End-to-End (E2E) Testing.

(aka System Testing)

Overview

- Testing the entire system as a whole.
  (UI + Server-side + Database)

- Concerns:
  - Functionality. ****
    - From the USER interface perspective.
  - Performance.
  - Load/Stress.

Overview

- <u>Blackbox</u> approach – we are not focused on the internals of the system; the only concern is expected response for specific inputs.

- Preceded by Unit and subsystem (e.g. web API) testing to resolve ‘internal’ errors.

- May also be interested in <u>side-effects</u>, e.g. database changes.

- The Asynchronous nature (for web/mobile apps).

Web App Target

- Web apps - Targeting the browser interface.
  - Functionality.
  - Form submits.
  - Navigation.
  - Flows e.g. shopping cart checkout.
  - Visual testing (CSS).

Automation Tool Suite

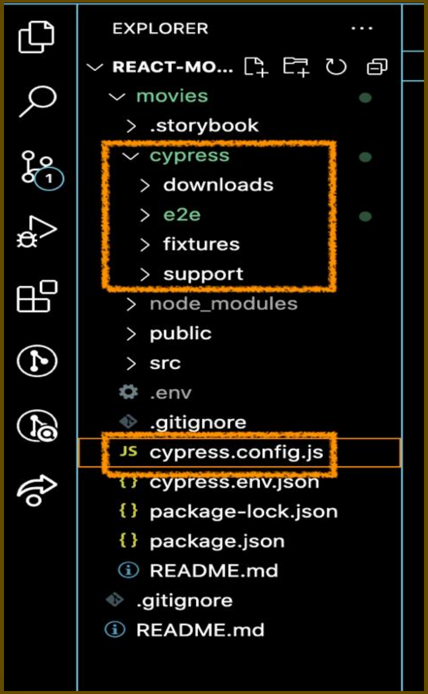
- Traditional tool suite: Mocha + Chai + Selenium.

- Modern tool suite: Cypress.
  - Uses Mocha and Chai internally.

- Cypress.
  - Win / Mac / Linux.
  - MIT License.
  - Open Source.
  - Not suitable for Visual testing; Use Percy instead.

Cypress - Overview

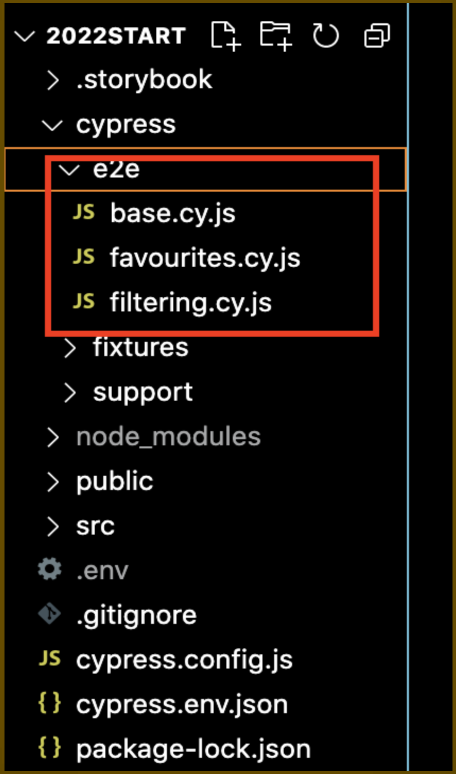
- Getting started: $ npm install –save-dev cypress



```
EXPLORER
REACT-MO...
  movies
    .storybook
    cypress
      downloads
      e2e
      fixtures
      support
    node_modules
    public
    src
    .env
    .gitignore
    cypress.config.js
    cypress.env.json
    package-lock.json
    package.json
    README.md
  .gitignore
  README.md
```

- cypress folder
1. e2e– test code, termed specs.
2. fixtures – sample data.
3. support – utility code.

- **cypress.config.js** – config file.

