "2bc9cebc-eade-4787-a26d-11d7f6edcd29",
    "username": "TestUser463",
    "id": "2bc9cebc-eade-4787-a26d-11d7f6edcd29",
    "firstName": "Wanda",
    "lastName": "Maximoff",
    "dependants": 2,
    "salary": 52000,
    "gross": 2000,
    "benefitsCost": 76.92308,
    "net": 1923.0769
}
```

Bug #6: Inadequate Response for DELETE Employees Request

- **Description:** When sending a DELETE request to remove an employee, the API responds with just `1`, which is unclear and lacks descriptive feedback. A more informative response, such as a success message, would be helpful for the user.
- **Steps to Reproduce:**
  1. Send a DELETE request to the API using the following URL format:

     ```
     DELETE https://wmxrwq14uc.execute-api.us-east-1.amazonaws.com/Prod/api/employees/{employeeId}
     ```

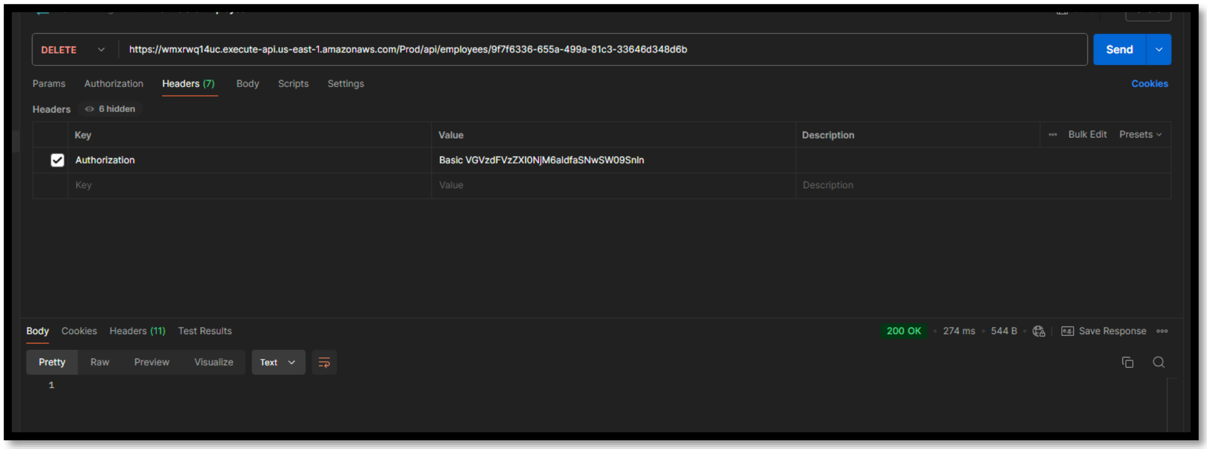
  2. Example request:

     ```
     DELETE https://wmxrwq14uc.execute-api.us-east-1.amazonaws.com/Prod/api/employees/9f7f6336-655a-499a-81c3-33646d348d6b
     ```

  3. The API returns only `1` as the response.

- **Expected Result:** The API should return a clear message, such as `Employee deleted successfully`, or at least a more meaningful status code like `200 OK` with a descriptive message.

Screenshot:



DELETE https://wmxrwq14uc.execute-api.us-east-1.amazonaws.com/Prod/api/employees/9f7f6336-655a-499a-81c3-33646d348d6b

	Key	Value	Description
☑	Authorization	Basic VGVzdFVzZXI0NjM6aldfaSNwSW09Snln	
	Key	Value	Description

Body — 200 OK · 274 ms · 544 B

```
1
```

Bug #7: Incorrect Response for DELETE Employees with Invalid ID

- **Description:** When sending a DELETE request with an invalid employee ID, the API still returns `1` but with a `405 Method Not Allowed` status code. This is misleading because the response does not clearly indicate that the employee does not exist or the request was incorrect.
