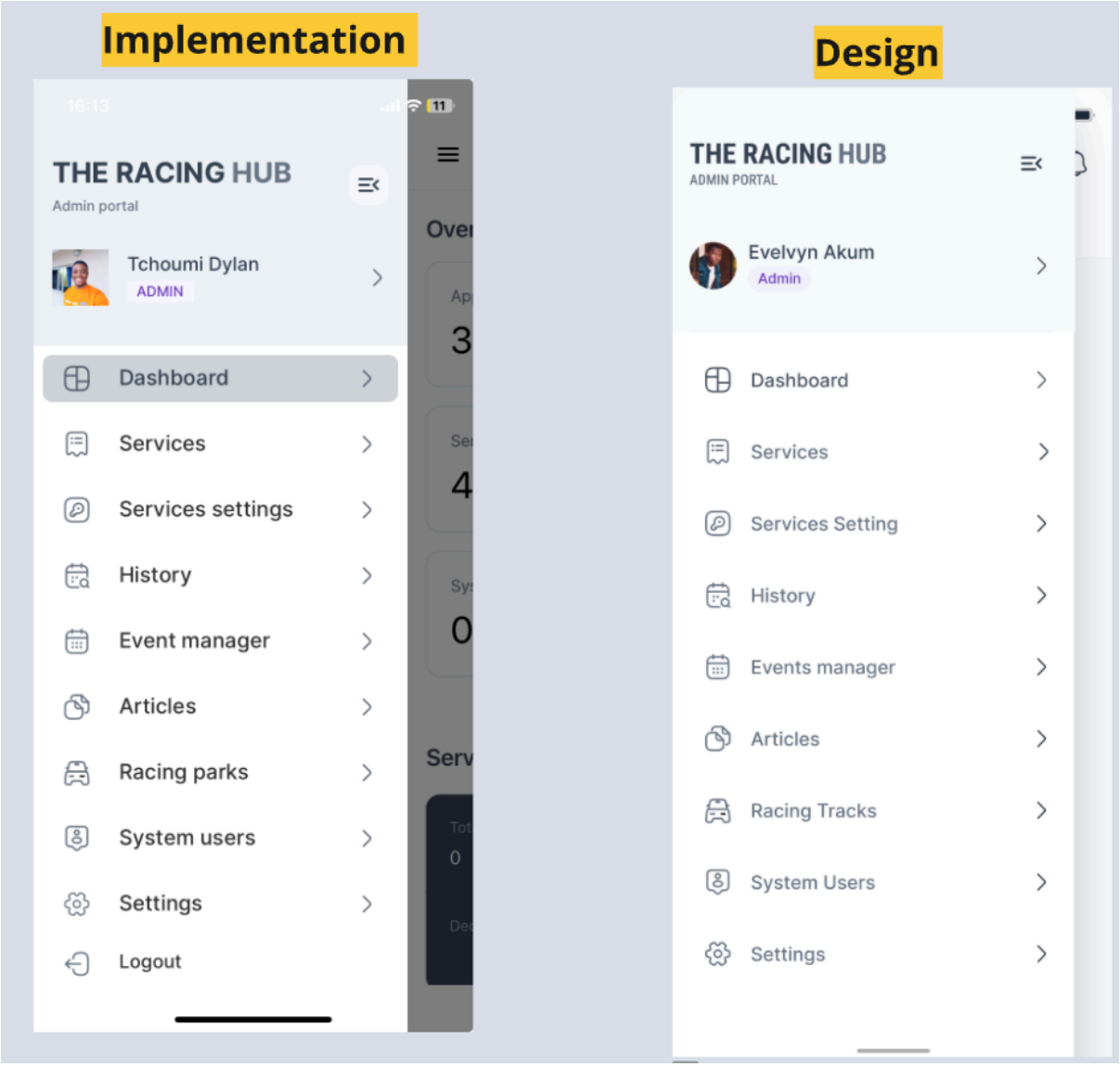
QA Engineer Challenge

Hey Do your best 😃

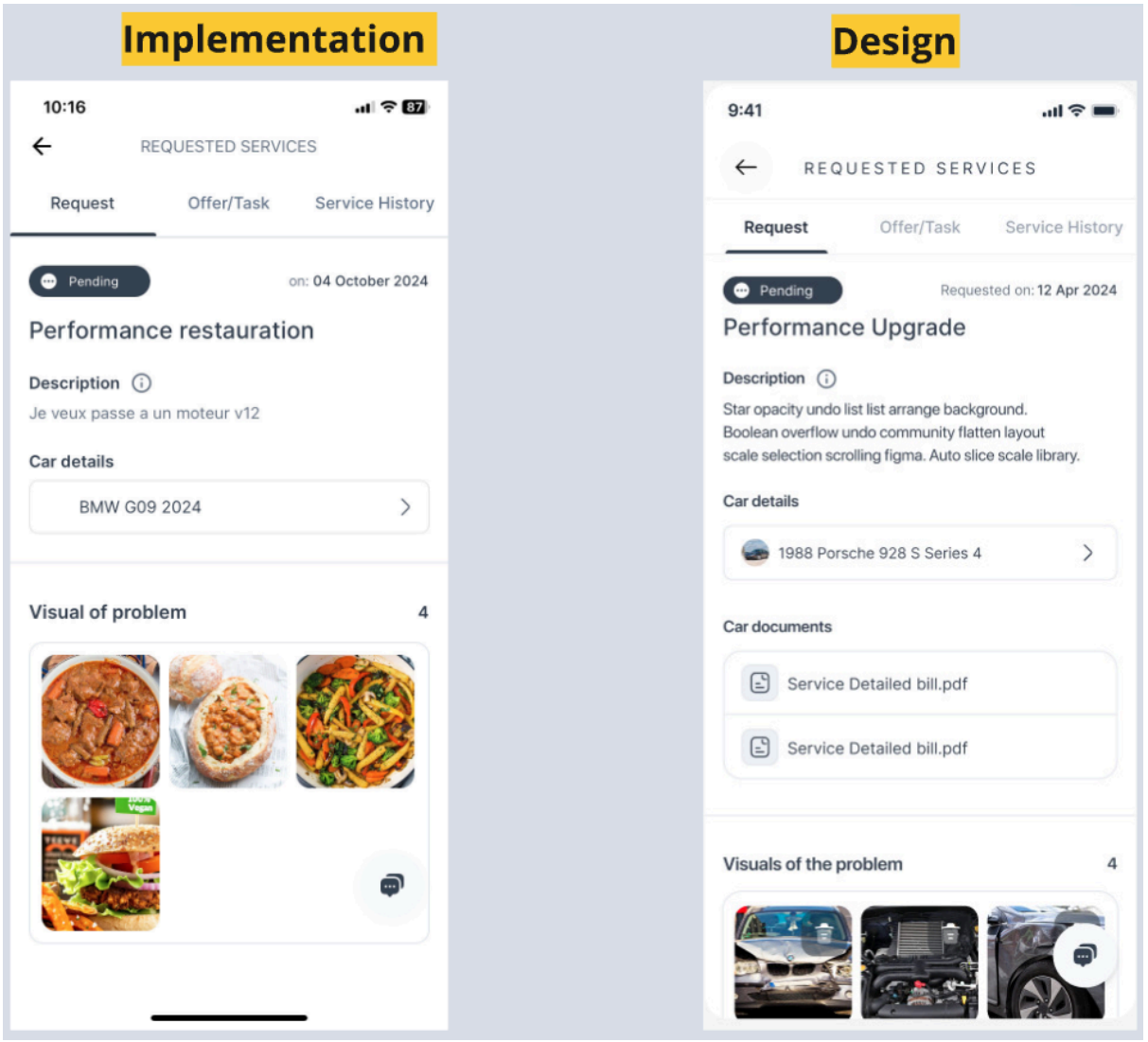
MISSION 1: UI Testing 🚀

The list of screenshots below shows the interface implemented by a developer on the left and the Figma design, which is what is expected, on the right.
Provide clear and meaningful test reports, including any visual problems.

