**Question 1 :Test Plan & Test Strategy**

**#Case of study**

We’re managing our Roche employees in a web application. The PO has requested us to verify the following requirements:

**2 functionalities :**

* The possibility to see the city of origin of the selected employees in a list.
* Login functionality

### **Test Strategy:**

#### **1. Objective:**

Ensure that the product meets quality standards and responds to the requirements stated :

* The possibility to see the city of origin of the selected employees in a list.
* Login functionality

#### **2. Scope:**

We will focus on testing the core functionality of the application, including login, data display, and basic user interactions.

#### **3. Testing Approach:**

* **Manual Testing:** For user interface (UI) checks, user interactions, and exploratory testing.
* **Automated Testing:** For repetitive tasks like login validation and regression testing.

#### **4. Test Tools:**

* Automation tool: **Cypress**.
* browser : **chrome**
* Api testing : **Postman**

#### **5. Test Environment:**

Testing will be performed on Chrome on a desktop environment.

**Test Plan:**

#### **1. Objective:**

Test the login functionality

ensure that the employee's city of origin is displayed correctly in the employee list.

#### **2. Scope:**

* **In-Scope:**
  + Login page functionality.
  + Display of employee city of origin in the list.
* **Out of Scope:** Other employee details like lastName or position and pagination

#### **3. Test Approach:**

* Manual testing for fonctionality validation
* Automated testing for repeated login cases.

#### **4. Test Cases:**

1. **Login Tests:**
   1. Test login with valid credentials.
   2. Test login with invalid credentials.
      1. Test login with invalid username
      2. Test login with invalid password
      3. Test with empty fields
   3. Test locked account after multiple failed attempts.
2. **Employee List Tests:**
   1. Verify city of origin is displayed correctly for selected employee.
   2. Verify city of origin is displayed correctly for multiple selected employees.
   3. Verify city of origin is displayed correctly for selected employee with no city mentioned.
3. **Endpoint Test :** 
   1. verify the endpoint of the login
      1. Test login with valid credentials.
      2. Test login with invalid credentials.
         1. Test login with invalid username
         2. Test login with invalid password
         3. Test with empty fields
   2. verify the endpoint to retrieve the list of the employees and their data
      1. retrieve a list of employees
      2. retrieve an specific employee

#### **5. Test Data:**

* Users with valid and invalid credentials.
* Employee data with and without city of origin.

#### **6. Test Environment:**

* Desktop browser - Chrome

**Test execution**

our test execution will be referring to 2 test plans :

* Login Test plan
* Employee list Test plan

1. **Login Test Cases**

**case 1** : Test login with valid credentials.

precondition : valid test data / the user is not already signed in

data : no data available (in our case -no details in the repo)

steps :

1-navigate to the login page

2-enter a valid username

3-enter a valid password

4-click sign in

**Expected Result** :

* User is logged in
* (Redirection in a case where the login page is displayed in an independent page )

**case 2** : Test login with invalid credentials.

precondition : invalid test data / the user is not already signed in

data : username : test

password : test

**scenario 1** : login with invalid Username

steps :

1-navigate to the login page

2-enter an invalid username

3-enter a valid password

4-click sign in

**Expected Result** :

* User isn’t logged in
* an error message

**scenario 2** : login with invalid password

steps :

1-navigate to the login page

2-enter a valid username

3-enter an invalid password

4-click sign in

**Expected Result** :

* User isn’t logged in
* an error message

**scenario 3** : login without filling the fields

steps :

1-navigate to the login page

2-leave the username field blank

3-leave the password field blank

4-click sign in

**Expected Result** :

* User isn’t logged in
* an error message

**case 3** : Test locked account after multiple failed attempts.

precondition :invalid test data

data : no data available (in our case -no details in the repo)

steps :

1-navigate to the login page

2-enter a invalid username

3-enter a invalid password

4-click sign in

5- repeat steps 2/3/4 - 5 or 6 times (depends on the app)

**Expected Result** :

* account locked
* an error message explaining that the account was locked

1. **Employee list Test Cases**

**case 1** : Verify city of origin is displayed correctly for selected employee.

precondition : user logged in / database contains multiple data of the employees

steps :

1- login to the application

2-navigate to the page where the employee list is displayed

3-check an employee with a city data

4-click on view selected data

5-verify the result

**Expected Result** :

* the employee and its city are displayed correctly

**case 2** : Verify city of origin is displayed correctly for multiple selected employees.

precondition : user logged in / database contains multiple data of the employees

steps :

1- login to the application

2-navigate to the page where the employee list is displayed

3-check multiple employees with a city data

4-click on view selected data

5-verify the result

**Expected Result** :

* the employees and their city are displayed correctly

**case 3** : Verify city of origin is displayed correctly for selected employee with no city mentioned.

precondition : user logged in / at least one employee without data

steps :

