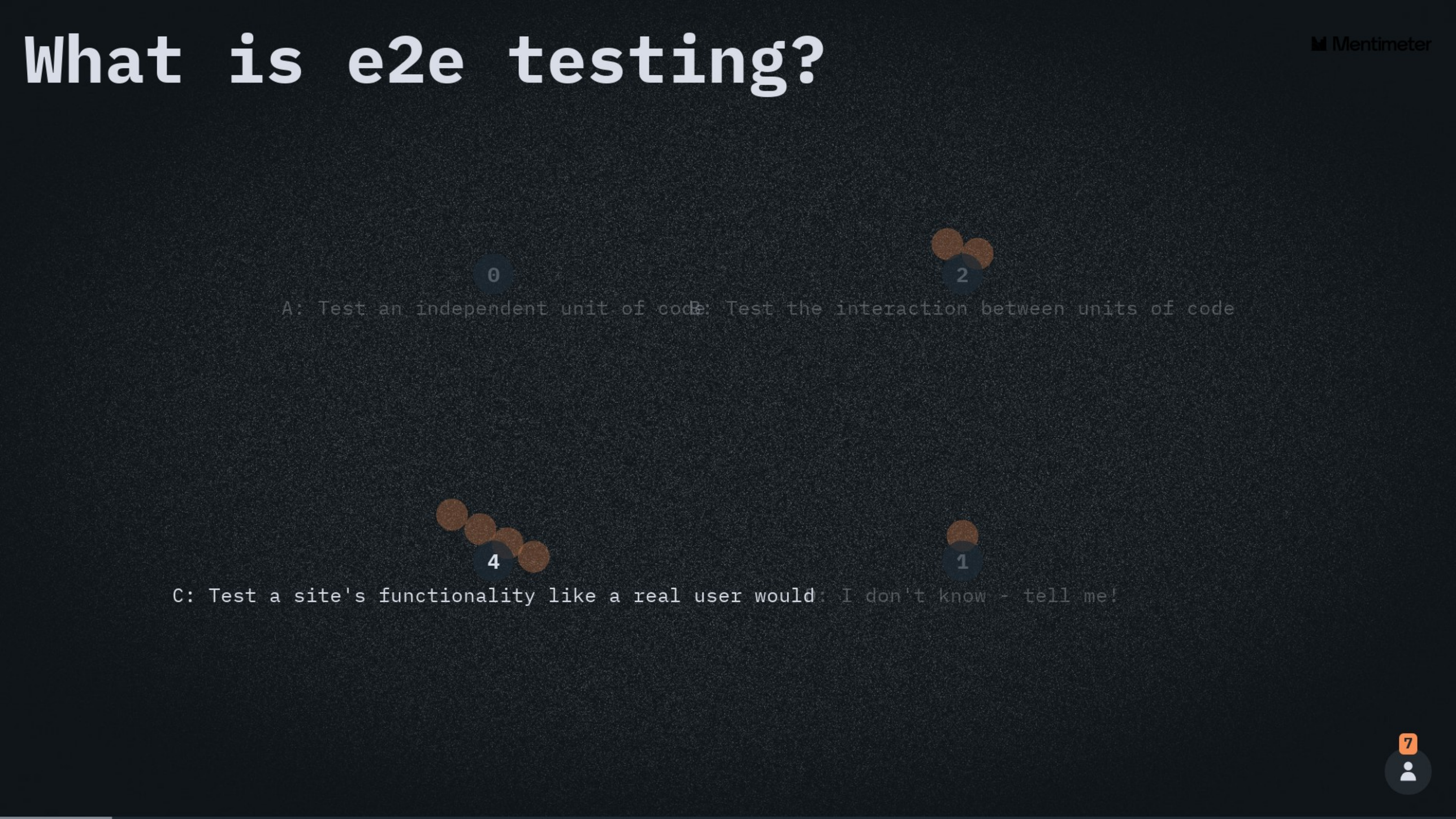
E2E TEST Your App With Cypress.io

In this presentation:

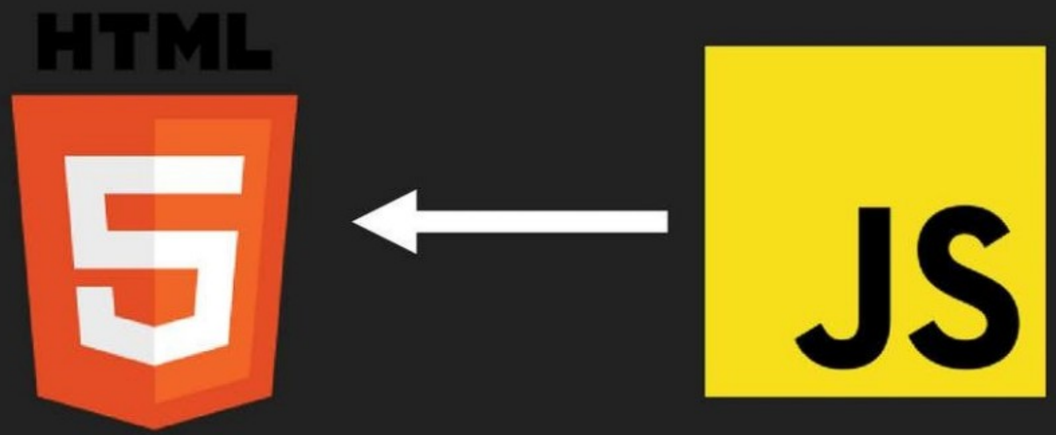
- What are E2E tests?
- How Cypress solves the problems old frameworks face
- How to start writing Cypress today

What is e2e testing?



