UI Bugs:

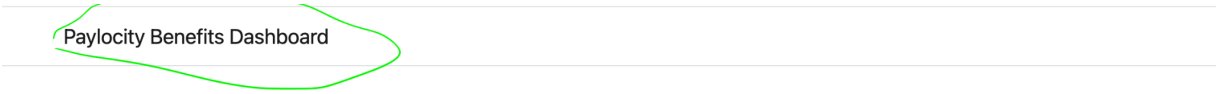
Summary:
Incorrect header “Paylocity Benefits Dashboard” on the login page
Description:
The login page displays a header titled “Paylocity Benefits Dashboard,” which should not be present. Furthermore, clicking on the header redirects the user to the benefits dashboard page, where a table is displayed along with an “Add Employee” button. This functionality should not be accessible from the login page.

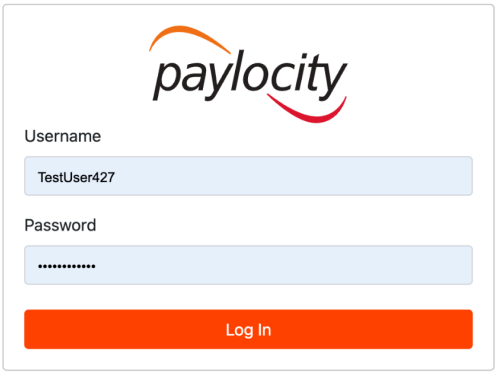
Steps to Reproduce:

1. Navigate to the login page.
2. Observe the presence of the “Paylocity Benefits Dashboard” header.
3. Click on the header.
4. Note that it redirects to the benefits dashboard page.

Environment:
Browser: Google Chrome Version 127.0.6533.120 (Official Build) (arm64)
Platform: macOS 14.3.1
Actual behavior:

- The login page displays an incorrect header.
- Clicking on the header redirects the user to a dashboard with a table and an “Add Employee” button.





Expected behavior:

- The login page should not display any headers.
- Clicking on any elements on the login page should not redirect the user to the benefits dashboard.

—----------

Summary:
Application crashes with a 405 error when the username length is not 11 characters
Description:
When logging into the application, if the username has a length different from 11 characters, the application crashes and returns a 405 error. This issue occurs regardless of the password or other input fields.
Steps to Reproduce:

1. Open the application login page.
2. Enter a username with a length different from 11 characters.
3. Enter a valid password.
4. Click on the login button.
5. Observe the application crashing with a 405 error.

Environment:
Browser: Google Chrome Version 127.0.6533.120 (Official Build) (arm64)
Platform: macOS 14.3.1
Actual behavior:

- The application crashes and returns a 405 error when a username with a length different from 11 characters is entered.

Expected behavior:

- The application should accept usernames of various lengths without crashing, as long as they meet the valid username criteria.

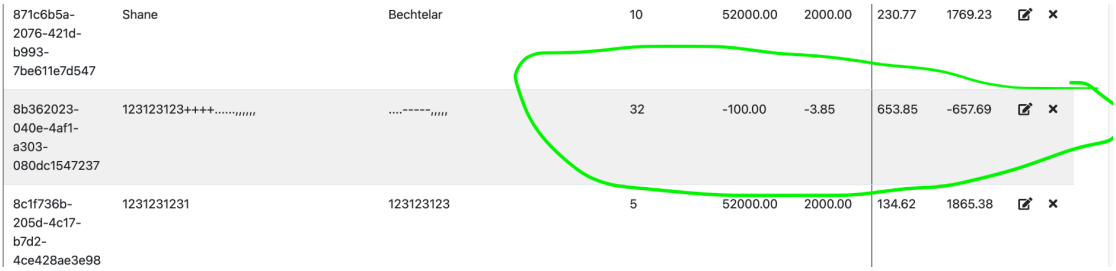
—----------

Summary:
Table layout shifts right when Employee name is too long
Description:
When an Employee name in the table is too long, the table exceeds its boundaries, causing the styling to shift to the right. This issue leads to misalignment of the table’s content and affects the overall appearance of the page.
Environment:
Browser: Google Chrome Version 127.0.6533.120 (Official Build) (arm64)
Platform: macOS 14.3.1
Steps to Reproduce:

1. Navigate to the dashboard page with the table.
2. Enter an Employee name that is excessively long.
3. Observe the table’s layout shifting to the right.

