



**BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC NGUYỄN TẤT THÀNH
VIỆN ĐÀO TẠO QUỐC TẾ NTT (NIIET)**

Giảng viên hướng dẫn:
TS. Trần Sơn Hải
Nhóm thực hiện:

Quality

ĐỒ ÁN MÔN HỌC KIỂM THỬ PHẦN MỀM

**TÌM HIỂU VÀ PHÂN TÍCH CÔNG
CỤ KIỂM THỬ CYPRESS**



Nhóm chúng tôi

PHẠM NGUYỄN PHÚC ÂN

NGUYỄN THỊ THANH TÂM

TRẦN NGUYỄN QUỐC ANH

NGÔ THỊ THÙY TRANG

Mục lục

Phần I TỔNG QUAN ĐỀ TÀI

Phần II CƠ SỞ LÝ THUYẾT

Phần III THỰC HÀNH VỚI CYPRESS

Phần IV KẾT LUẬN

Phần V TỔNG KẾT



PHẦN I. TỔNG QUAN ĐỀ TÀI

1.1 Tình hình chung

- Lập trình web ngày càng quan trọng trong nhiều lĩnh vực.
- Doanh nghiệp đầu tư mạnh vào nền tảng web, kéo theo yêu cầu cao về chất lượng phần mềm.
- Kiểm thử phần mềm trở nên thiết yếu, đặc biệt là kiểm thử tự động với các công cụ như Cypress.
- Cypress giúp kiểm thử giao diện nhanh chóng, chính xác, dễ tích hợp CI/CD, phát hiện lỗi sớm, tiết kiệm thời gian và nâng cao trải nghiệm người dùng.
- Kiểm thử tự động là yếu tố then chốt để phát triển ứng dụng web chất lượng cao trong bối cảnh công nghệ liên tục đổi mới.

PHẦN I. TỔNG QUAN ĐỀ TÀI

1.2 Mục đích và nội dung kiểm thử

Mục đích: Phân tích lý thuyết và kỹ thuật kiểm thử phần mềm.

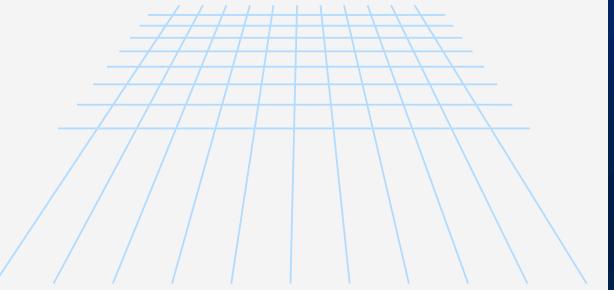
Nội dung:

- Tổng quan về API.
- Nghiên cứu tính năng nổi bật của Cypress.
- Trình bày quy trình ứng dụng Cypress để kiểm thử trang web nhà trường.

PHẦN II. CƠ SỞ LÝ THUYẾT

2.1 Kiểm thử phần mềm

- Là quá trình đánh giá hệ thống hoặc thành phần phần mềm.
- Mục tiêu: Tìm lỗi hoặc xác minh hoạt động đúng theo yêu cầu.
- Bước không thể thiếu trong phát triển phần mềm.
- Nhằm đảm bảo chất lượng, độ ổn định và độ tin cậy trước khi triển khai.