Selenium

Injecting JavaScript into web pages




MakeAGIF.com

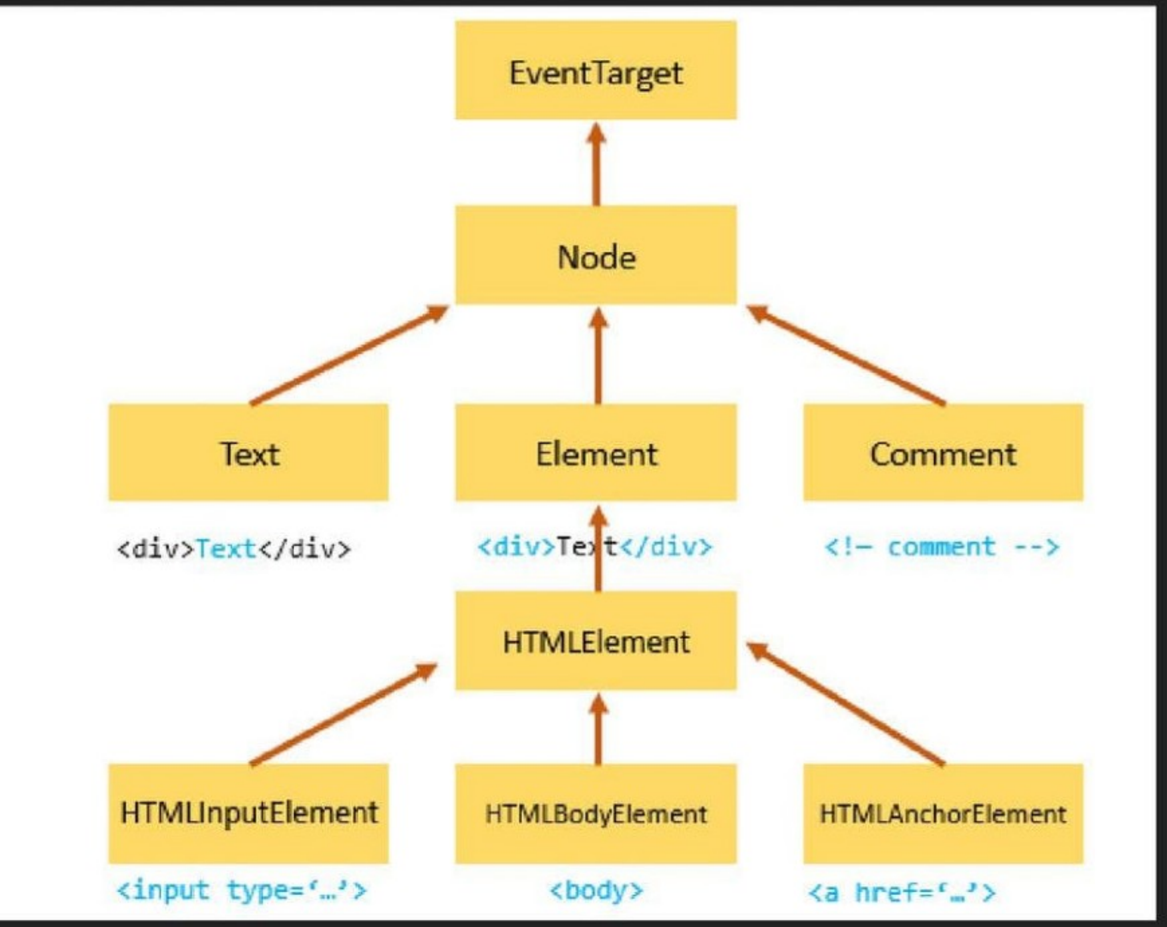
more dynamic = less reliable

Designers

Testers

findElementById(“comment”)

...