Actual behavior:

- The table content shifts to the right when an Employee name is too long, and the table exceeds its designated layout boundaries.



871c6b5a-2076-421d-b993-7be611e7d547	Shane	Bechtelar	10	52000.00	2000.00	230.77	1769.23	
8b362023-040e-4af1-a303-080dc1547237	123123123++++.....,,,,,,	....-----,,,,,	32	-100.00	-3.85	653.85	-657.69	
8c1f736b-205d-4c17-b7d2-4ce428ae3e98	1231231231	123123123	5	52000.00	2000.00	134.62	1865.38	

Expected behavior:

- The table should properly contain long Employee names within its boundaries without causing layout shifts, perhaps by truncating the name or adding word wrapping.

—----------

Summary:
No error message displayed when invalid data is submitted in AddEmployeeModal
Description:
When attempting to add an employee with invalid data (e.g., empty FirstName, LastName, or Dependents fields), no error message is shown after clicking the “Add” button. The modal remains open, and the employee is not created, but the user is not informed of the issue. Proper validation errors should be displayed to guide the user.
Environment:
Browser: Google Chrome Version 127.0.6533.120 (Official Build) (arm64)
Platform: macOS 14.3.1
Steps to Reproduce:

1. Open the AddEmployeeModal.
2. Leave one or more of the required fields (FirstName, LastName, Dependents) empty.
3. Click the “Add” button.
4. Observe that no error message is displayed and the employee is not created.

Actual behavior:

- When submitting invalid data (empty FirstName, LastName, or Dependents), the “Add” button does not trigger any error message. The employee is not created, but the user is not informed of what went wrong.

Expected behavior:

- An appropriate validation error message should be displayed if required fields (FirstName, LastName, or Dependents) are empty, informing the user to correct the input before submitting.

—----------

Summary:
No option to add EmployeeSalary in the AddEmployeeModal
Description:
In the AddEmployeeModal, there is no field available to input the EmployeeSalary. This omission prevents the user from adding the salary information when creating a new employee, which is an essential part of employee data.

Environment:
Browser: Google Chrome Version 127.0.6533.120 (Official Build) (arm64)
Platform: macOS 14.3.1
Steps to Reproduce:

1. Open the AddEmployeeModal.
2. Attempt to find a field for entering EmployeeSalary.
3. Observe that no such field is present.

Actual behavior:

- The AddEmployeeModal does not include a field for adding the EmployeeSalary, making it impossible to input salary data for new employees.

Expected behavior:

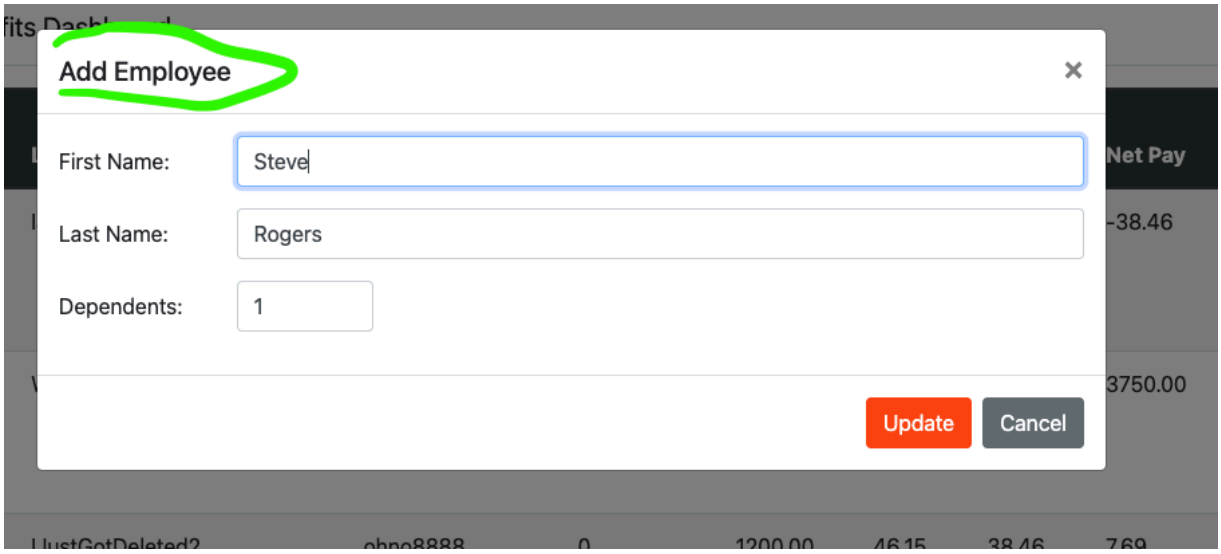
- The AddEmployeeModal should provide a field for entering EmployeeSalary when adding a new employee, allowing complete employee data to be submitted.

