1. ATM Testing

UI Test Scenarios for ATM

- Verify all label, text boxes, buttons, images, and links are displayed on the screen
- Verify if all informative text displayed on the screen are visible clear
- Verify if the size, color, and UI of the different objects are accordingly of the specifications
- Verify that the application's UI is responsive for example it should adjust to different screen resolutions of ATM

Functional Test Scenarios

- Verify the functionality of all the buttons on the keypad
- Verify if when inserting a valid card, different options, buttons will display on screen
- Verify if alert is display when inserting incorrect the card
- Verify if alert is display when inserting an invalid card
- Verify if no actions are allowed for expired card and proper alert is displayed for the user
- Check if pin is requested to user before displaying a card/bank information
- Verify if after an amount of failed attempt for user to add the card pin, the card is blocked in the ATM or any other operation is done in such a situation
- Verify if an alert is displayed after any failed attempt to enter a incorrect PIN
- Verify if PIN is displayed in masked form when is typed by the user
- Verify if user can check balance account
- Verify if other options like saving, currency, other accounts information are available
- Verify the ATM machine successfully takes out the money
- Verify if user receive an alert to type another amount of money to withdrawal when the initial typed amount is less than the account balance
- Verify if user can withdrawal a bigger amount of money than the account balance
- Verify if a alert is displayed when the user try to withdrawal a bigger amount of money than the account balance
- Verify if user an receive printed information of the transactions
- Verify if user can receive email information of the transactions
- Verify if ATM is out of money, a proper message is displayed to the user
- Verify if in case of electricity loss before the withdrawing cash, the transactions are marked as null and the amount on money is not withdrawn from the user account
- Verify if the ATM provides support for different languages
- Verify how much time the system takes to log out

2. How and for what to test inputs like:

Strings

Test with different types of strings, including empty strings, alphanumeric characters, special characters and long strings.

- Verify if application accepts valid strings and rejects invalid ones according to the defined criteria (length restrictions, character set limitations)

- Verify if input validation works as expected, such as handling edge cases (minimum and maximum lengths, boundary conditions)

- Check if applications handle special characters and encoding properly to prevent security vulnerabilities like SQL injections

Path / Files

Test with different types of file paths, including absolute paths, relative paths. paths with special characters and non-existing paths.

Test with various files types and sizes.

- Verify if application cand handle file uploads/downloads correctly and securely
- Check the behavior of the application when dealing with large files to ensure t doesn't degrade the performance or crash
- Verify and ensure that the application handles file errors like file not found, permission denied or disk space full
-

Time and Date

Testing input including Time and Date would depend on the context and requirements of the application.

Some common scenarios:

- Timezone handling: testing how the app / system handles timezones and check if correctly converts time and date for different timezones and no errors appear
- Format validation: check if application correctly validate the format of input time and date and reject invalid format.
- Boundary testing: check if system handles various boundary cases correctly. This includes testing the minimum and maximum values for time and date, leap years and edge cases like the end of months.

- Functional testing: check if all features related to date and time work as expected. This includes scheduling events, displaying timestamps, calculating durations, and any functionalities dependent on time and date information.
- Localization testing: verify if application support multiple languages and regions and ensure that time and date formats are displayed correctly according to the user's locale preferences.
- Regression Testing: verify if after making changes to the time and date related functionalities or the overall system, all that existing features work as intended.
- Error handling: verify how system handles error related to time and date inputs, including scenarios such as invalid input, missing input or unexpected errors during processing.