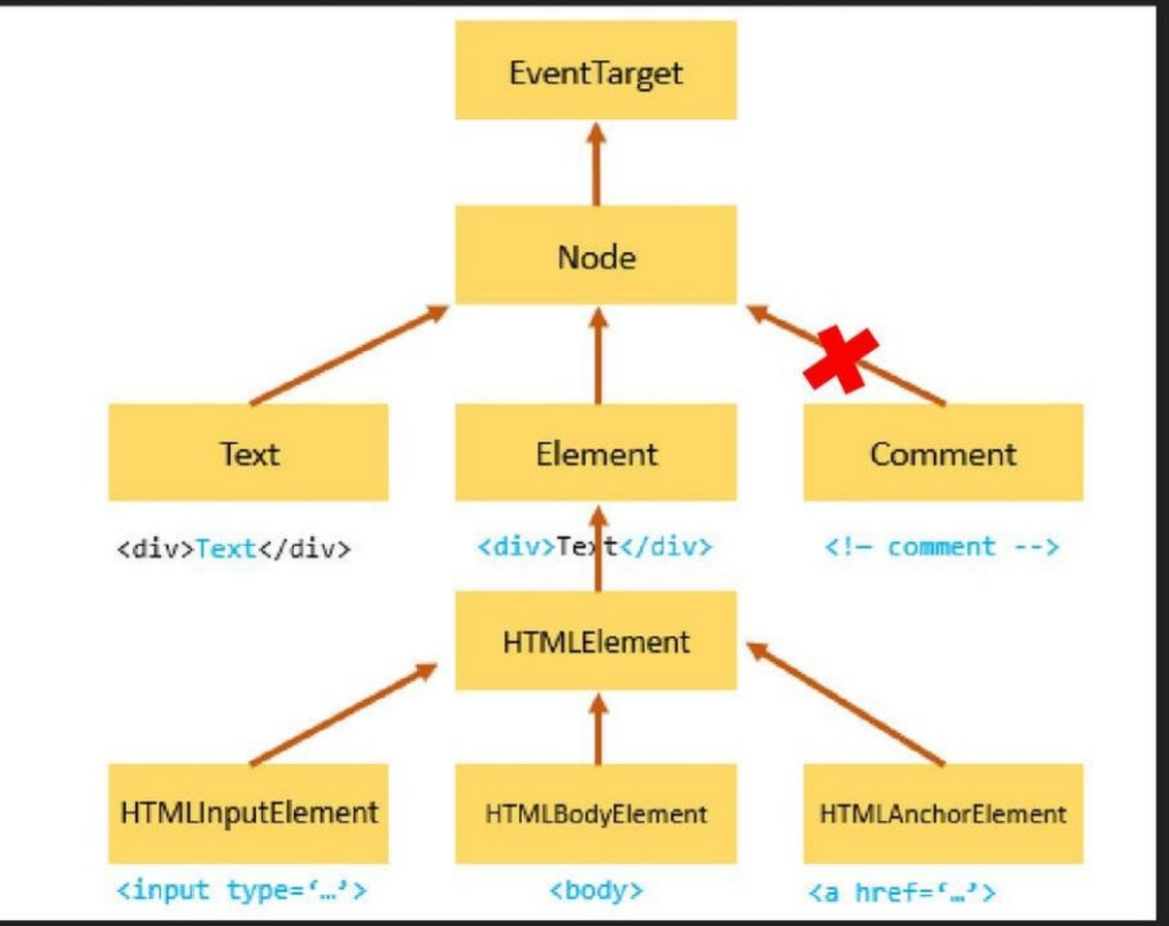


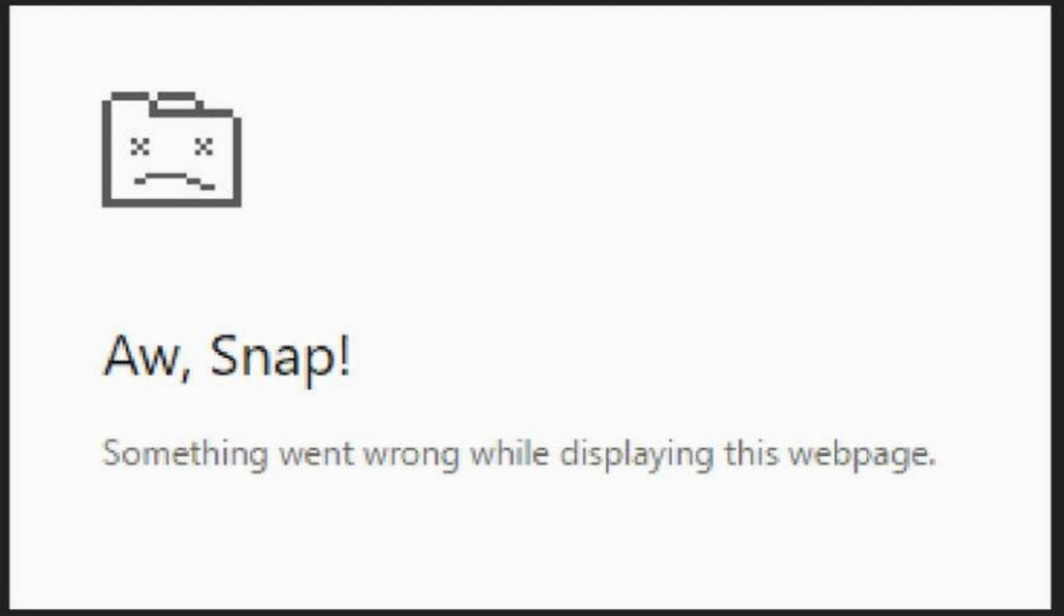
Mentimeter

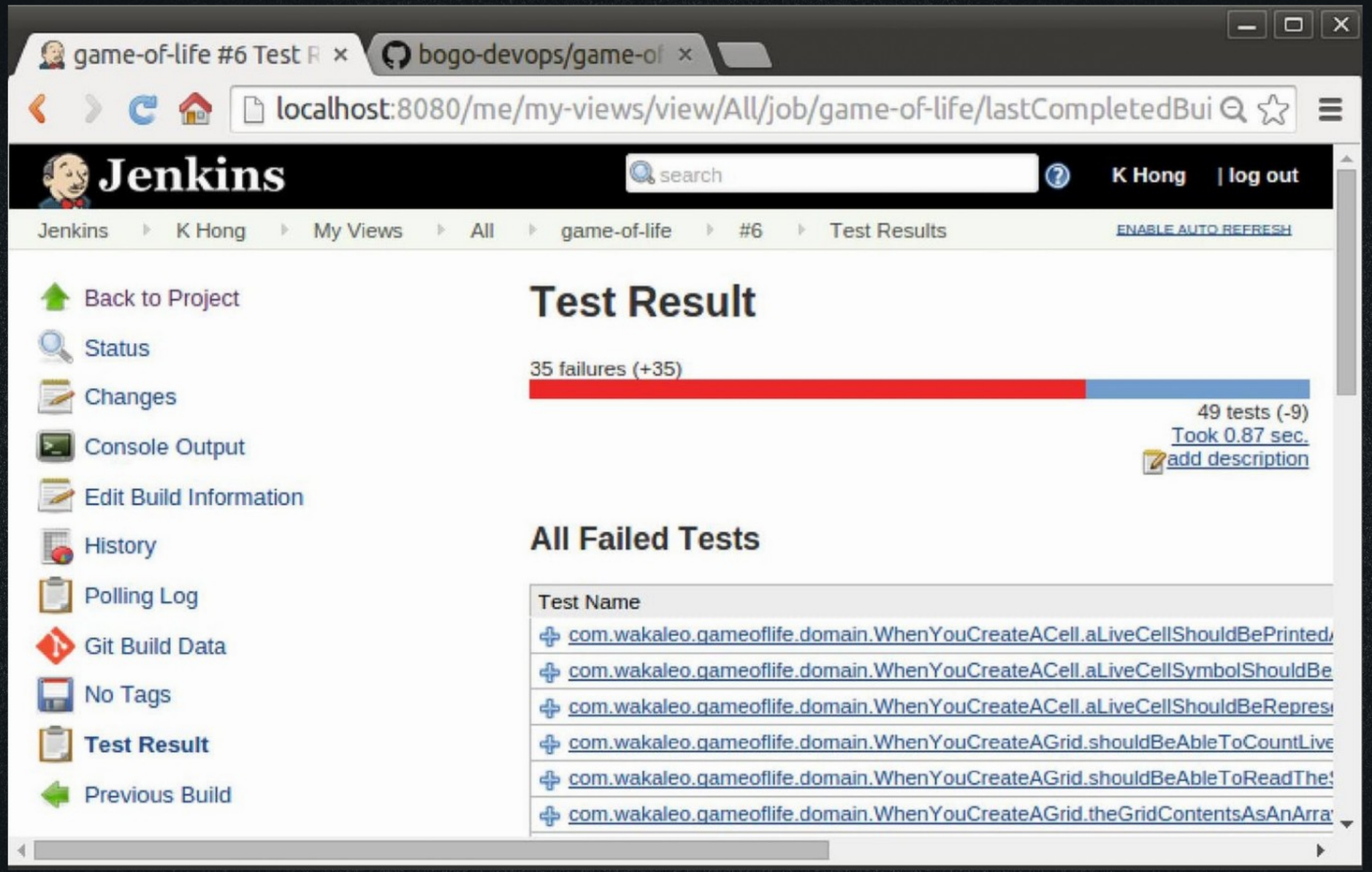
findElementById(“comment”)

...