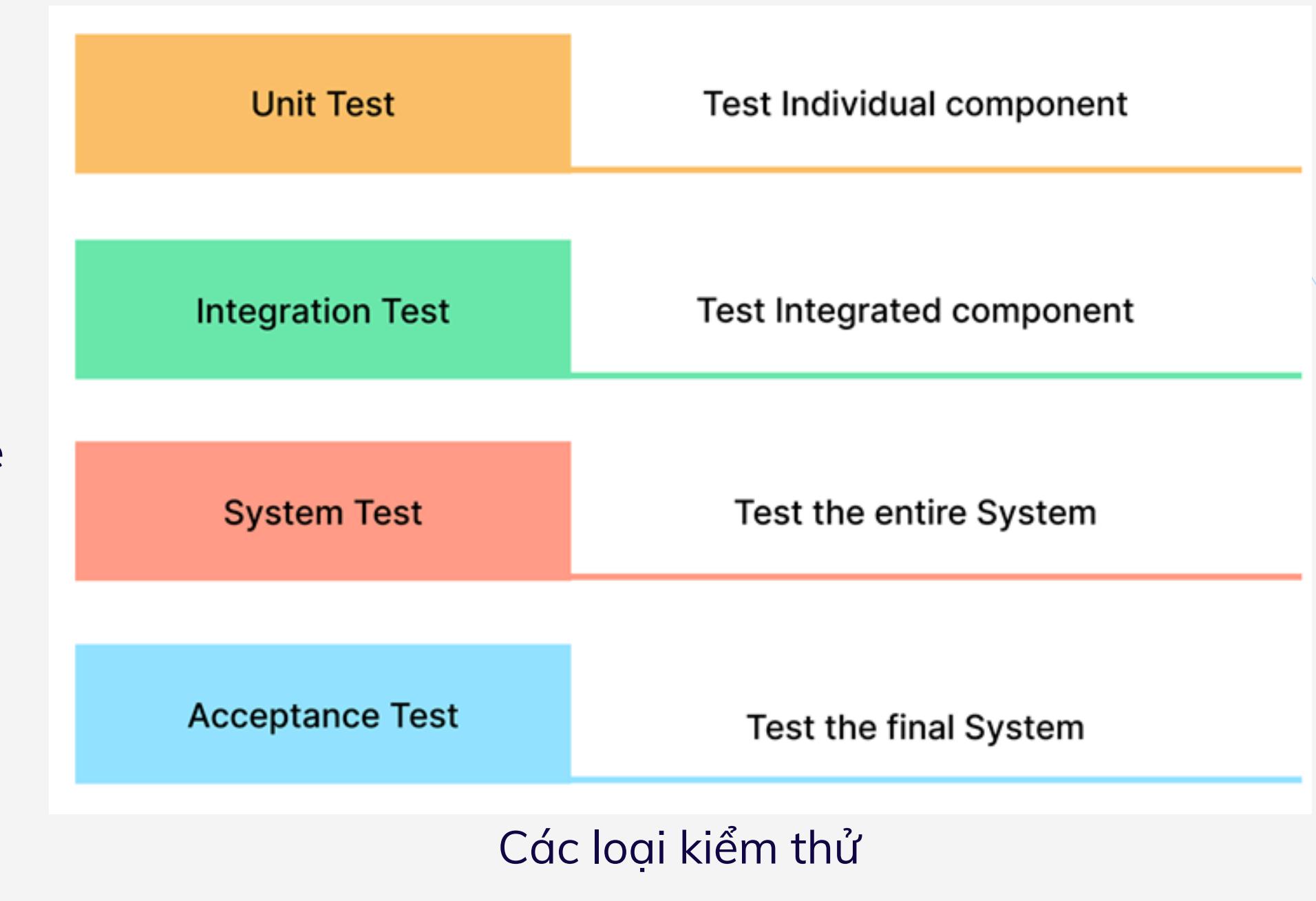
PHẦN II. CƠ SỞ LÝ THUYẾT

2.1 Kiểm thử phần mềm

2.1.1 Mục tiêu: Phát hiện lỗi, đảm bảo chất lượng, xác nhận & thẩm định, tăng cường tin cậy.

2.1.2 Các loại kiểm thử: Đơn vị, tích hợp, hệ thống, chấp nhận.

2.1.3 Phân loại theo mục tiêu: Chức năng, phi chức năng, hồi quy, thăm dò.



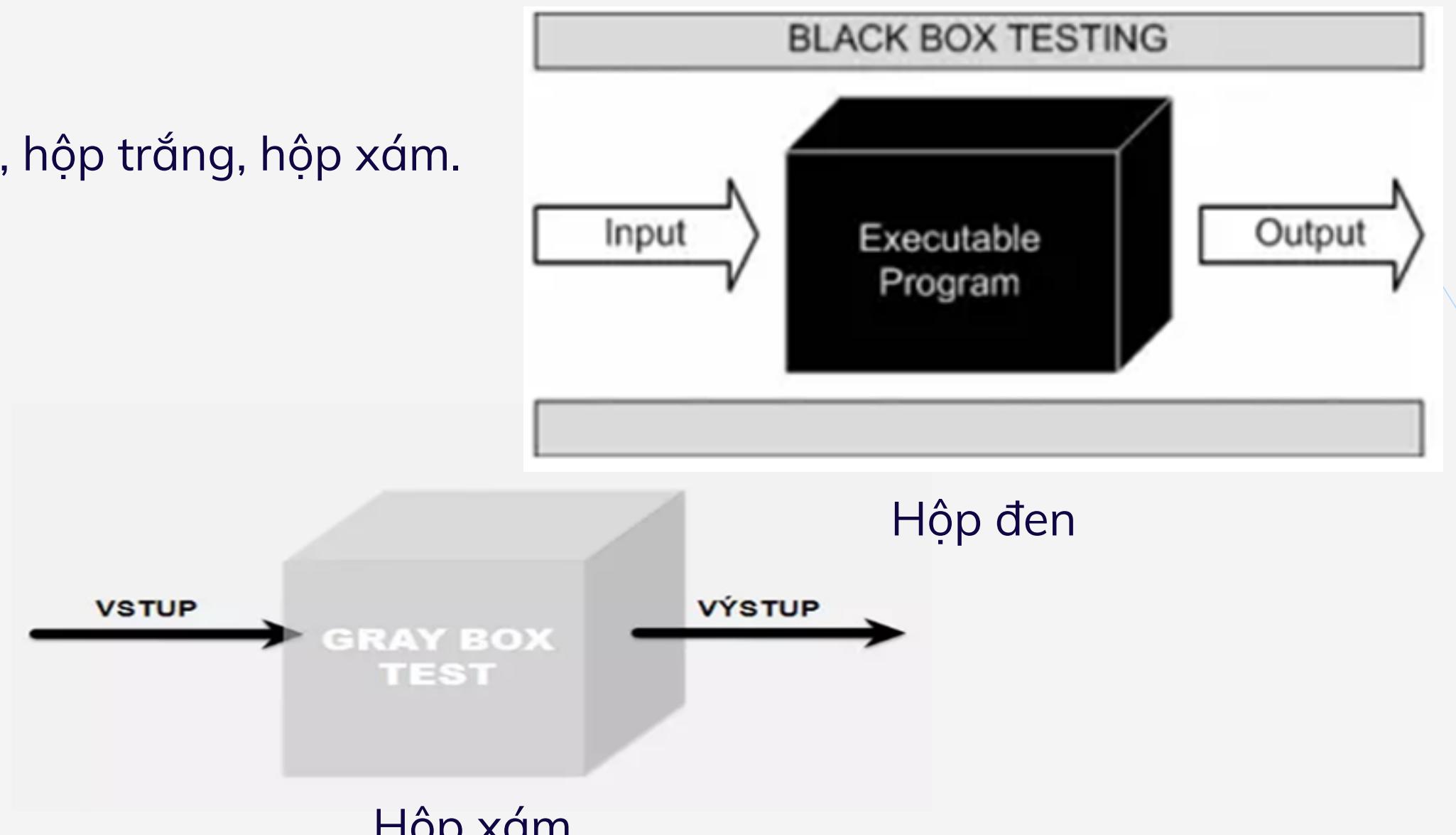
PHẦN II. CƠ SỞ LÝ THUYẾT

2.1 Kiểm thử phần mềm

2.1.4 Phân loại theo kỹ thuật: Hộp đen, hộp trắng, hộp xám.



Hộp trắng



PHẦN II. CƠ SỞ LÝ THUYẾT

2.1 Kiểm thử phần mềm

2.1.5 Quy trình: Phân tích yêu cầu → Lập kế hoạch → Thiết kế test case → Thiết lập môi trường → Thực hiện → Báo cáo lỗi → Kiểm thử lại & hồi quy.

2.1.6 Kỹ thuật thiết kế test case:
Phân vùng tương đương, phân tích giá trị biên, bảng quyết định, biểu đồ chuyển trạng thái, phỏng đoán lỗi.



Test Case Design Techniques

- Boundary Value Analysis (BVA)
- Equivalence Class Partitioning
- Decision Table
- State Transition
- Error Guessing

Kỹ thuật TK test case

PHẦN II. CƠ SỞ LÝ THUYẾT

2.2 Công cụ Cypress



2.2.1 Giới thiệu về Cypress:

- Công cụ kiểm thử tự động mã nguồn mở, hiện đại, dành riêng cho ứng dụng web.
- Giải quyết khó khăn của các framework truyền thống (như Selenium).
- Hoạt động trực tiếp trong trình duyệt, tiếp cận DOM và sự kiện trình duyệt nhanh chóng, đồng bộ.
- Chạy trên cùng vòng lặp sự kiện với ứng dụng web, kiểm thử mượt mà, dễ theo dõi.
- Phù hợp kiểm thử đơn vị, tích hợp, E2E, đặc biệt là SPA (React, Vue, Angular).