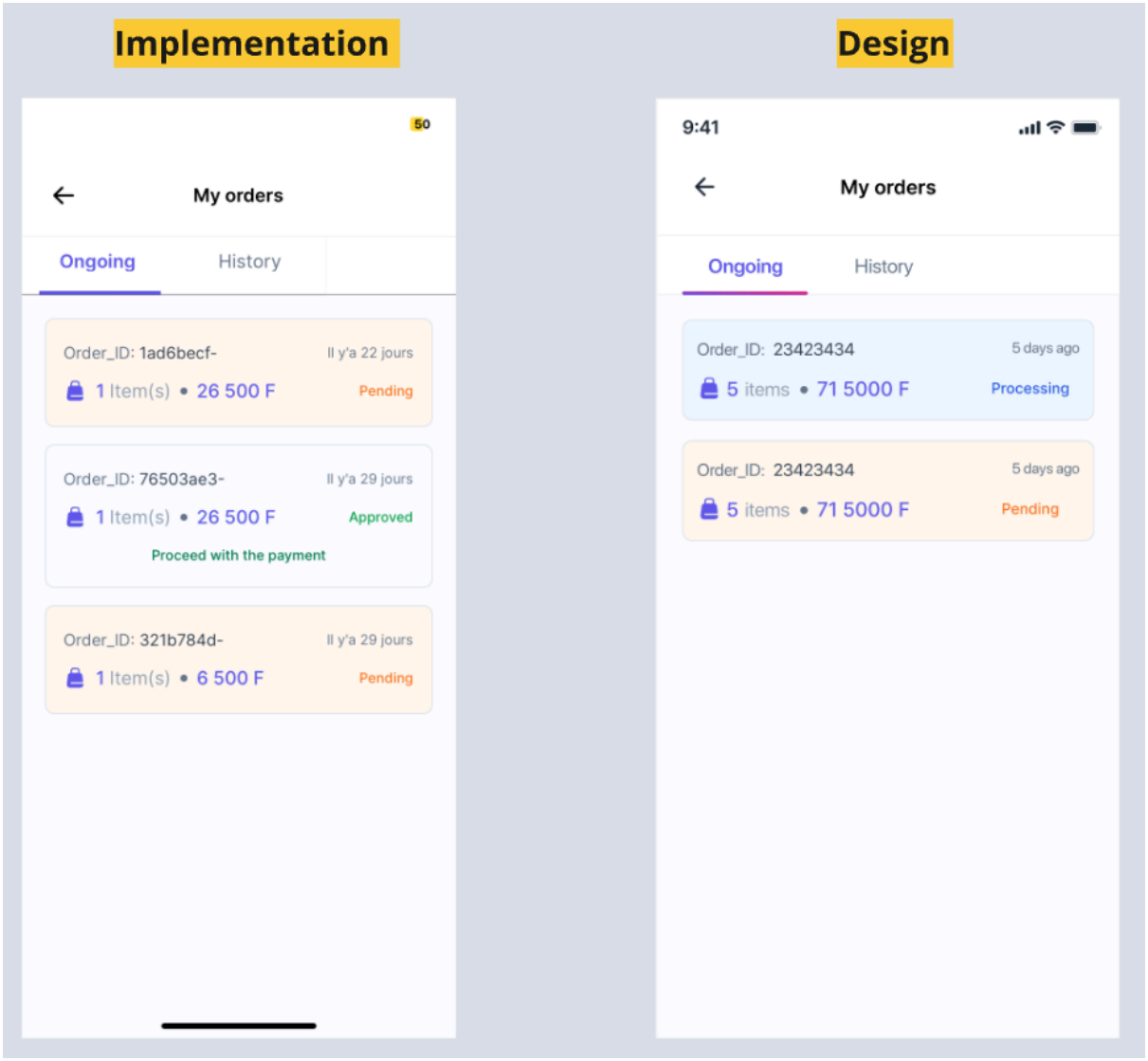
Interface 1.



Interface 2.


Implementation
10:16
87
REQUESTED SERVICES
Request
Offer/Task
Service History
Pending
on: 04 October 2024
Performance restauration
Description
Je veux passe a un moteur v12
Car details
BMW G09 2024
Visual of problem
4
Design
9:41
REQUESTED SERVICES
Request
Offer/Task
Service History
Pending
Requested on: 12 Apr 2024
Performance Upgrade
Description
Star opacity undo list list arrange background. Boolean overflow undo community flatten layout scale selection scrolling figma. Auto slice scale library.
Car details
1988 Porsche 928 S Series 4
Car documents
Service Detailed bill.pdf
Service Detailed bill.pdf
Visuals of the problem
4

Interface 3.