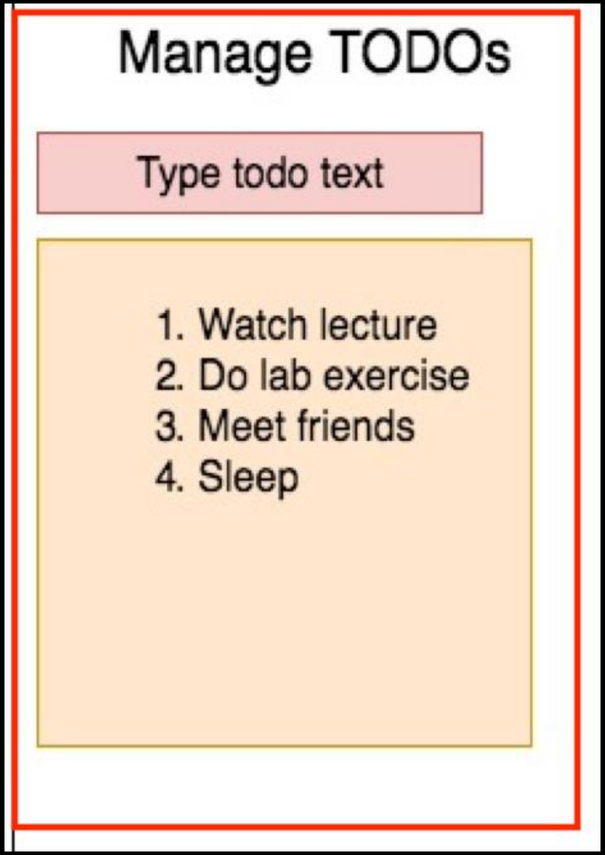


```
2022START
  .storybook
  cypress
    e2e
      base.cy.js
      favourites.cy.js
      filtering.cy.js
    fixtures
    support
  node_modules
  public
  src
  .env
  .gitignore
  cypress.config.js
  cypress.env.json
  package-lock.json
```

- CLI has two main commands:

```
$ npx cypress open  # GUI interactive mode
$ npx cypress run   # headless mode.
```

Sample Test Code.



```
describe(“TODO app", () => {
   it(‘should add 2 todos', () => {
     cy.visit('http://localhost:3000')
     cy.get(‘input')
        .type(‘Watch lecture{enter}')
        .type(’Do lab exercise{enter}')
     cy.get('li')
        .should('have.length', 2)
   })
})
```

- **Declarative style.**
- **Method Chaining style e.g.**
  cy.get(…).type(…)

Cypress statements.

- cy.get(..selector..).should(..expectation..)

  Command Expectations (Optional)

- Commands – All about accessing and interacting with browser DOM elements.
  - get(selector) - Get the DOM elements that match the selector.
  - contains(text) - Get the DOM element that contains the text, e.g. cy.contains(‘Add’)
  - find(selector) - Get the <u>child</u> DOM element(s).

    e.g. Fund the dropdown menu inside the web form and select the Medium option.

  cy.get(‘form').find(‘select’).select(‘Medium’)

Cypress statements.

- Commands (Contd.) –
  - next() – get the next DOM element, e.g. cy.get('button’).next()
  - eq(n) – Get the nth DOM element in an array of elements, e.g.

    cy.get(‘input’).eq(2).type(‘1 Main Street’);

    cy.get('li’).eq(4).should(‘equal’, ‘Agile’)