PHẦN II. CƠ SỞ LÝ THUYẾT

2.2 Công cụ Cypress

2.2.2 Giới thiệu về End-to-End (E2E):

- Kỹ thuật kiểm thử toàn bộ luồng hoạt động của ứng dụng từ đầu vào đến đầu ra cuối cùng.
- Mục tiêu: Mô phỏng hành vi người dùng thực tế để kiểm tra toàn bộ hệ thống.

2.2.3 Kiểm thử end-to-end (E2E):

- Điểm kiểm tra cuối cùng trước khi phát hành.
- Lý do cần thiết: Xác thực toàn diện, trải nghiệm người dùng, phát hiện lỗi sớm, đảm bảo hiệu suất, kiểm tra bảo mật, độ tin cậy, tuân thủ yêu cầu

PHẦN II. CƠ SỞ LÝ THUYẾT

2.2 Công cụ Cypress

2.2.4 Kiến trúc Cypress:

- Chạy trong cùng quy trình với ứng dụng web (trong trình duyệt).
- Thành phần chính: Test Runner (GUI), Cypress Node Server, Browser (Trình duyệt), Spec Files (Test files)

2.2.5 Mục đích dùng Cypress (Các loại kiểm thử hỗ trợ):

Loại kiểm thử	Mục đích chính
E2E Testing	Mô phỏng hành vi người dùng thật
Component Testing	Kiểm thử từng phần giao diện độc lập
API Testing	Đảm bảo các API hoạt động đúng
Network Testing	Giả lập phản hồi API, kiểm thử UI với nhiều tình huống
Smoke Testing	Kiểm thử nhanh sau khi deploy
Regression Testing	Đảm bảo chức năng cũ không hỏng sau cập nhật
Negative Testing	Kiểm thử hệ thống xử lý dữ liệu sai/thất bại như thế nào

PHẦN II. CƠ SỞ LÝ THUYẾT

2.2 Công cụ Cypress

2.2.6 Những tính năng chính của Cypress:

- Kiểm thử End-to-End (E2E)
- Tự động reload (Auto-reload)
- Kiểm thử component / UI riêng lẻ
- Time-travel debugging
- Kiểm soát mạng (Network Control)
- Đợi tự động (Automatic waiting)
- Giao diện Test Runner trực quan
- Chụp ảnh màn hình, quay video khi test fail/chạy
- Tích hợp CI/CD dễ dàng
- Dễ mở rộng & có hệ sinh thái plugin

2.2.7 Thực hành với Cypress:

- Công cụ kiểm thử tự động mã nguồn mở, hiện đại, mạnh mẽ cho kiểm thử web (E2E)
- Tích hợp chặt chẽ với trình duyệt, dễ viết, chạy, gỡ lỗi
- Tốc độ và phản hồi nhanh
- Tích hợp DevTools mạnh mẽ
- Kiến trúc đơn giản (dựa trên Node.js)
- Hỗ trợ nhiều loại kiểm thử (E2E, tích hợp, đơn vị)
- Tích hợp CI/CD dễ dàng
- Hệ sinh thái plugin phong phú
- Khả năng ghi log và debug mạnh
- Giao diện thân thiện (Test Runner GUI)

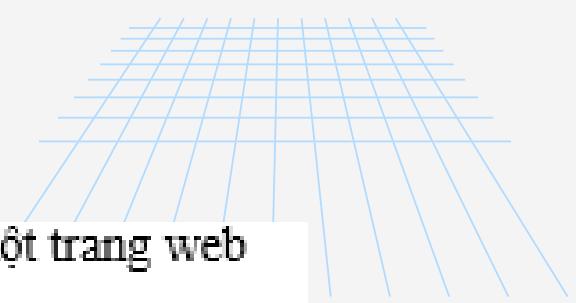
PHẦN II. CƠ SỞ LÝ THUYẾT

2.2 Công cụ Cypress

2.2.8 Điều hướng quy trình làm việc:

- Luồng hoạt động cơ bản:
 1. Cypress khởi động trình duyệt.
 2. Tải ứng dụng + mã test vào cùng frame.
 3. Các bài test chạy và truy cập trực tiếp vào DOM,...
 4. Yêu cầu mạng có thể bị chặn/sửa đổi/giả lập.
 5. Kết quả hiển thị ở Test Runner.