1- login to the application

2-navigate to the page where the employee list is displayed

3-check an employee without a city data

4-click on view selected data

5-verify the result

**Expected Result** :

* a message explaining that this employee doesn’t have a city mentioned

1. **Endpoint Test Cases**

**simulation because we don’t have endpoints provided in our case**

**Login Endpoint**

**case 1**

**testing tool : Postman**

**precondition :** Endpoint provided / valid test data

since it is a login method we are going to be using the **POST** method

identify the endpoint ( no information mentioned in our case so we are going to use an exemple / exemple : endpoint : /Login)

**steps :**

1. enter the endpoint in postman in a new request
2. send a post request with valid credentials in the body
3. verify that the response status is 200
4. in some cases the retrieval on a token and a refresh token (not mentioned)

**case 2**

**testing tool : Postman**

**precondition :** Endpoint provided / invalid test data

since it is a login method we are going to be using the **POST** method

identify the endpoint ( no information mentioned in our case so we are going to use an exemple / exemple : endpoint : /Login)

**steps :**

1. enter the endpoint in postman in a new request
2. send a post request with invalid credentials in the body
3. verify that the response status is 401 => Unauthorized

**list of employees and their data Endpoint**

**case 1** retrieve a list of employees

**testing tool : Postman**

**precondition :** Endpoint provided / database implemented with valid test data

since it is a method to retrieve data so we are going to be using the **GET** method

identify the endpoint ( no information mentioned in our case so we are going to use an exemple / exemple : endpoint : /GetEmployees)

**steps :**

1. enter the endpoint in postman in a new request
2. send a GET request to get all employees
3. verify that the response status is 200 with a list of employees in the response’s body

**case 2** retrieve an specific employee

**testing tools : Postman**

**precondition :** Endpoint provided / database implemented with valid test data

since it is a method to retrieve data so we are going to be using the **GET** method

identify the endpoint ( no information mentioned in our case so we are going to use an exemple / exemple : endpoint : /GetEmployees/:id)

**steps :**

1. enter the endpoint in postman in a new request
2. in the params specify the key “ID” and its value
3. send a GET request to that specific employee
4. verify that the response status is 200 with the chosen employee displayed and his informations are displayed in the response body

**Question 2**

Identify possible issues and report one if you identify something .

**Defects Report**

the test execution of the test cases mentioned resulted in extracting 4 Defects

**1** Defect of the Login functionality => **Defect 1**

**2** Defects of the employees list => **Defect 2** and **Defect 3**

**1** Defect after exploratory Tests => **Defect 4**

1. **Defect Title :** the login functionality fails

**Description**

whenever we enter valid data or invalid data in both the username field or the password field , or we leave them both blank , when we click on sign in the functionality doesn’t return anything

After inspection of the button “sign in” it turns out that it doesn’t invoke any function or method .

**Priority** : Highest

**Steps to reproduce**

1-open browser and and navigate to the application

2-enter data in the username field

3-enter data in the password field

4-click on sign in

**Expected Result** :

a successful login in the case of valid credentials and an error message in the case of invalid credentials .

**Actual Result :**

the page refreshes without any sign of success or failure .

1. **Defect Title :** Displaying "null" for City of Employee without City

**Description**

When the user picks an employee with a city that is blank or not mentioned the result is shown as the following: “employee is from null” .