Summary:
Incorrect header and missing Salary field in UpdateEmployee modal
Description:
The UpdateEmployee modal displays an incorrect header, showing “AddEmployee” instead of “UpdateEmployee.” Additionally, there is no field available to add or update the Salary, making it impossible to modify the salary information for an existing employee. Both issues need to be addressed to ensure proper functionality and user clarity.
Environment:
Browser: Google Chrome Version 127.0.6533.120 (Official Build) (arm64)
Platform: macOS 14.3.1
Steps to Reproduce:

1. Open the UpdateEmployee modal.
2. Observe the header text displaying “AddEmployee” instead of “UpdateEmployee.”
3. Attempt to find a field for entering or updating Salary.
4. Observe that no such field is present.

Actual behavior:

- The header displays “AddEmployee” instead of “UpdateEmployee.”
- The UpdateEmployee modal does not include a field to add or update the Salary.



Expected behavior:

- The header should correctly display “UpdateEmployee.”
- The UpdateEmployee modal should provide a field to add or modify the EmployeeSalary.

Summary:

LastName and FirstName headers swapped in the dashboard table

Description:

In the dashboard table, the LastName and FirstName headers are incorrectly swapped. This causes confusion when viewing and interpreting the employee data, as the columns do not align with the correct headers.

Environment:

Browser: Google Chrome Version 127.0.6533.120 (Official Build) (arm64)

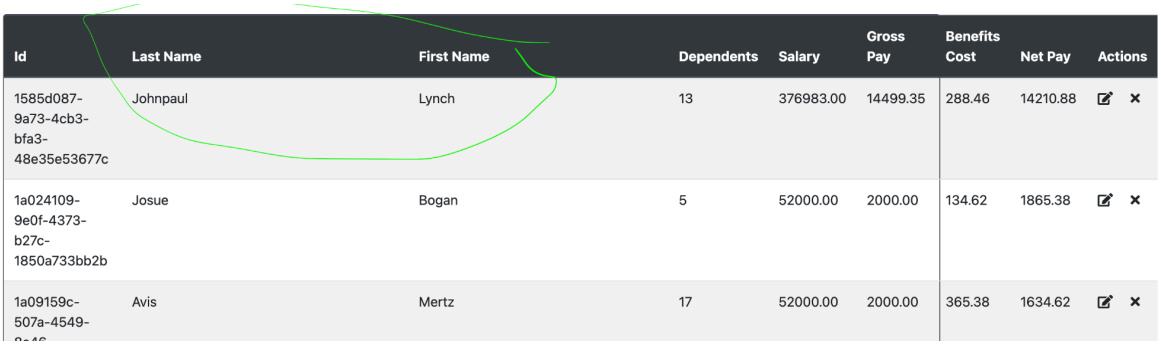
Platform: macOS 14.3.1

Steps to Reproduce:

1. Navigate to the dashboard page with the employee table.
2. Observe the placement of the LastName and FirstName headers.
3. Note that the headers are swapped, leading to confusion in the table data.

Actual behavior:

- The LastName and FirstName headers are placed incorrectly in the table, leading to a mismatch between the headers and the corresponding data.



Id	Last Name	First Name	Dependents	Salary	Gross Pay	Benefits Cost	Net Pay	Actions
1585d087-9a73-4cb3-bfa3-48e35e53677c	Johnpaul	Lynch	13	376983.00	14499.35	288.46	14210.88	
1a024109-9e0f-4373-b27c-1850a733bb2b	Josue	Bogan	5	52000.00	2000.00	134.62	1865.38	
1a09159c-507a-4549-	Avis	Mertz	17	52000.00	2000.00	365.38	1634.62	

Expected behavior:

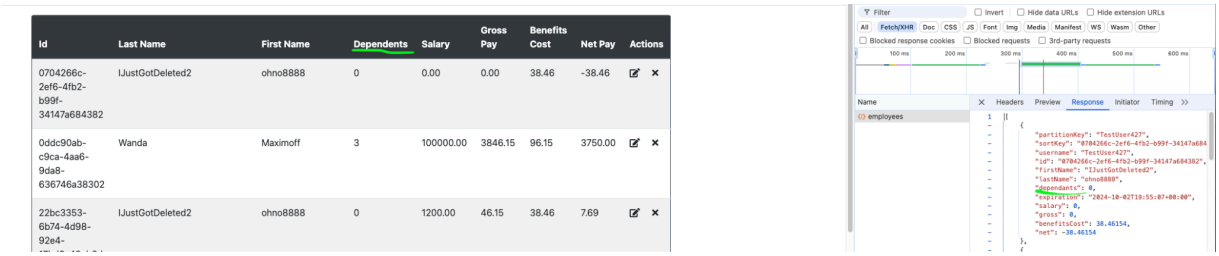
- The LastName and FirstName headers should be correctly aligned with their respective columns to ensure data is presented accurately.

Summary:
Inconsistent spelling of “Dependents” in dashboard table header and API request
Description:
The spelling of the word “Dependents” is inconsistent between the dashboard table header and the API request. The table header displays a different spelling compared to how it is spelled in the API request, leading to confusion and potential issues when trying to align the data.
Environment:
Browser: Google Chrome Version 127.0.6533.120 (Official Build) (arm64)
Platform: macOS 14.3.1
Steps to Reproduce:

1. Navigate to the dashboard page with the employee table.
2. Observe the spelling of the “Dependents” header.
3. Compare the spelling in the API request.
4. Note the inconsistencies in spelling.

Actual behavior:

- The word “Dependents” is spelled differently in the dashboard table header compared to the spelling in the API request.



Expected behavior:

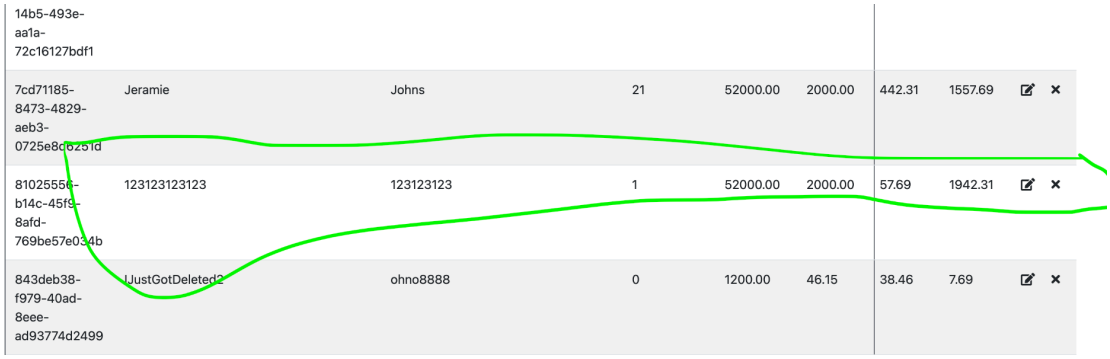
- The spelling of “Dependents” should be consistent between the dashboard table header and the API request to avoid any confusion or data mismatches.

Summary:
No validation for special characters or numeric input in FirstName and LastName fields
Description:
The FirstName and LastName fields do not have proper validation, allowing users to input any special characters or numeric values. This lack of validation can result in non-realistic names being entered, which could affect data integrity and lead to issues down the line.
Environment:
Browser: Google Chrome Version 127.0.6533.120 (Official Build) (arm64)

Platform: macOS 14.3.1

Actual behavior:

- Users can input special characters or numeric values in the FirstName and LastName fields without any restrictions or validation errors.



Expected behavior:

- The FirstName and LastName fields should only accept alphabetic characters, with validation in place to prevent the entry of special characters, numeric values, or any other non-alphabetic inputs.

API Bugs:

Summary:

POST call to /Prod/api/employees returns incorrect status code

Description:

When making a POST request to the /Prod/api/employees endpoint, the API returns a status code of 200 instead of the expected 201. The 201 status code should indicate that a new resource has been successfully created, but currently, the response incorrectly returns 200, which typically indicates a successful GET request.