See https://docs.cypress.io/api/table-of-contents

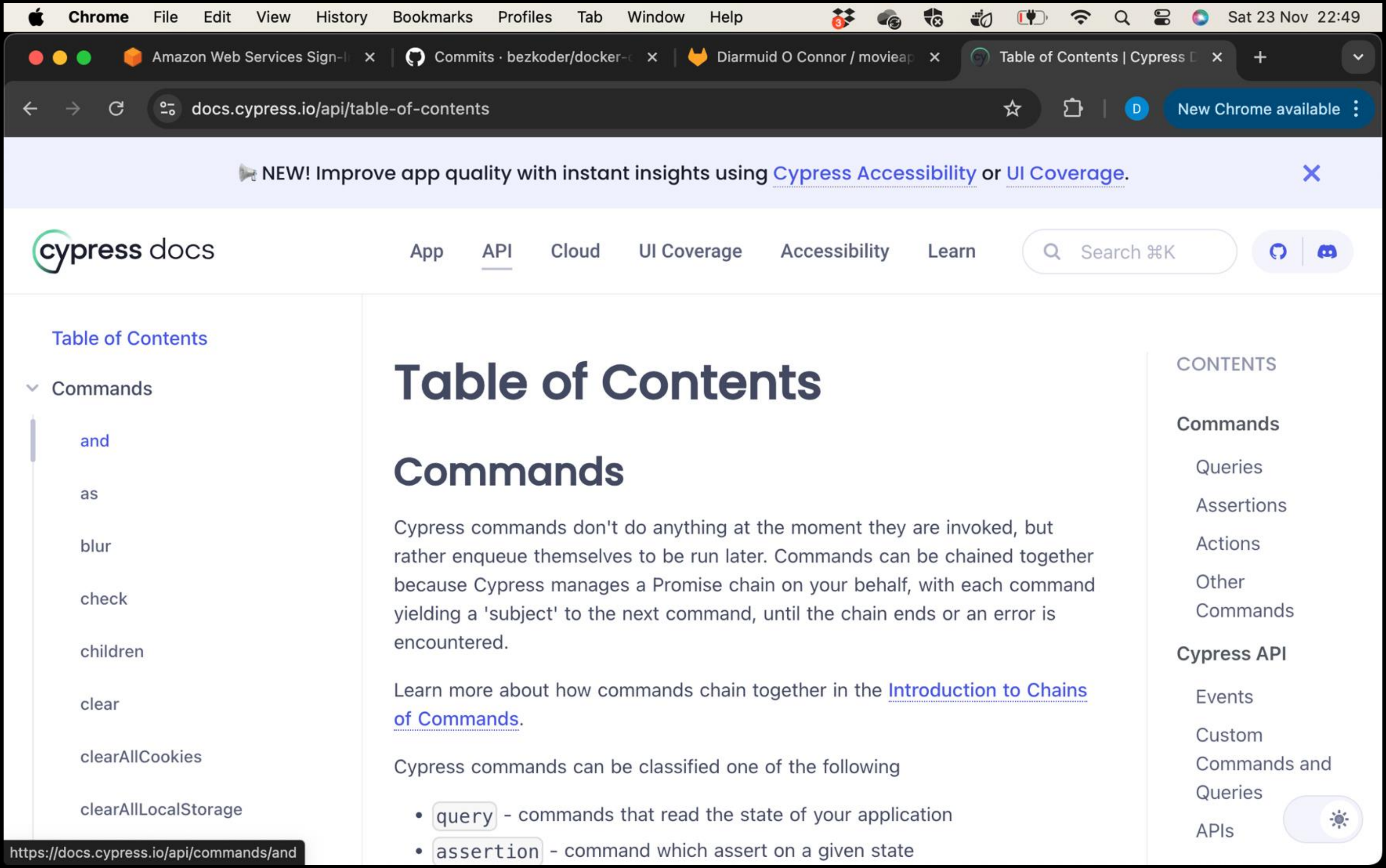

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Sat 23 Nov 22:49
Amazon Web Services Sign-I
Commits · bezkoder/docker-
Diarmuid O Connor / moviea
Table of Contents | Cypress D
docs.cypress.io/api/table-of-contents
New Chrome available
NEW! Improve app quality with instant insights using Cypress Accessibility or UI Coverage.
cypress docs
App API Cloud UI Coverage Accessibility Learn
Search ⌘K
Table of Contents
Commands
and
as
blur
check
children
clear
clearAllCookies
clearAllLocalStorage
https://docs.cypress.io/api/commands/and
CONTENTS
Commands
Queries
Assertions
Actions
Other Commands
Cypress API
Events
Custom Commands and Queries
APIs

Table of Contents

Commands

Cypress commands don't do anything at the moment they are invoked, but rather enqueue themselves to be run later. Commands can be chained together because Cypress manages a Promise chain on your behalf, with each command yielding a 'subject' to the next command, until the chain ends or an error is encountered.

Learn more about how commands chain together in the Introduction to Chains of Commands.

Cypress commands can be classified one of the following

- `query` - commands that read the state of your application
- `assertion` - command which assert on a given state

Cypress statements - Selectors.

cy.get(selector).should(expectation)