3. Test can be found on this repository
   https://github.com/Andreea-Gheban/GhostWebsiteTest

4. Test cases for http://dummy.restapiexample.com/ API

TC 1: Check Get for all employee data

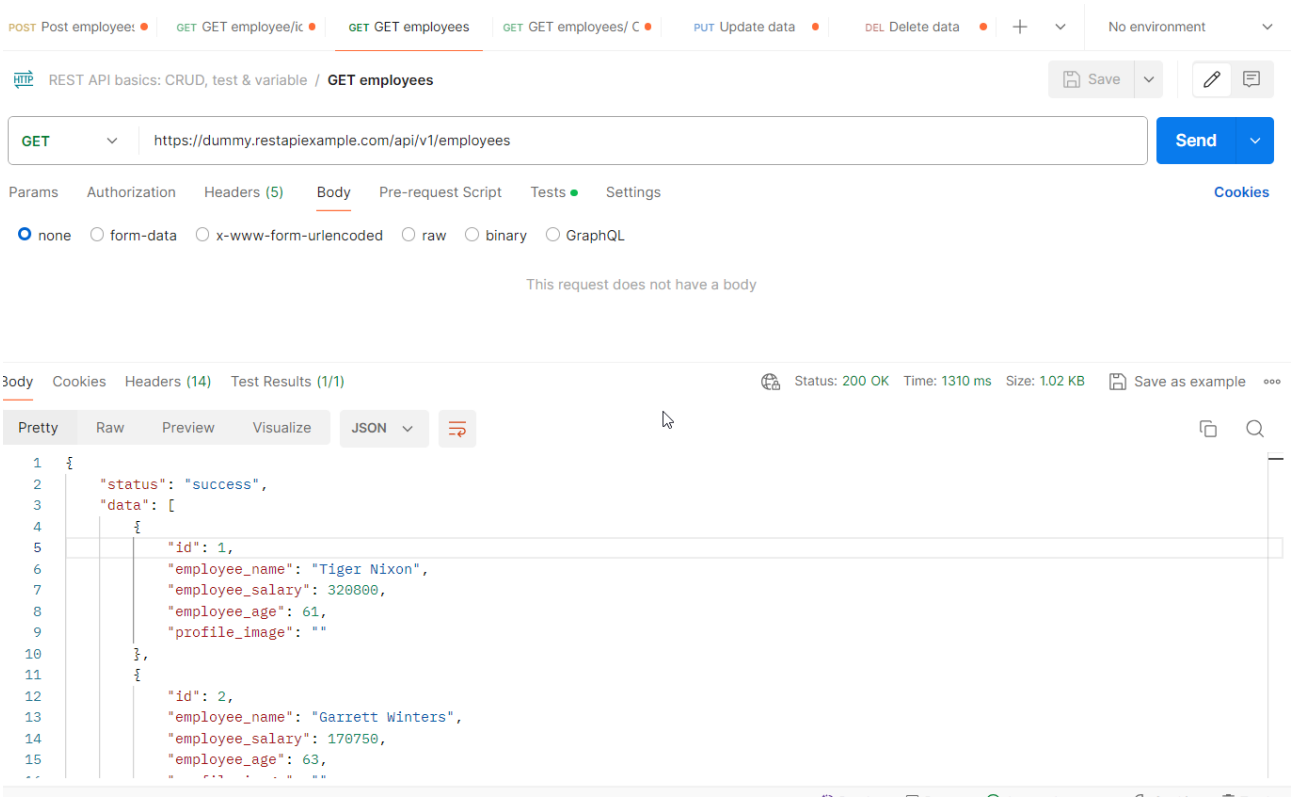
Given I perform GET request for https://dummy.restapiexample.com/api/v1/employees endpoint

When I check status code

Then I should receive 200 Ok

When I check body response

Then I should see a body response with value "success" for key "status"
And I should see a list with all employee details like “id”, "employee_name”, "employee_salary", “employee_age”, "profile_image



TC 2: Get employee details for specific id

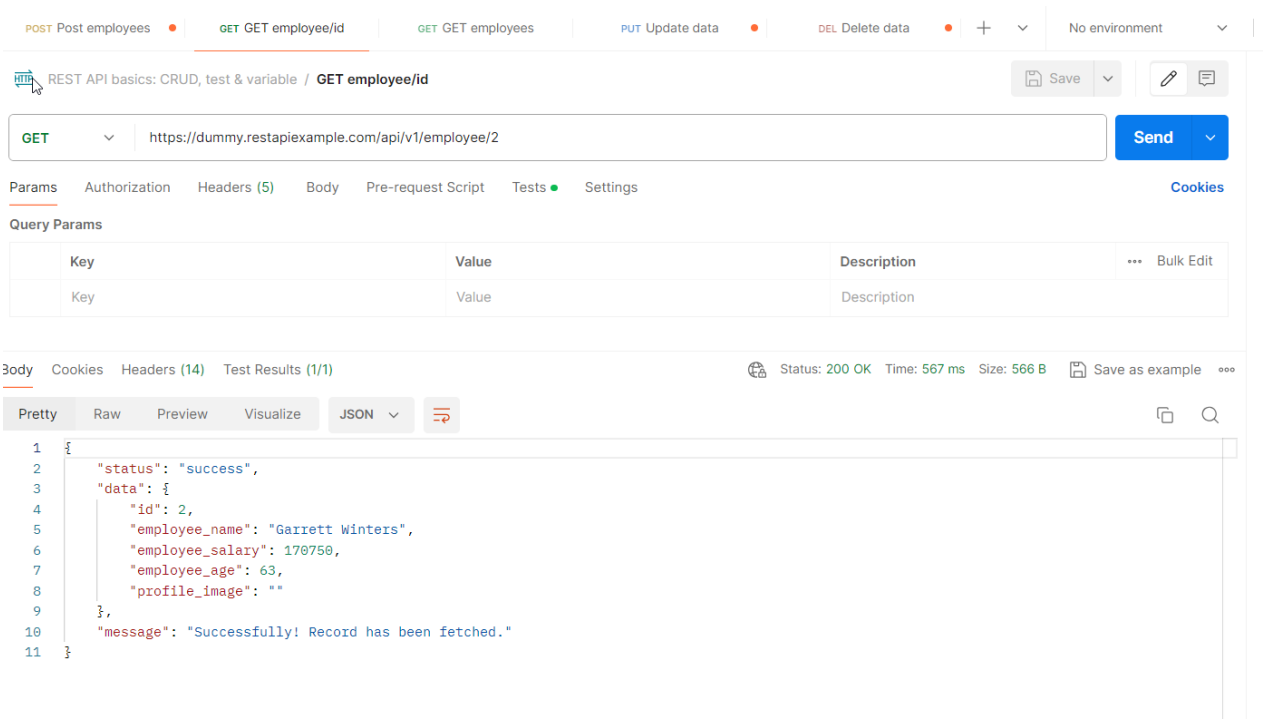
Given I perform GET request for https://dummy.restapiexample.com/api/v1/employee/2 endpoint