- **Steps to Reproduce:**

1. Send a DELETE request to the API with an invalid or non-existent employee ID:

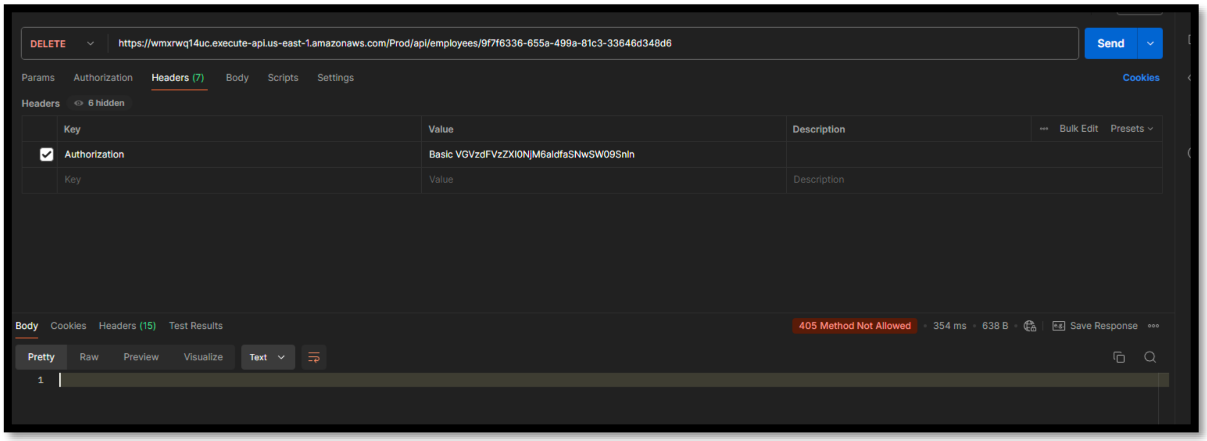
   ```
   DELETE https://wmxrwq14uc.execute-api.us-east-
   1.amazonaws.com/Prod/api/employees/{invalidEmployeeId}
   ```

2. Example request:

   ```
   DELETE https://wmxrwq14uc.execute-api.us-east-
   1.amazonaws.com/Prod/api/employees/invalid-id
   ```

3. The API returns `1` as the response, along with a `405 Method Not Allowed` status code.

- **Expected Result:** The API should return an appropriate error message, such as `Employee not found` or `Invalid employee ID`, along with a `404 Not Found` or `400 Bad Request` status code, instead of `405 Method Not Allowed`.



DELETE https://wmxrwq14uc.execute-api.us-east-1.amazonaws.com/Prod/api/employees/9f7f6336-655a-499a-81c3-33646d348d6

Headers 6 hidden

	Key	Value	Description
☑	Authorization	Basic VGVzdFVzZXI0NjM6aldfaSNwSW09SnIn	
	Key	Value	Description

405 Method Not Allowed • 354 ms • 638 B

```
1
```

Conclusion for API Bugs Report

During the testing of the **Paylocity Benefits Dashboard API**, several critical issues were found, impacting both the reliability and the security of the API. These bugs primarily revolve around improper error handling, inconsistent responses, and issues with validation and data retrieval.

Key findings include:

- **Error Handling**: When an incorrect or invalid `employee ID` is provided in certain requests (e.g., `DELETE` and `GET`), the API either returns an unclear response (e.g., "1") or a misleading error status code (e.g., `405 Method Not Allowed` instead of a more appropriate `404 Not Found` or `400 Bad Request`).
- **Response Consistency**: The API sometimes returns non-informative or partial responses (like `1`) that don't clarify the outcome of the operation, making debugging difficult for developers and error identification unclear for end-users.
- **ID Format Requirements**: The API requires very large and specific UUIDs for operations, making it difficult to use for manual testing or integration with other systems without generating these long IDs.

These bugs reduce the clarity and reliability of the API, making integration challenging and prone to unexpected errors. Improving error messages, adjusting the response codes to industry standards, and refining the ID requirements will enhance the API’s usability and transparency, ultimately increasing developer trust and application reliability.