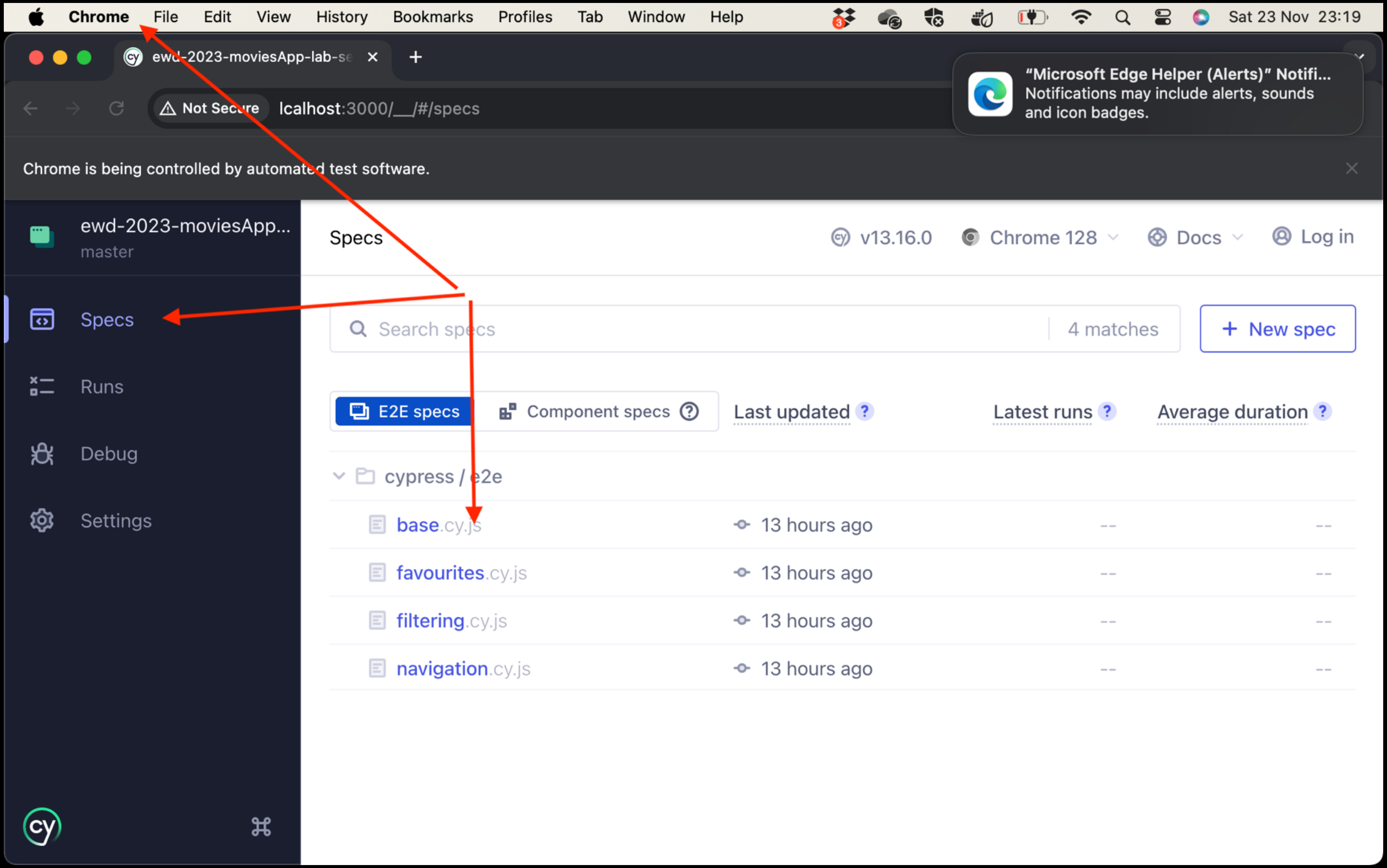
- Selector: Based on JQuery syntax.
  - HTML Tag type: e.g. cy.get(‘button’)
  - Elemen Id: e.g. cy.get(’#heading’)
  - CSS Class: e.g. cy.get(’.info-message’)
  - Attributes, e.g. cy.get('button[type=submit]').click()
  - nth-child, e.g. get the 8th column of the 3rd row in a table

  cy.get('tbody').find(‘tr’).eq(2).find(‘td’).eq(7)
  - Selectors can be combined, e.g. cy.get(‘div.container’) (the div tag with CSS class .container)

Test Runner

- Main features:
  - Tests run inside the browser.
  - Test code has Full access to browser’s resources, e.g. DOM, cookies, local storage.
  - Web-App-Framework-agnostic.
  - Flake-free test execution.
  - Deterministic, repeatable, consistent execution flow.
  - Auto retries commands to accommodate slow DOM construction.
  - Deals with asynchronous nature of the web.

$ npx cypress open



Chrome File Edit View History Bookmarks Profiles Tab Window Help Sat 23 Nov 23:19

ewd-2023-moviesApp-lab-se

Not Secure localhost:3000/__/#/specs

"Microsoft Edge Helper (Alerts)" Notifi...
Notifications may include alerts, sounds and icon badges.

Chrome is being controlled by automated test software.