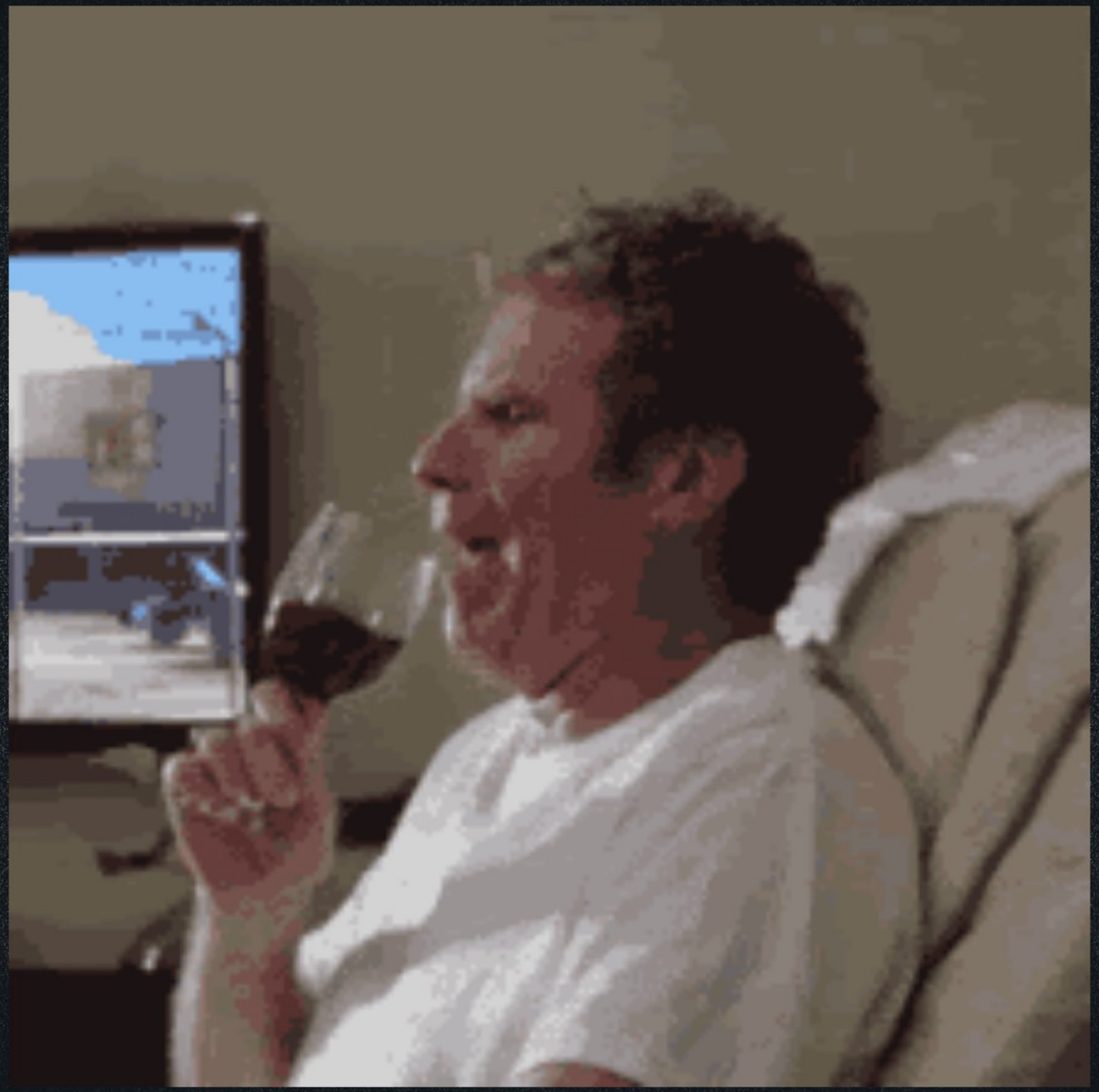
element.click()