Implementation
50
My orders
Ongoing
History
Order_ID: 1ad6becf-
Il y'a 22 jours
1 Item(s) • 26 500 F
Pending
Order_ID: 76503ae3-
Il y'a 29 jours
1 Item(s) • 26 500 F
Approved
Proceed with the payment
Order_ID: 321b784d-
Il y'a 29 jours
1 Item(s) • 6 500 F
Pending
Design
9:41
My orders
Ongoing
History
Order_ID: 23423434
5 days ago
5 items • 71 5000 F
Processing
Order_ID: 23423434
5 days ago
5 items • 71 5000 F
Pending

MISSION 2: Exploratory Testing and automation 🚀

Introduction

You have been assigned the task of automating the QA testing process for a system designed to align with the specified user story. Your final objective is to create automated tests for the system accessible via the provided URL: https://playwright-lab.web.app/

User Story

As a QA engineer with good skills in automation and CI/CD, the objective is to design test cases that validate the functionality of the web page in creating a user profile with accurate error handling and which will be automated.

The web page should include the following input fields:

Required Form Fields

1. **First name (mandatory):** A text field accepting alphabetical characters only (e.g., John).
2. **Last name (mandatory):** A text field accepting alphabetical characters only (e.g., Smith).
3. **Email (mandatory):** An email field accepting valid email addresses (e.g., john.smith@example.com).
4. **Password (mandatory):** A password field accepting alphanumeric characters and
symbols (e.g., P@ssw0rd).
5. **Confirm password (mandatory):** A password field matching the above password field.

Optional Form Fields

6. **Gender (optional):** A radio button or select field allowing the user to choose "male,"
"female," "non-binary," or "prefer not to say."
7. **Date of birth (optional):** A date field accepting a valid date in the format of YYYY-MM-
DD (e.g., 1990-01-01).
8. **Phone number (optional):** A text field accepting only numeric characters with a
maximum length of 10 digits (e.g., 1234567890).
9. **Address (optional):** A text field accepting alphanumeric characters, spaces, and symbols
(e.g., 123 Main St, Apt 1).
10. **LinkedIn URL (optional):** A text field accepting a valid LinkedIn profile URL
(e.g., https://www.linkedin.com/in/johnsmith).
11. **GitHub URL (optional):** A text field accepting a valid GitHub profile URL
(e.g., https://github.com/johnsmith).

Part 1: Exploratory Test + Testing Cases + Bug Defect Report

Objective:
The objective of this part is to create a simple test plan (insert test cases) and bug reports.

Part 2: Test Automation using Playwright

Objective:
Automate the test cases designed in Part 1 (in your test plan) using Playwright (*don't be afraid in you don't know about playwright take a moment learn about it a do the challenge* 😃) with TypeScript (*if you feel comfortable with python, C# or Java go ahead* 😃).

Requirements:

1. Create Playwright scripts to automate the test cases outlined in Part 1.
2. Organize your code, include comments, and follow best practices for writing automated tests.

3. Ensure the Playwright scripts handle errors gracefully, especially during interactions with the webpage elements.

4. Use the Page Object Model (POM) or any other suitable design pattern for maintaining your automation code.

5. Provide clear and meaningful test reports, including details of passed and failed test cases.

6. Create a README file with instructions on how to set up and run your automated tests.

7. Commit your automation code on the same Git repository created for Part 1, following a logical sequence of commits.

8. Create GitHub actions which will be run if a push on the main branch.

Deliverable:

Submit the link to your Git repository that contains your solution for the Mission 1 and 2.

Additional Notes:

- You are encouraged to leverage Playwright's capabilities to handle different browsers
- (e.g., Chrome, Firefox, WebKit) and consider cross-browser compatibility.
- Ensure that your Playwright scripts are maintainable and can be easily extended as new features are added to the webpage.

- Include any necessary documentations, configurations or environment setup instructions in your README file.
- If you encounter challenges or limitations during automation, document them in your README file along with potential solutions or workarounds.