ewd-2023-moviesApp...
master

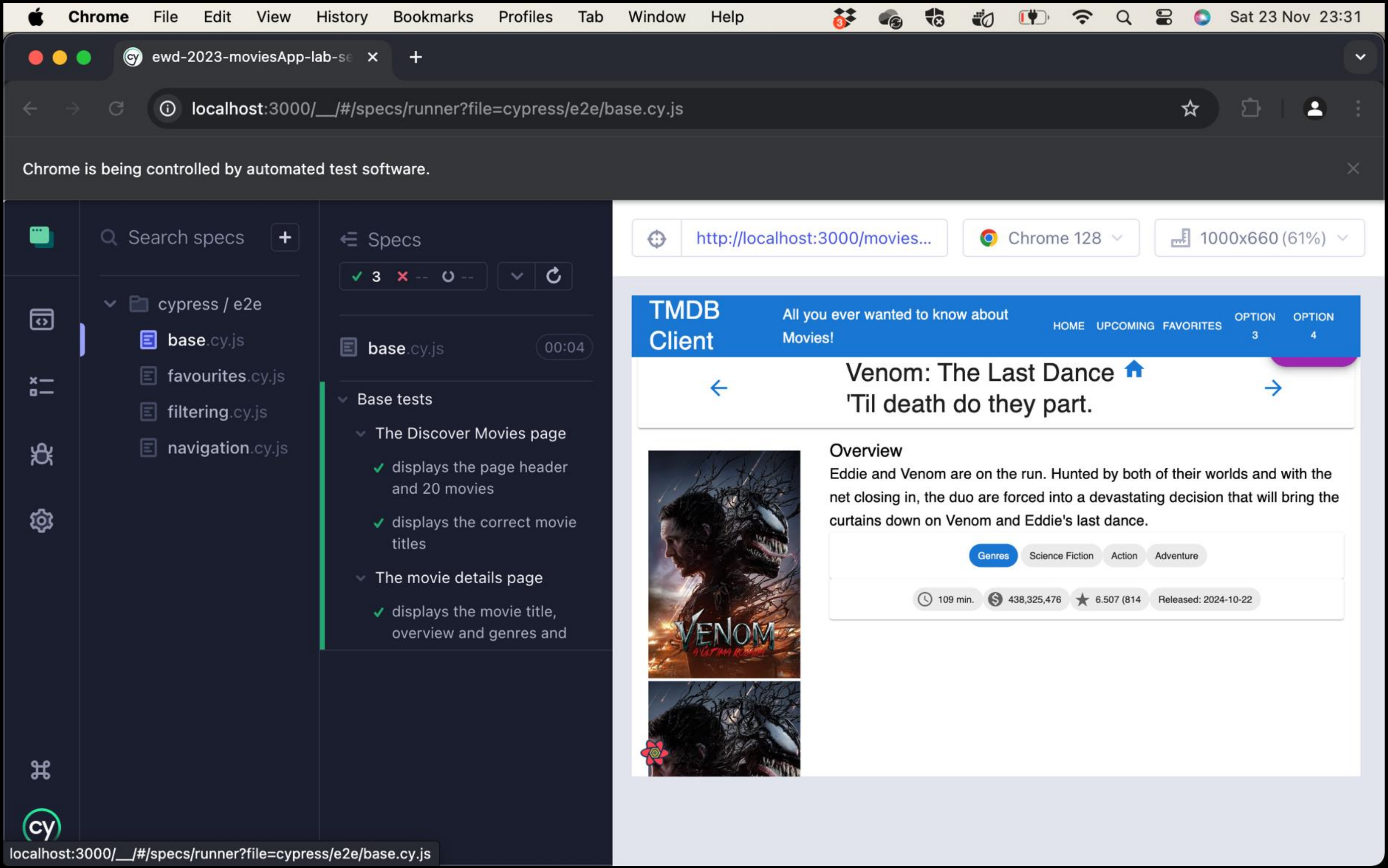
Specs v13.16.0 Chrome 128 Docs Log in

Search specs 4 matches + New spec

E2E specs Component specs

	Last updated	Latest runs	Average duration
cypress / e2e			
base.cy.js	13 hours ago	--	--
favourites.cy.js	13 hours ago	--	--
filtering.cy.js	13 hours ago	--	--
navigation.cy.js	13 hours ago	--	--

Test Execution
(Green bar / Red Bar)



Chrome File Edit View History Bookmarks Profiles Tab Window Help   Sat 23 Nov 23:31

ewd-2023-moviesApp-lab-se

localhost:3000/__/#/specs/runner?file=cypress/e2e/base.cy.js

Chrome is being controlled by automated test software.

base.cy.js 00:04

Base tests
- The Discover Movies page
  - ✓ displays the page header and 20 movies
  - ✓ displays the correct movie titles
- The movie details page
  - ✓ displays the movie title, overview and genres and

http://localhost:3000/movies...
Chrome 128
1000x660 (61%)
TMDB Client
All you ever wanted to know about Movies!
HOME UPCOMING FAVORITES OPTION 3 OPTION 4

Venom: The Last Dance
'Til death do they part.