This is very common 😰



CYPRESS

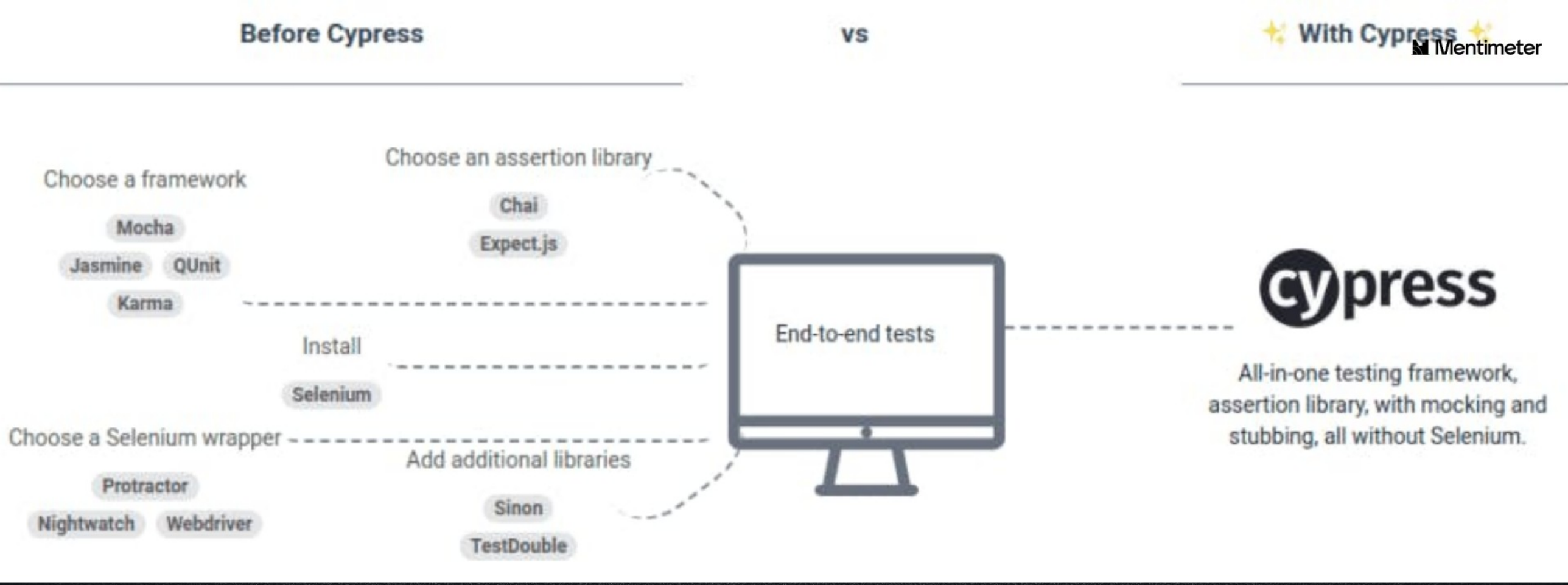


ILLUSTRATION FROM: HTTPS://WWW.CYPRESS.IO/HOW-IT-WORKS/

All features in one framework

But ease of use is not even the best part!

JS everything

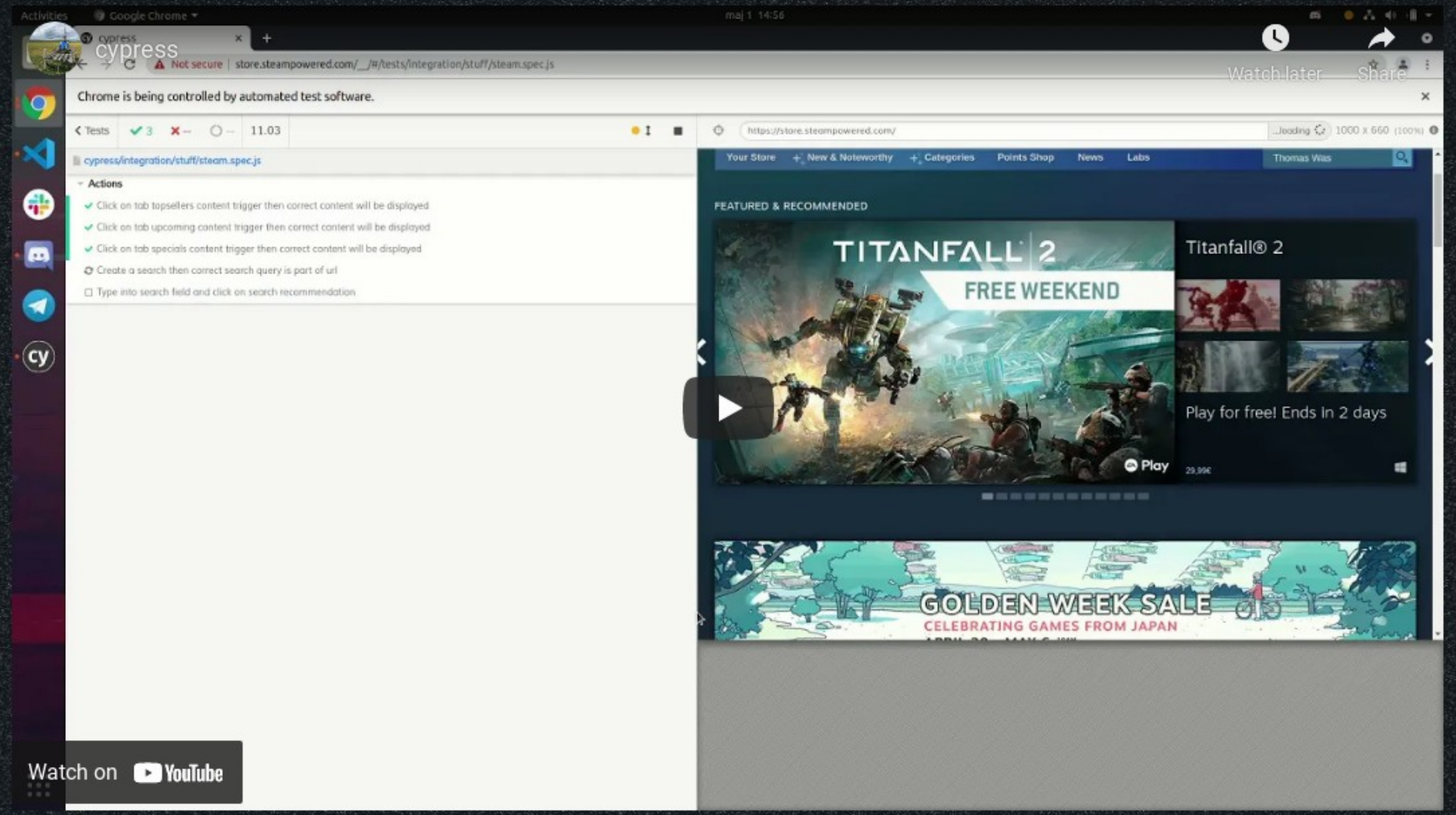
Cypress is built using a mix of battle-tested testing libraries and novel software design. All in JavaScript.

Mentimeter

Run fast!

Not flaky!

