Frontend Testing Libraries



Edita Pronckute | S8-Graduation Internship | 2023 April

# Frontend Testing

There is always a lot of emphasis on testing the software, but not so much the frontend part which is what the user is visually seeing and interacting with. Commonly, this gets tested with via clickthrough of a pre-set staging environment before releasing to end-users.

This is also the case in Drieam, where first line of live testing is the developers reviewing and clicking through each other’s PRs (Pull Requests). The second line is for the PO and/or the UX designer to try and break the interface. However, just like backend has unit tests and automated integration tests via pipelines (actions), so should the frontend. For this reason, the frontend unit tests are written for each component of the interface. It is a relatively new practice and thus causes a lot of discussions within the company.

A frontend unit test should validate that the smallest possible module (unit) is functioning as expected independent of the other modules. They can be split into state-based and interaction-based, meaning it either checks if the state of an element changed as expected or if an interaction is calling the right methods/making the right changes. Drieam uses a Jest library for front-end testing. It is a simple and lightweight JavaScript testing framework that is well documented and can run tests in parallel making it fast too. Since the tests are run periodically, they are automated using Cypress integrated in the CI actions (pipelines). Thus, each time the code is pushed, merged, released the cypress will run all the unit tests to ensure the application is still meeting expectations.

This document is a glance at potential testing solutions (libraries and frameworks) their benefits and weaknesses.

# Common testing problems

Changing UI – upgrades to core libraries or their components requires quick response and test updates too.

Flaky tests

Execution duration

Mocking

# Testing tools

The following Testing libraries and/or frameworks are widely popular for frontend unit tests. It is beneficial to know a bit about them as well as their pros and cons.

A picture containing carmine, red, clipart, graphics

Description automatically generated **Jest** – is the framework used by Drieam, as it has a strong community and active support. It is still the most popular framework (used by Facebook and officially supported by React dev team). It runs rather fast (as it supports parallel testing) and does not require separate assertion libraries or expensive configs creation.

**A brown hexagon with a cup of coffee

Description automatically generated with medium confidence Mocha** – a very flexible and customizable framework running on node.js and in browser, however, it is known to require importing other libraries to write unit tests.

Jasmine (software) - Wikipedia **Jasmine** – is another node.js and browser testing framework, it does not require external dependencies, however, together with Karma test runner it is a rather default option for Angular projects.

**A lightning bolt and check mark

Description automatically generated with low confidence Vitest** – is another reasonable option as the entire frontend is run via Vite, thus, sharing one pipeline (action), same plugins and vite.config.js. to run tests might speed things up.

The following libraries are mostly used for the DOM structure rendering for the Component unit tests or Integration tests:

Enzyme or Testing Library.

E2E testing tools? ->

Cypress – is a browser-based test suite and is already used in Portflow actions for E2E and component testing.

TestCafe for E2E

# Conclusion

Nowadays, most unit test frameworks can do the job well, the choice is either a personal preference or simply the one that is most popular and matches best with the rest of the test suite…. Or sth like that

The price of flexibility is complexity, keeping it simple is a challenge

Many more like Playwright, Pupeteer, Storybook etc.

# References

Błaszyński, Ł. (2021, December 3). *Front end testing frameworks for React Applications in 2022*. Retrieved from Espeo Software: https://espeo.eu/blog/front-end-testing-frameworks-2022/

Bose, S. (2023, March 16). *Front End Testing: A Beginner’s Guide*. Retrieved from BrowserStack: https://www.browserstack.com/guide/front-end-testing

G., A. (2022, June 7). *Vitest (Unit Testing) To Test React Application*. Retrieved from Medium: https://waresix.engineering/vitest-unit-testing-to-test-react-application-177ade1e6c1b

Sanders, M. (2021, August 25). *Challenges and Types of Front End Testing*. Retrieved from Binmile: https://binmile.com/blog/challenges-and-types-of-front-end-testing/

Taleb, M. (2022, October 6). *JavaScript unit testing frameworks in 2022: A comparison*. Retrieved from Raygun: https://raygun.com/blog/javascript-unit-testing-frameworks/

Vitest guide. (2023, March 5). *Comparisons with Other Test Runners*. Retrieved from Vitest: https://vitest.dev/guide/comparisons.html