Overview

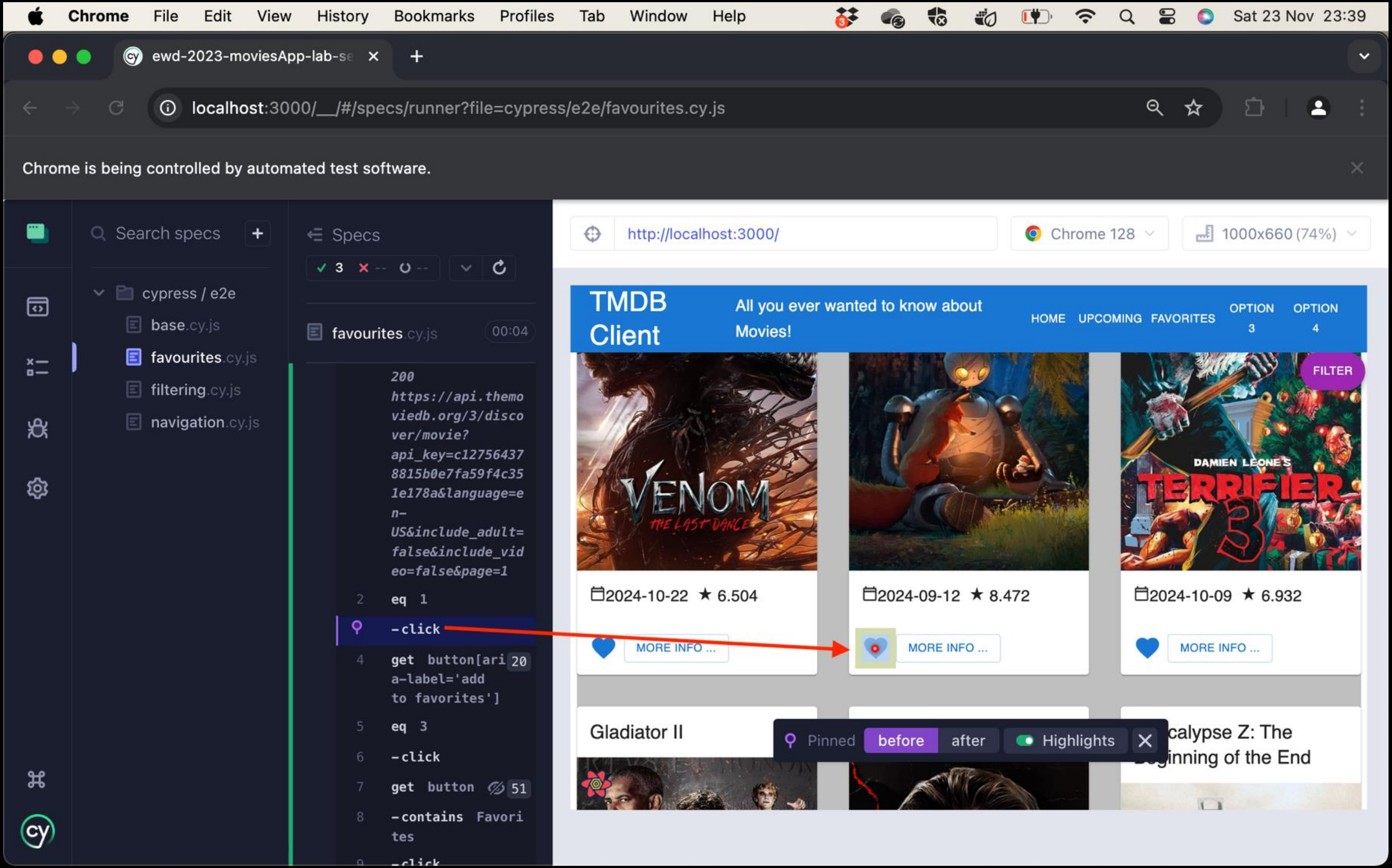
Eddie and Venom are on the run. Hunted by both of their worlds and with the net closing in, the duo are forced into a devastating decision that will bring the curtains down on Venom and Eddie's last dance.

Genres | Science Fiction | Action | Adventure

109 min. | 438,325,476 | 6.507 (814 | Released: 2024-10-22

localhost:3000/__/#/specs/runner?file=cypress/e2e/base.cy.js

- Time-travel – **Step through test code to track the app's UI state. Great for debugging.**