5



3

Cypress has a dedicated team and a great community

Get up to speed quickly:

→ yarn add cypress -d

→ yarn run cypress open

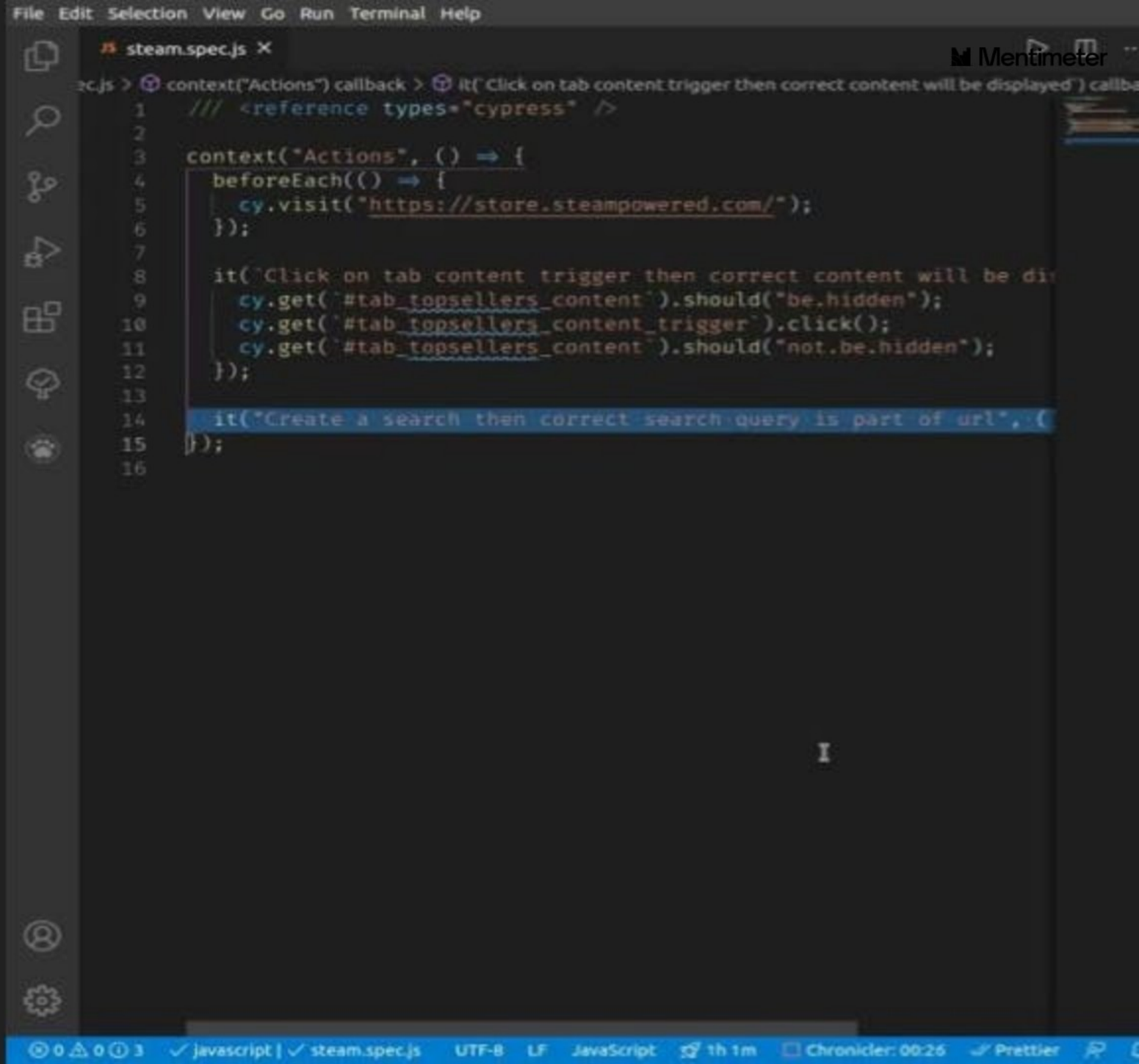
```
describe('My First Test', () => {
  it('Does not do much!', () => {
    expect(true).to.equal(false)
  })
})
```