When I check status code

Then I should receive 200 Ok

When I check body response

Then I should see a body response with value "success" for key "status"
And I should see a list with all employee details like “id”, "employee_name”, "employee_salary", “employee_age”, "profile_image



TC 3: Create new record in database

Given I perform POST request for https://dummy.restapiexample.com/api/v1/create with body containing value for fields :

“name”, “salary”, “age”

And I press send button

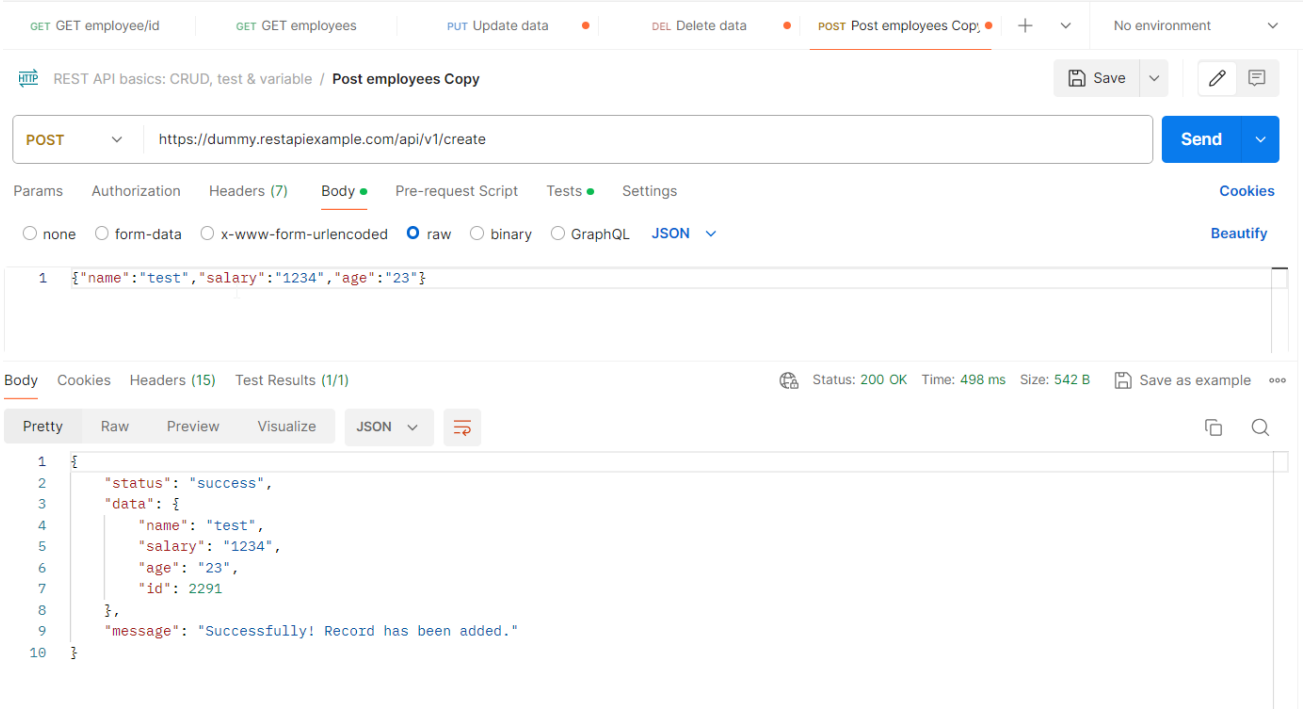
When I check status code

Then I should receive 200

When I check body response

Then I should see a body response with value "success" for key "status"
And I should see in body response the same employee details for “name”, "salary", "age", “id”

