Introduction

Go maestro is a management business webpage where the business can create their own components where the user can manage different widgets and perform actions on these widgets to manage different kinds of actions and also users.

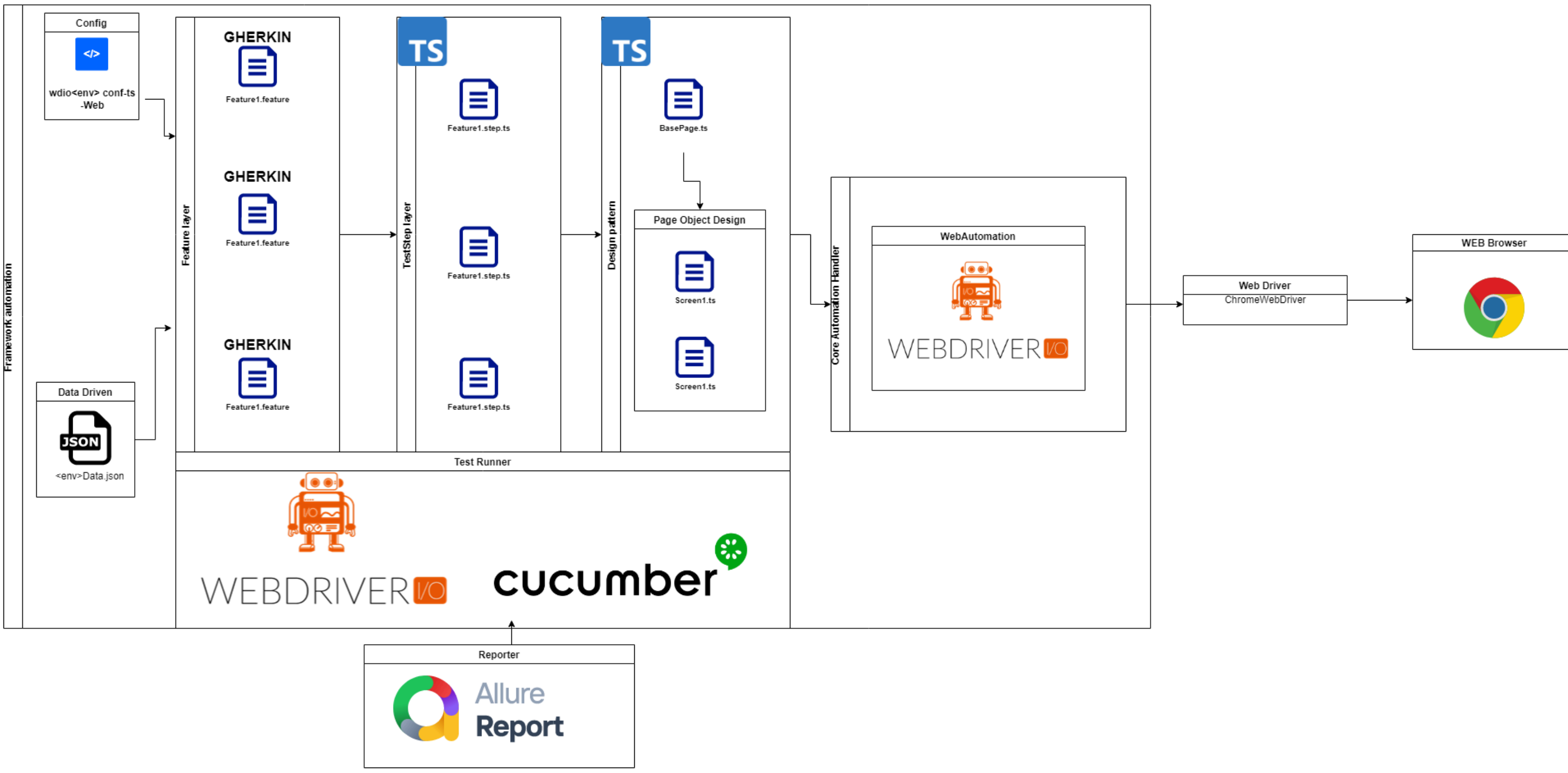
Test strategy

For more quality on the process of go maestro will be implemented automated test as the main scope trying to cover the maximum of automated tests but also will exist manual tests and some performance tests.

Test Implementation.

In order to execute the tests quickly and efficiently focused on generic tasks of the system, that is, main tasks of the system were tested to ensure high code coverage and also quality in the processes presented in the system, therefore, Both frontend and backend were tested.

Framework architecture.



It was decided to use the behavior-driven development (BDD) architecture through the use of Gherkin to test the code and functionalities for their behavior in the system with the different identified scenarios that are expressed with a natural language. for greater understanding. The use of this architecture provides the ability to reuse the code to obtain high coverage, maintainability, scalability and primarily reduction in test execution time.

Risks.

Some risks in the test could be specific behaviors that would not allow the automation of the test, such as interaction with the systems or challenges for automation, such as drag-and-drop behavior and some widgets that cannot be automated. To address this risk, there should be a focus on creating more automated tests. There will be a package of manual testing that will contain certain tests that are difficult to automate or are not suitable for automation.

Objective

The test objective is to provide more assurance regarding the main process, such as creating new enterprises and enabling those enterprises to create and interact with widgets easily. To ensure quality, we aim to implement an automation framework with the primary focus on testing, which will run constantly to validate that all tests are working and the main flow is functioning as expected

Resources

Personal resources:
We will need at least two QA engineers to implement these tests. This will help ensure continuous delivery and integration, as well as bug tracking and documentation across all the sprints. Having everything well-documented will provide a strong foundation for future team members.
Technical resources:
Credentials stored in the system Technologies that closely match actual technologies to facilitate implementation and avoid potential time loss

Environment

Preferably Windows environment.
Node.js and WebdriverIO and TypeScritp for automation.
Use Jenkins, TestRail, and GitHub Actions for CI/CD.
GitHub or GitLab for version control.
IDE for software development.
Jira for bug tracking and test case creation.
Zephyr or TestRail to store test cases and execution history.

Please make a note of the following information:

- Testing will commence on July 10, 2024.
- The estimated duration of the tests is 10 days.
- Testing will conclude on July 20, 2024.

However, please note that these dates are subject to change based on the project and specific circumstances. The intention is to always keep the testing active. This schedule accounts for the time needed to implement the framework from scratch and the ongoing work to maintain and deliver new behaviors with each release.

Daily working idea

Day	US	state	Actividades por día
1 Analysis and implementation design	*Create a new widget for a existing enterprise	DONE	Create the POM if doesn't exist yet Create the gherkin scenario
2 Construction of the solution		DONE	Get the selector to interact with the webelement in order to automate the behavior Create the methods to work with this selectors
3-4 Solution development		DONE	Implement the step definition and run the test
Total time	hours	DONE	The task is automated