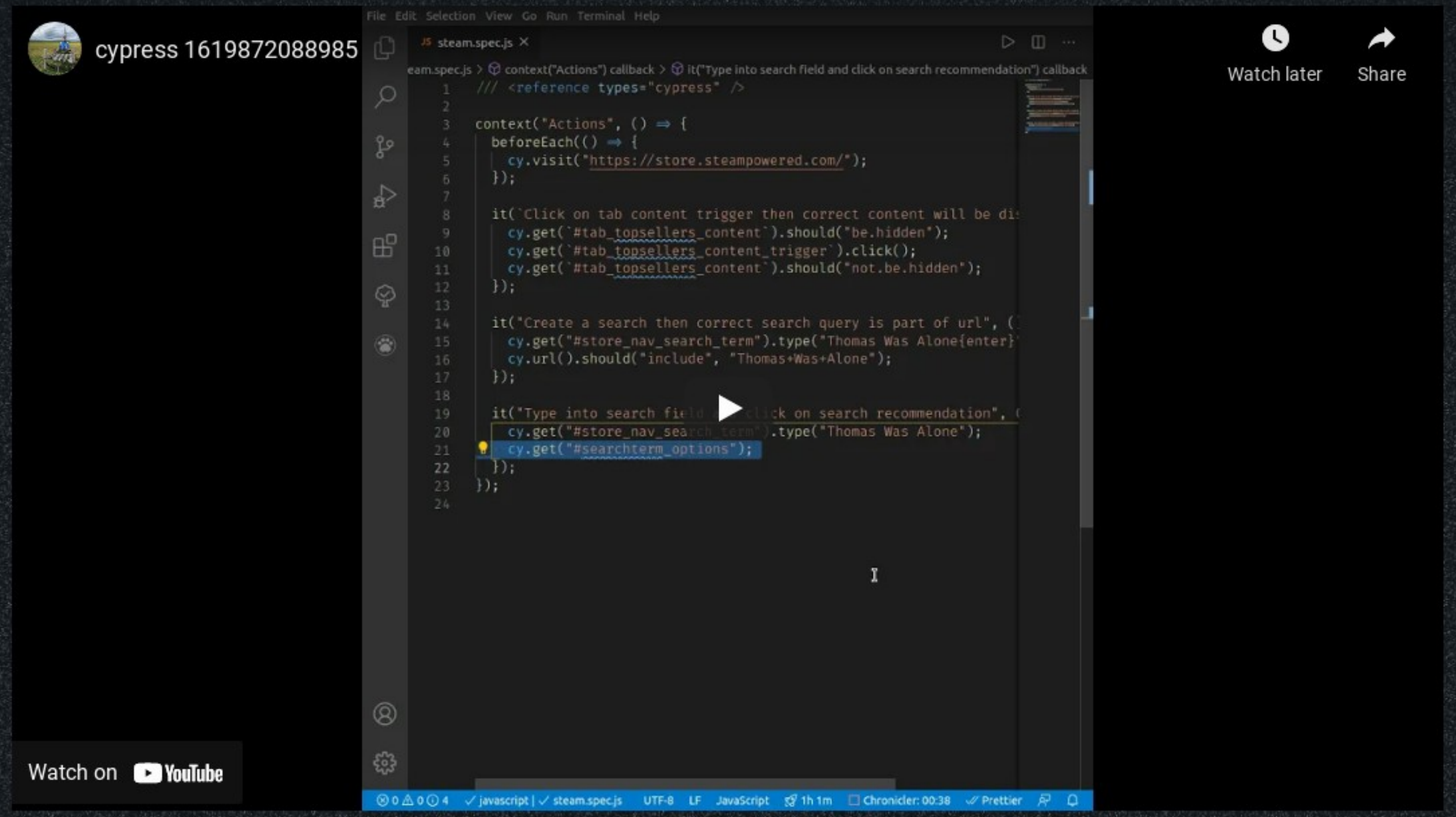
Testing the Steam store

www.store.steampowered.com/

We are testing:

- Content offering tabs
- Store search
- Store search recommendations





CREATING TESTS FOR THE STEAM STORE

```
describe("landing page", () => {
  it("Search for non-existing item return nothing", () => {
    cy.get("#login-email-input").type("user");
    cy.get("#login-password-input").type("pass");
    cy.get("#login-submit").click();

    . . .
  });

  it("Search for existing item return result", () => {
    cy.get("#login-email-input").type("user");
    cy.get("#login-password-input").type("pass");
    cy.get("#login-submit").click();

    . . .
  });
});
```

This is slow!

(Even if we write it in a command or in beforeEach)

You want to set application state programmatically

You can design your app for testability

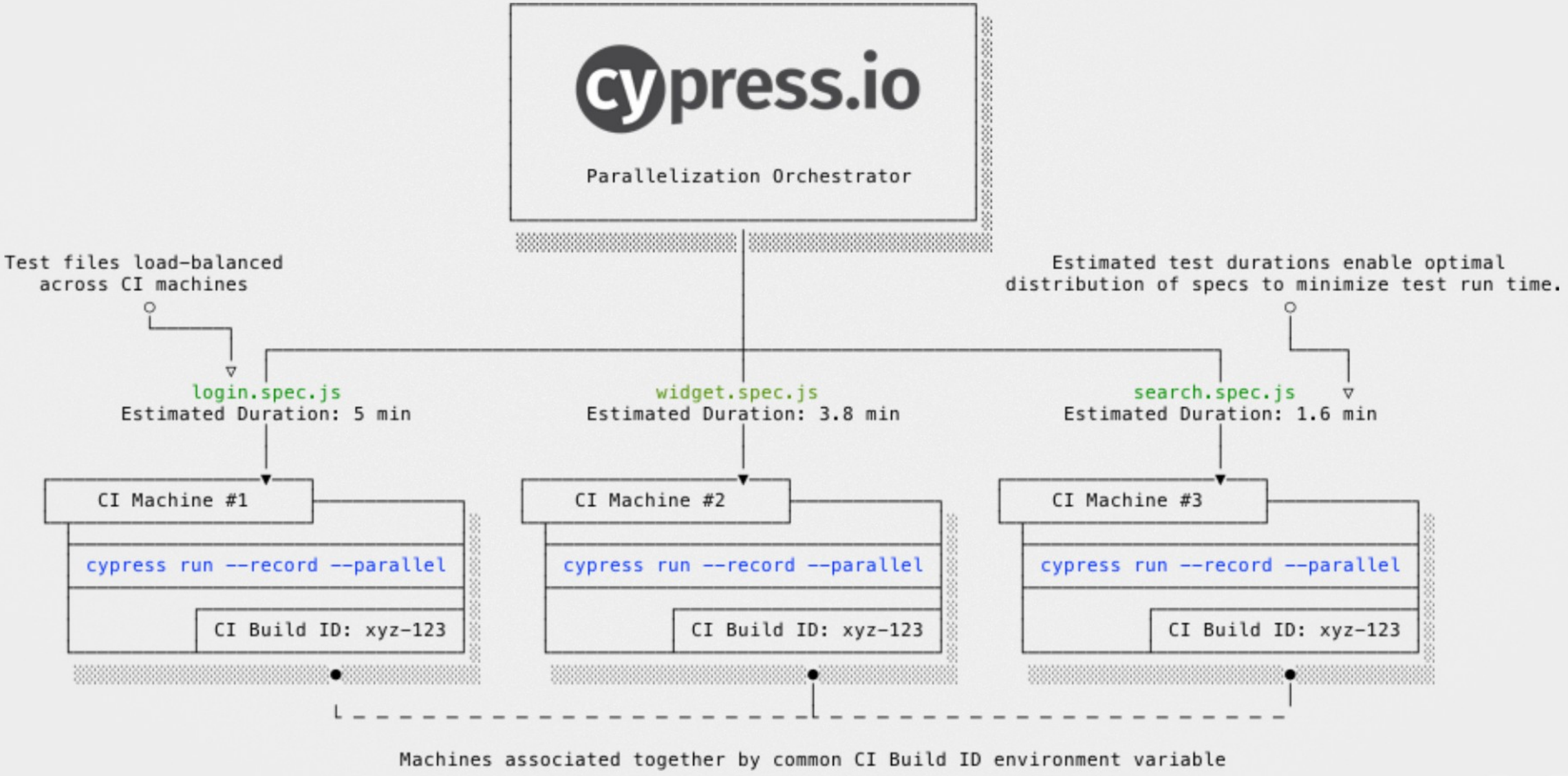
```
// index.js
if (process.env.NODE_ENV === 'test'){
  dbURI = process.env.TEST_DB_URI
}
db.connect(dbURI);
```