**Priority** : High

**Test data used** :

First Name : Anne

Last Name : Dodsworth

City : Blank

**Steps to reproduce**

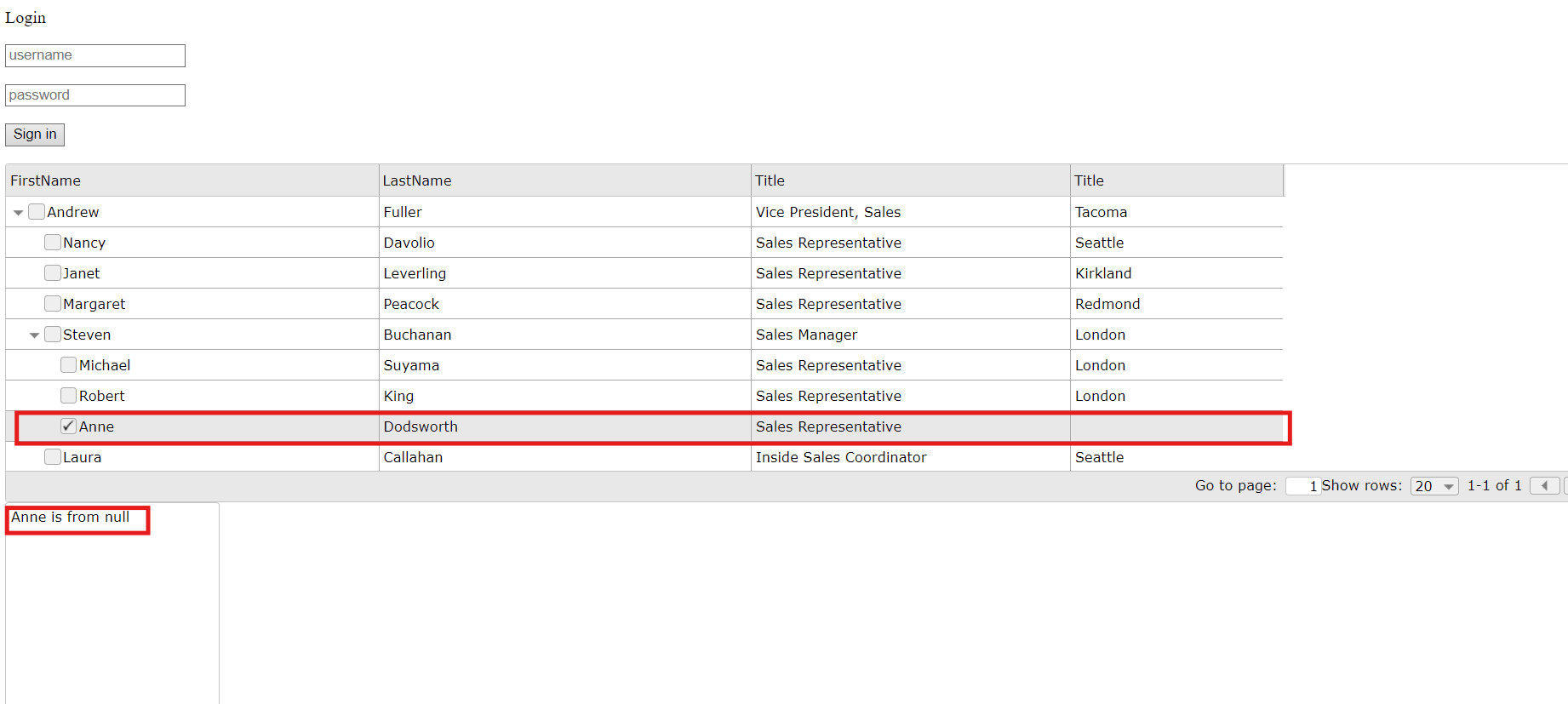
1. Open the browser and navigate to the app
2. Select an employee who does not have a city mentioned.
3. Click on the "View Selected Data" button.

**Expected Result** :

A more specific message should display , that mentions that the current employee does not have a city data

**Actual Result :**

The application displays : “employee is from null”



1. **Defect Title :** The title ofthe city column is incorrect

**Description**

when the user open the list of the employees , a table will be displayed ; where the column title of the city is mentioned as “Title” and not as “City”

**Priority** : High

**Steps to reproduce**

1-open browser and and navigate to the application

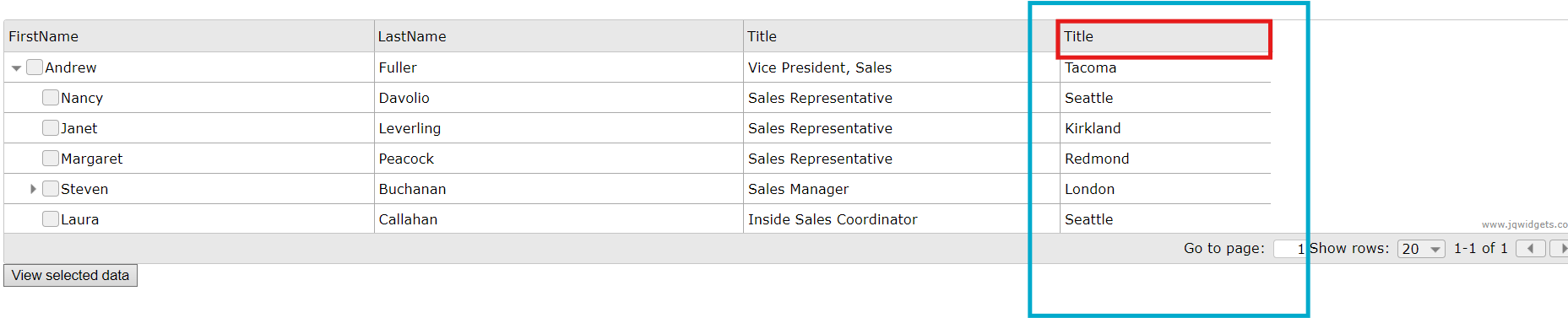
2-verify the table headers

**Expected Result** :

the column of the cities should have a header more significant , for example : city

**Actual Result :**

the column header of the city is mentioned as “Title” and not as “City”



1. **Defect Title :** Non-responsive content

**Description**

when the user opens the application and resize it , the content doesn't follow up the new dimensions

**Priority** : Medium

**Steps to reproduce**

1-open browser and and navigate to the application

2-resize the browser to new dimensions

**Expected Result** :

the application should handle responsive design

**Actual Result :**

the application doesn’t follow up the dimensions of the windows

**Question 3** : Do you agree with the requirement acceptance criteria? What are you missing? Put one example.

In our case we are missing the acceptance criteria in the file but in the case where the acceptance criteria were present and they cover the essential aspects of the feature I would suggest adding to them the behavior of the application while handling errors , it will be better to explicitly state the error message .

**question 4 :** Test automation implementation

**Respo :**