Environment:

Endpoint: /Prod/api/employees

Steps to Reproduce:

1. Make a POST request to the /Prod/api/employees endpoint with valid employee data.
2. Observe the response status code.
3. Note that the API returns a status code of 200 instead of the expected 201.

Actual behavior:

- The POST request to /Prod/api/employees returns a status code of 200.

Expected behavior:

- The POST request to /Prod/api/employees should return a status code of 201, indicating that a new resource has been created.

—----------

Summary:

Duplicated fields in Employee object contain identical values

Description:
In the Employee object returned by the /Prod/api/employees endpoint, the fields sortKey and id have identical values. Similarly, the fields partitionKey and username also contain the same value. These duplicated fields should either have distinct values or be consolidated to avoid redundancy in the data structure.

Environment:
Endpoint: /Prod/api/employees

Steps to Reproduce:

1. Make a GET request to the /Prod/api/employees endpoint.
2. Inspect the Employee object in the response.
3. Observe that the sortKey and id fields have identical values, as do the partitionKey and partitionKey fields.

Actual behavior:

- The fields sortKey and id contain the same value.
- The fields partitionKey and username also contain the same value.

Expected behavior:

- The sortKey and id fields should have distinct values or be consolidated into a single field.
- The partitionKey and username fields should have distinct values or be consolidated into a single field.

—----------

Summary:
POST call to /Prod/api/employees ignores provided salary value and defaults to 52,000

Description:
When making a POST request to the /Prod/api/employees endpoint, the salary value provided in the request body is ignored. Regardless of the salary input, the Employee object in the response always returns with a salary of 52,000. This issue prevents the correct salary from being saved and reflected in the system.

Environment:
Endpoint: /Prod/api/employees

Steps to Reproduce:

1. Make a POST request to the /Prod/api/employees endpoint with an Employee object, specifying a salary value other than 52,000.
2. Inspect the response for the created Employee object.
3. Observe that the salary is always set to 52,000, regardless of the input.

Actual behavior:

•The Employee object is returned with a salary of 52,000, regardless of the salary value provided in the POST request.

Expected behavior:

•The Employee object should reflect the salary value provided in the POST request.

—----------

Summary:
Net pay value should not be be less than zero

Description:
In the Employee object, the net pay value should not be calculated less than zero. Currently, there is no validation preventing a negative net pay, which is incorrect, as net pay should always be a non-negative value.
Environment:
Endpoint: /Prod/api/employees
Steps to Reproduce:

1. Make a PUT request to the /Prod/api/employees endpoint with an Employee object, specifying a close to 0 salary value.
2. Inspect the response or the saved Employee object.

Actual behavior:

- The system calculation isn't correct net pay calculated with negative value

Expected behavior:

- The system calculation should not allow net pay calculated with negative value

—----------

Summary:
Missing validation for First Name, Last Name, Salary in POST and PUT call to /Prod/api/employees
Description:
The /Prod/api/employees endpoint lacks proper validation for the fields First Name, Last Name, Salary fields. It allows empty values to be submitted, which results in a generic 405 error without any specific message explaining what went wrong or why the error occurred. There should be proper validation to ensure these fields are not left empty, and clear error messages should be provided.
Environment:

Endpoint: /Prod/api/employees
Steps to Reproduce:

1. Make a POST request to the /Prod/api/employees endpoint, leaving the First Name, Last Name, Salary fields empty.
2. Observe that the system returns a 405 error without specifying what went wrong.
3. Note the lack of validation for these required fields.

Actual behavior:

- The system allows empty values for First Name, Last Name, Salary, resulting in a generic 405 error without any specific error message.

Expected behavior:

- The system should validate that First Name, Last Name, Salary not empty.
- If any of these fields are left empty, the system should return a clear validation error message, explaining which field is missing or invalid.

—----------

Summary:
PUT call creates a new Employee object when using a deleted Employee ID
Description:

When attempting to update an Employee object using a PUT call with the ID of a previously deleted employee, the system creates a new Employee object with a new ID instead of returning an error. The PUT call should not create new objects if no existing Employee object is found for the provided ID.

Environment:

Endpoint: /Prod/api/employees

Steps to Reproduce:

1. Delete an Employee object using the DELETE call and note the ID of the deleted Employee.
2. Make a PUT request to /Prod/api/employees using the deleted Employee's ID.
3. Observe that a new Employee object is created with a new ID instead of returning an error that no such Employee exists.