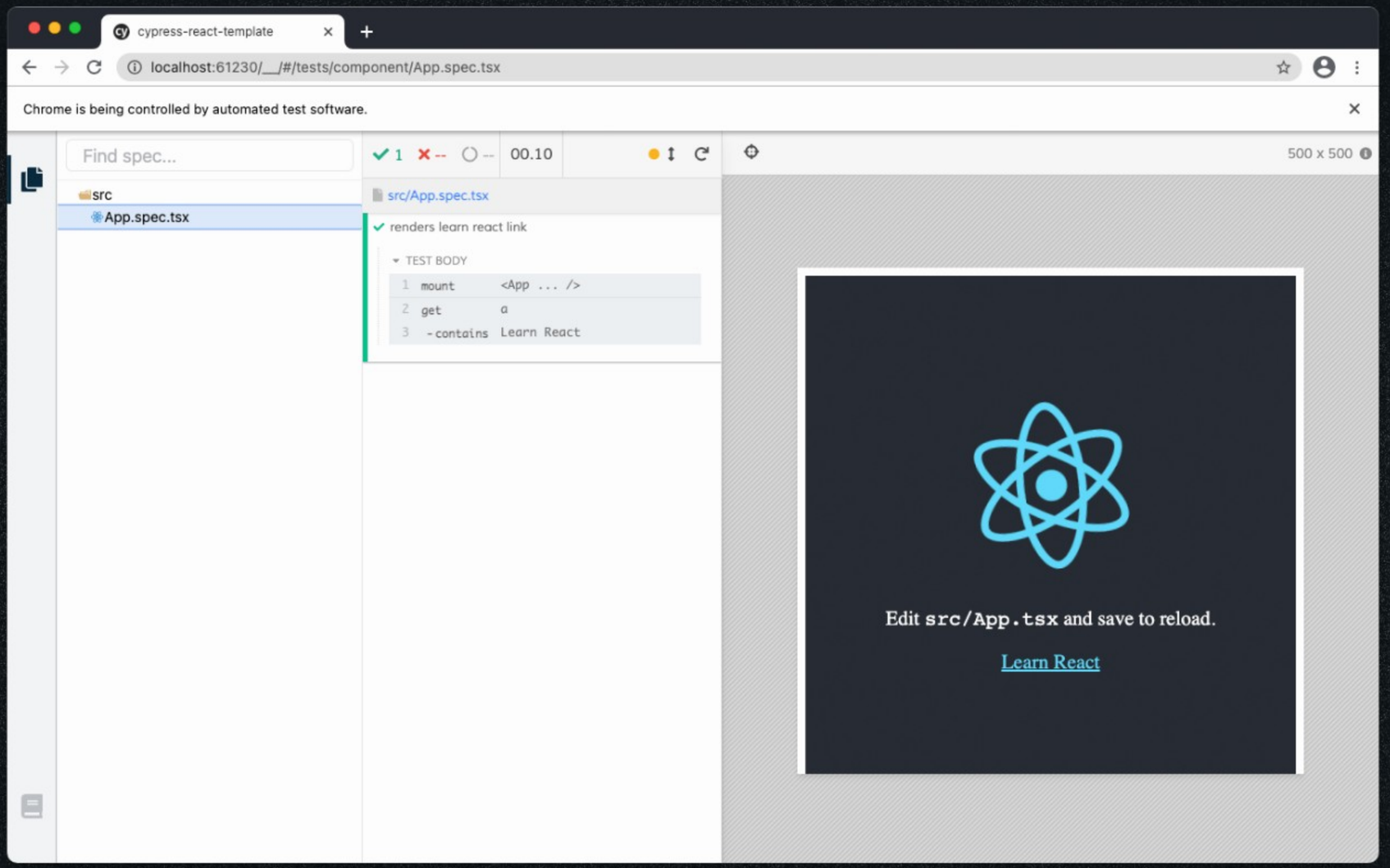
```
// cypress/plugins/index.js
module.exports = async (on, config) => {
  const db = await DB.connect();
  on('task', {
    async createUser(data) return db.insert(data);
  })
}
```

```
// docker-compose.test.yml

app:
  build: Dockerfile.test
  environment:
    -    NODE_ENV=test
    -    TEST_DB_URI=${DB_USER}@db:5432
  depends_on:
    -    db
db:
  environment:
    -    DB_USER
  ports:
    -    5432:5432
```

Directly in your application, test, and environment configuration


cypress.io
Parallelization Orchestrator
Test files load-balanced across CI machines
Estimated test durations enable optimal distribution of specs to minimize test run time.
login.spec.js
Estimated Duration: 5 min
widget.spec.js
Estimated Duration: 3.8 min
search.spec.js
Estimated Duration: 1.6 min
CI Machine #1
cypress run --record --parallel
CI Build ID: xyz-123
CI Machine #2
cypress run --record --parallel
CI Build ID: xyz-123
CI Machine #3
cypress run --record --parallel
CI Build ID: xyz-123
Machines associated together by common CI Build ID environment variable



Cypress testing a React component directly

More things plugins help you do:

1. Create snapshot tests
2. Get the code-coverage of your app
3. Integrate Cypress with Slack and Jest
4. More. New plugins are released all the time.

CYPRESS: 3 to Take Home

1. e2e tests are **important** to be able to **assure the business logic and user flow** of any application
2. Selenium is outdated. **Cypress is easy!**
3. Just do a **npm install cypress --save-dev** and get up to speed with Cypress today!

Thank you

& Happy testing!