- Status code for POST request should be 201 Created



TC 3: Update an employee record

Given I perform POST request for https://dummy.restapiexample.com/api/v1/create with body containing value for fields : “name”, “salary”, “age”

And I press send button

When I check status code

Then I should receive 200

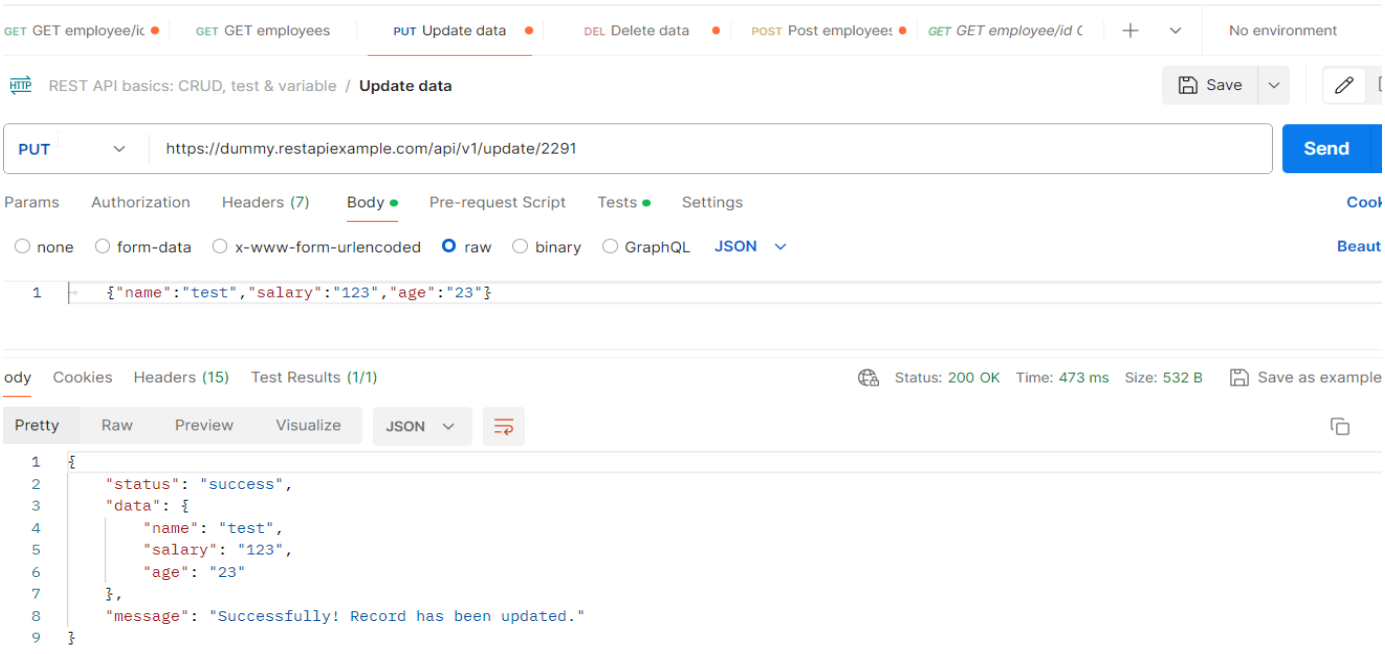
And I should see in body response the same employee details for “name”, "salary”, "age", “id”
When I send PUT request for https://dummy.restapiexample.com/api/v1/update/2291/ with different values for “name”, "salary”, "age", “id”

Then I should receive 200

Then I should see a body response with value "success" for key "status"
And I should see in body response the same employee details for “name”, "salary”, "age", “id”
When I perform GET request for https://dummy.restapiexample.com/api/v1/employee/2291 endpoint

And I should receive 200 Ok

Then I check on body response if values for “name”, "salary”, "age", “id” are updated



TC 3: Delete an employee record

Given I perform POST request for https://dummy.restapiexample.com/api/v1/create with body containing value for fields :

“name”, “salary”, “age”

And I press send button

When I check status code

Then I should receive 200

And I should see in body response the same employee details for “name”, "salary”, "age", “id”
When I send DELETE request for https://dummy.restapiexample.com/api/v1/delete/2

Then I should receive 200

Then I should see a body response with value "success" for key "status"
And I should see in body response the value “successfully! deleted Records” for “message”

When I perform GET request for https://dummy.restapiexample.com/api/v1/employee/21 endpoint

And I should receive 404