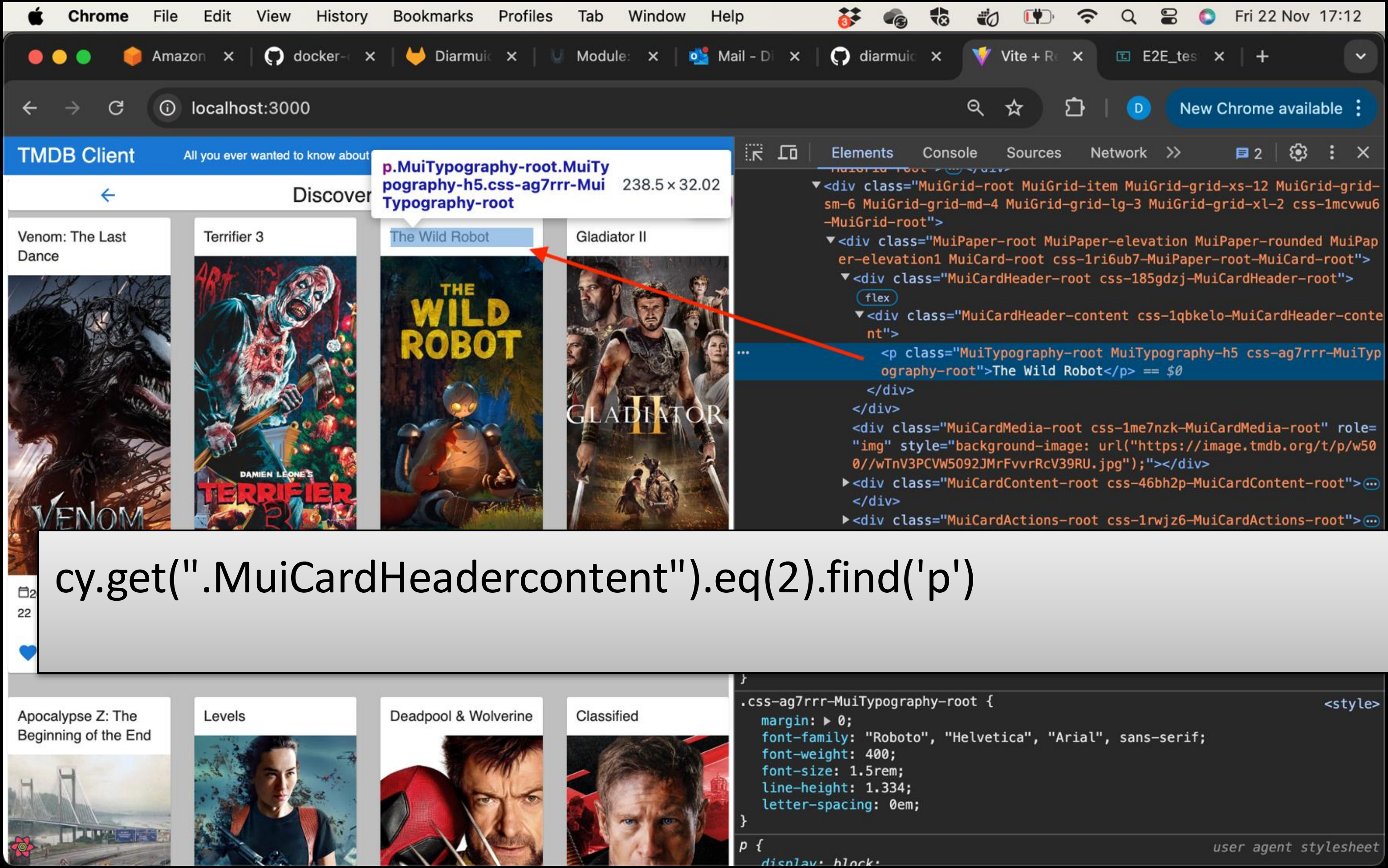


Chrome is being controlled by automated test software.

localhost:3000/__/#/specs/runner?file=cypress/e2e/favourites.cy.js

```
200
https://api.themoviedb.org/3/discover/movie?api_key=c127564378815b0e7fa59f4c351e178a&language=en-US&include_adult=false&include_video=false&page=1
eq 1
-click
get button[aria-label='add to favorites'] 20
eq 3
-click
get button 51
-contains Favorites
-click
```

- Use Chrome’s dev tools to help with choosing a selector for a command.


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Fri 22 Nov 17:12
localhost:3000
New Chrome available
TMDB Client
All you ever wanted to know about
Discover
p.MuiTypography-root.MuiTypography-h5.css-ag7rrr-MuiTypography-root 238.5 × 32.02
Venom: The Last Dance
Terrifier 3
The Wild Robot
Gladiator II
Apocalypse Z: The Beginning of the End
Levels
Deadpool & Wolverine
Classified
Elements Console Sources Network

```
▼<div class="MuiGrid-root MuiGrid-item MuiGrid-grid-xs-12 MuiGrid-grid-sm-6 MuiGrid-grid-md-4 MuiGrid-grid-lg-3 MuiGrid-grid-xl-2 css-1mcvwu6-MuiGrid-root">
  ▼<div class="MuiPaper-root MuiPaper-elevation MuiPaper-rounded MuiPaper-elevation1 MuiCard-root css-1ri6ub7-MuiPaper-root-MuiCard-root">
    ▼<div class="MuiCardHeader-root css-185gdzj-MuiCardHeader-root"> flex
      ▼<div class="MuiCardHeader-content css-1qbkelo-MuiCardHeader-content">
          <p class="MuiTypography-root MuiTypography-h5 css-ag7rrr-MuiTypography-root">The Wild Robot</p> == $0
        </div>
      </div>
      <div class="MuiCardMedia-root css-1me7nzk-MuiCardMedia-root" role="img" style="background-image: url("https://image.tmdb.org/t/p/w500//wTnV3PCVW5092JMrFvvrRcV39RU.jpg");"></div>
    ▶<div class="MuiCardContent-root css-46bh2p-MuiCardContent-root">…
      </div>
    ▶<div class="MuiCardActions-root css-1rwjz6-MuiCardActions-root">…
```

```
.css-ag7rrr-MuiTypography-root {                                   <style>
  margin: ▶ 0;
  font-family: "Roboto", "Helvetica", "Arial", sans-serif;
  font-weight: 400;
  font-size: 1.5rem;
  line-height: 1.334;
  letter-spacing: 0em;
}
p {                                                   user agent stylesheet
  display: block;
```

```
cy.get(".MuiCardHeadercontent").eq(2).find('p')
```

