Software Testing’s Project

Project Logbook Report

Project Details
Team Members:
Senhua Liu, Van Alenn Pham, Huiting Feng, Camille Four, Jonathan EL BAZ
Project Start Date: 17/01/2024
Project End Date: 24/01/2024

Executive Summary

The project involved addressing several critical issues and bugs in our application, which required a systematic and collaborative approach to testing and resolution. This report documents our journey, the challenges we faced, the lessons we learned, and the solutions we implemented.

Challenges and Bugs

Bug: Hiring Contract Year Date
Description: We gave it a hiring contract which starts in 3001, by this time, I would have the time to rebirth at least 10 times if not more. In terms of data consistency and sense, it is wrong.
Questions Asked:
What is causing the bug in the Hiring Contract Year Date?
How can we test for it?

Bug: City and ZIP Code Difference
Description: When we enter the city (for example Paris), we can enter a ZIP code that doesn't correspond to Paris (like 78600). Which shouldn't be possible
Questions Asked:
Why is there a difference between the City and ZIP Code fields?
How can we ensure consistency?

Bug: Name's Field Correct Syntax
Description: We can add an employee that has a number as a name. Which is irrelevant since it is not a real name
Questions Asked:
What is the correct syntax for the Name field?
How can we validate it in our tests?

Bug: Address Field's Consistency
Description: We wanted to make sure that the addresses that were being typed in were correct. The thing is that whenever we want to add an address that is solely made of integers, it would work, such as 77777777. Which is irrelevant since it is not a real address.

Questions Asked:
How can we ensure consistency in the Address field?
What are the requirements?

Bug: ZIP Code Consistency
Description: We wanted to see if we could put an unlimited amount of numbers in the Zip Code field, naturally there isn't a Zip Code on Earth that is such as 11111111111111111111111111111111111111111. Adding this Zip Code provoked a Server Error (500).
Questions Asked:
How can we ensure that ZIP codes are consistent throughout the application?

Bug: Teams Names according to Caps
Description: We tried adding teams such as: "Arial" and "Arial". Despite being the same name, the program considered it was fine simply by the fact that the L was in cap. Thus, might lead to future errors when it comes to team identifications.
Questions Asked:
How should the team's name be formatted, and how can we enforce this?

Bug: Reset Database Without Password
Description: Fix the security issue, no one can reset the database like this without any password except the admin.
Questions Asked:
Is it possible to reset the database without entering a password?
How can we test this?

Bug: Remove Manager Role
Description: We can add the Manager role to an employee, but we cannot remove it
Questions Asked:
Can the Manager role be removed?
How can we write tests to check for this?

Approach and Testing

Our approach to addressing these challenges and bugs involved a series of steps:

Identification: During the initial phase of the project, we carefully reviewed the application and identified several challenges and bugs that needed attention. These issues were hindering the application's functionality and user experience. We identified the key challenges and bugs in the project according to the Excel we had done in the previous session and the issues in our GitHub and established a plan to dispatch tasks as a whole group.

Questioning: After identifying the challenges, our next step was to ask critical questions related to each issue. These questions were designed to delve deeper into the root causes and implications of the problems.

We wanted to ensure that we fully understood the issues and their impact on the application. This questioning phase allowed us to gain clarity on what needed to be tested and verified.

Test Planning: With a clear understanding of the challenges and bugs, we proceeded to plan our testing strategy. This involved determining the scope of testing, defining test objectives, and outlining the specific scenarios that needed to be covered thus according to the Excel we established and all the resources we have stacked up about the application. We also considered various edge cases and potential scenarios that users might encounter while using the application.

Collaboration: Collaboration played a vital role in our testing approach. We worked as a team to develop test cases that covered a wide range of scenarios. This collaborative effort ensured that we noticed all important aspects of testing.
Team members with different perspectives and expertise contributed to the creation of comprehensive test cases.

Execution: Using Playwright, we executed our test cases to validate the functionality of the application. We followed a systematic approach to ensure that the identified issues were resolved and that the application behaved as expected.
During the execution phase, we also looked for any unexpected behavior or errors that could have arisen.

Lessons Learned

Our project journey taught us valuable lessons:

One of the key takeaways from this project is the importance of effective testing. We realized that thorough and systematic testing is essential to catch and prevent issues at an early stage of the development process.
By investing time and effort in testing, we were able to identify and address potential problems before they reached the production environment. This not only saved time but also improved the overall quality of the application.
We learned that testing should be an integral part of the development lifecycle, and test cases should be designed to cover various scenarios and edge cases.

Conclusion

This report serves as a logbook of our project's progress and a testament to our team's dedication in addressing critical challenges and bugs. It highlights the importance of thorough testing, collaboration, and learning from our experiences in software development.