Actual behavior:

- The system creates a new Employee object with a new ID when using a deleted Employee's ID in a PUT call.

Expected behavior:

- The PUT call should not create a new Employee object if the specified Employee ID does not exist. Instead, it should return an error indicating that no Employee exists with the provided ID.

—----------

Summary:

PUT call allows negative values for salary

Description:

The /Prod/api/employees endpoint allows the submission of negative salary values through the PUT request. There is no validation preventing entering a salary less than zero, which can lead to incorrect data in the system, such as negative Net Pay values.

Environment:

Endpoint: /Prod/api/employees

Steps to Reproduce:

1. Make a PUT request to the /Prod/api/employees endpoint, specifying a negative salary value (e.g., -10,000).
2. Inspect the response or the updated Employee object.
3. Observe that the negative salary value is accepted without any validation errors.

Actual behavior:

- The system allows negative salary values to be submitted and saved through the PUT request.

Expected behavior:

- The system should validate the salary field to ensure that only positive values are accepted. Submissions with negative salary values should be rejected with an appropriate error message.

—----------

Summary:

DELETE call to /Prod/api/employees returns 200 instead of 204

Description:

When making a DELETE request to the /Prod/api/employees endpoint, the API returns a status code of 200 instead of the expected 204. The 204 status code should indicate that the request was successful and that no content is returned. Currently, the response incorrectly returns a 200 status code.

Environment:

Endpoint: /Prod/api/employees

Steps to Reproduce:

1. Make a DELETE request to the /Prod/api/employees endpoint with a valid Employee ID.
2. Observe the response status code.
3. Note that the API returns a status code of 200 instead of 204.

Actual behavior:

- The DELETE request to /Prod/api/employees returns a status code of 200.

Expected behavior:

- The DELETE request to /Prod/api/employees should return a status code of 204, indicating that the request was successful and no content is returned.

—----------

Summary:

Gross pay calculation is incorrect, resulting in an extra pay period

Description:

The gross pay is being calculated incorrectly. When dividing the salary by the gross pay, it results in 26 units, which implies 13 months of payment instead of the correct 12 months, even if the salary is meant to represent bi-weekly pay. This leads to an inaccurate salary distribution over the year.

Environment:

Endpoint: /Prod/api/employees

Steps to Reproduce:

1. Make a GET request to retrieve an Employee object from the /Prod/api/employees endpoint.
2. Divide the salary by the gross pay value.
3. Observe that the calculation results in 26 units, which implies 13 months of payment instead of 12.

Actual behavior:

- The gross pay is calculated incorrectly, resulting in 26 units when dividing the salary by the gross pay, which implies 13 months instead of 12.

Expected behavior:

- The gross pay should be calculated accurately, ensuring that it reflects 12 months of payment, with the correct bi-weekly or monthly distribution.

—----------

Summary:

expiration field in Employee object incorrectly set to one month after creation

Description:

The expiration field in the Employee object is incorrectly set to one month after the creation of the Employee. Employees should not expire after one month, and this expiration logic needs to be removed or corrected to reflect a more appropriate timeframe (or no expiration at all).

Environment:

Endpoint: /Prod/api/employees

Steps to Reproduce:

1. Make a POST request to create a new Employee object.
2. Retrieve the Employee object via a GET request.
3. Inspect the expiration field and note that it is set to one month from the creation date.

Actual behavior:

- The expiration field is automatically set to one month from the Employee creation date.

Expected behavior:

- The expiration field should not automatically expire Employees after one month, or there should be a more appropriate expiration logic.

Summary:

GET call using deleted Employee ID returns 200 instead of 404

Description:

When making a GET request to the /Prod/api/employees/ endpoint with the ID of a deleted Employee, the API returns a status code of 200 instead of 404. Since the Employee has been deleted, the API should return a 404 status code, indicating that no record exists for the provided ID.

Environment:

Endpoint: /Prod/api/employees/

Steps to Reproduce:

1. Delete an Employee object using the DELETE call and note the Employee ID.
2. Make a GET request to /Prod/api/employees/ using the deleted Employee ID.
3. Observe that the API returns a status code of 200, even though no record exists.

Actual behavior:

- The API returns a 200 status code when trying to retrieve a deleted Employee by ID.

Expected behavior:

- The API should return a 404 status code, indicating that the Employee with the given ID no longer exists.