Headless Test Runner

- Running Cypress tests <u>without</u> a UI.
- Headless mode:

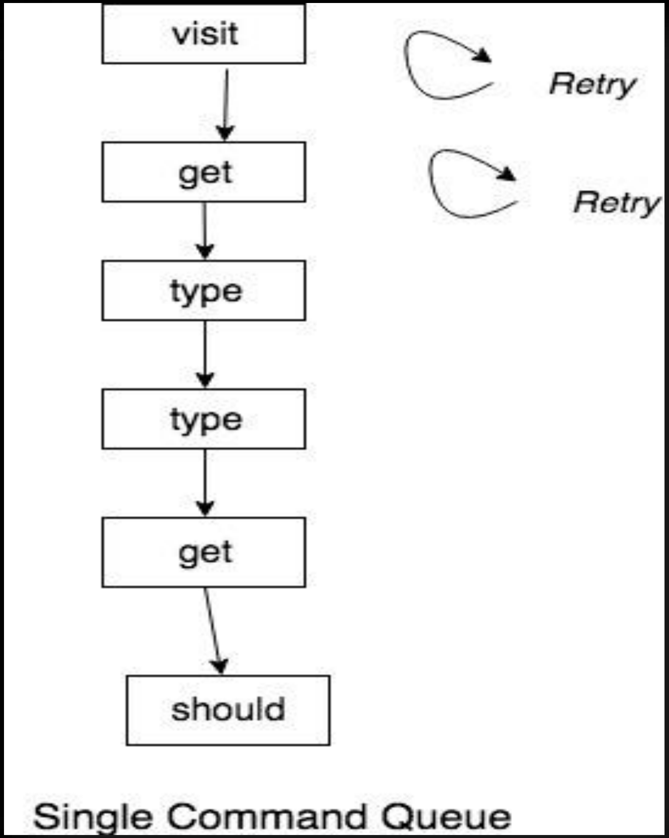
  ```
  $ npx cypress run
  ```
- Ideal for CI (Continuous Integration) environment.

- Can generates video recordings (disabled by default) and stored in *cypress/e2e/videos*.
  - .mp4 file type.
  - Facilitates sharing and project visibility.
- Lots of command line options, e.g.
  - --spec file_name_pattern
  - --config – override settings in cy.config.js
  - --record, --browser etc,

Commands are enqueued.

```
describe(“TODO app", () => {
 it(‘should add 2 todos', () => {
  cy.visit('http://localhost’)
  cy.get(‘input')
   .type(‘Watch lecture{enter}')
   .type(’Submit lab {enter}')
  cy.get('li')
   .should('have.length', 2)
 })
})
```

– Commands are first enqueued and then run serially in a controlled manner, i.e. retries, delays.
– Guarantees a deterministic or flake-free test behaviour.



Command chaining

- Chain of commands.

```
 cy.get("h4").next().contains(movie.overview);
 cy.get("li").eq(2).find(‘h3’)
               .should(‘contain’,’Waterford’)
```

- A chains always begins with cy.
- Each command yields a subject to the next one in the chain.
- We can act on a subject directly with .then()

```
 cy.get(‘div').eq(2).find(‘button')
   .then ( (buttonElement) => {
     onst cls = buttonElement.class()
     . . . . . . . .
   })
```

Summary

- E2E testing – aka System testing.

- Black-box mindset – does app produce expected output for a given input.

- Cypress – deterministic, repeatable, consistent test execution.
- Spec (specification) files.
- Test code structured according to Mocha framework.
- Commands – mainly concerned with querying the DOM and interacting with elements.
- Assertions/Expectations - built on Mocha and Chai libraries.