2.2.9 Một số câu lệnh cơ bản:



```
cy.visit('https://example.cypress.io'); Khởi động một trang web  
cy.get('selector')  
cy.get('#submit-button').click(); Click vào phần tử  
cy.get('#email').type('user@example.com'); Nhập dữ liệu vào input
```

- Kỹ thuật CSS Selector: Dùng cú pháp CSS để xác định và truy cập phần tử.
- Các CSS Selector phổ biến: Theo ID, class, thẻ HTML, thuộc tính, kết hợp, con/tổ tiên, pseudo-class.

PHẦN II. CƠ SỞ LÝ THUYẾT

2.2 Công cụ Cypress

2.2.10 So sánh các công cụ kiểm thử tự động:

Tiêu chí	Cypress	Selenium	Playwright
Ngôn ngữ	JavaScript	Java, Python, JS...	JavaScript, Python
Tốc độ	Nhanh	Chậm hơn	Nhanh
Debug	Giao diện tốt	Khó	Tốt
Giao diện đồ họa	Có	Không	Không
Cài đặt	Dễ (npm)	Phức tạp	Dễ
Hỗ trợ kiểm thử API	Có	Cần thêm tool	Có
Tích hợp CI/CD	Tốt	Tốt	Tốt
Hỗ trợ mobile	Không	Có	Có

PHẦN II. CƠ SỞ LÝ THUYẾT

2.3 API

2.3.1 API là gì?

- API (Application Programming Interface): Giao diện lập trình ứng dụng.
- Tập hợp định nghĩa, quy tắc, công cụ cho phép các phần mềm giao tiếp và tương tác.
- Đóng vai trò "cầu nối" giữa các phần mềm, gửi yêu cầu và nhận kết quả mà không cần biết logic bên trong.
- Ví dụ: Ứng dụng đặt đồ ăn gọi API của hệ thống nhà hàng để lấy món, đặt hàng, theo dõi trạng thái.
- Thành phần:
 - Endpoint: Địa chỉ cụ thể để thực hiện hành động.
 - Phương thức (GET, POST, PUT, DELETE).
 - Tham số và tiêu đề: Dữ liệu đầu vào và thông tin xác thực.

PHẦN II. CƠ SỞ LÝ THUYẾT

2.3 API

2.3.2 Các loại API phổ biến: REST API, SOAP API, GraphQL, WebSocket API, Open API / Swagger.

2.3.3 Đặc điểm của kiểm thử API: Không phụ thuộc giao diện người dùng, Tốc độ xử lý nhanh hơn, Tính chính xác cao, Tự động hóa tốt, phù hợp CI/CD, Phát hiện lỗi sớm, Thủ nghiệm nhiều tình huống phức tạp dễ dàng.

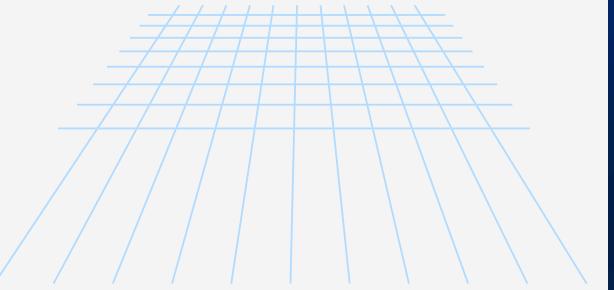
2.3.4 Các phương pháp kiểm thử API: Functional Testing, Performance Testing, Security Testing, Compatibility Testing, Negative Testing, Regression Testing,

2.3.5 Tầm quan trọng của kiểm thử API: Đảm bảo chất lượng lỗi hệ thống, Tiết kiệm chi phí sửa lỗi sớm, Tăng tốc độ phát triển (frontend/mobile không lo dữ liệu sai), Hỗ trợ CI/CD (phát hiện lỗi tự động), Cải thiện trải nghiệm người dùng gián tiếp (API nhanh, chính xác).

PHẦN III. THỰC HÀNH VỚI CYPRESS

3.1 Yêu cầu hệ thống

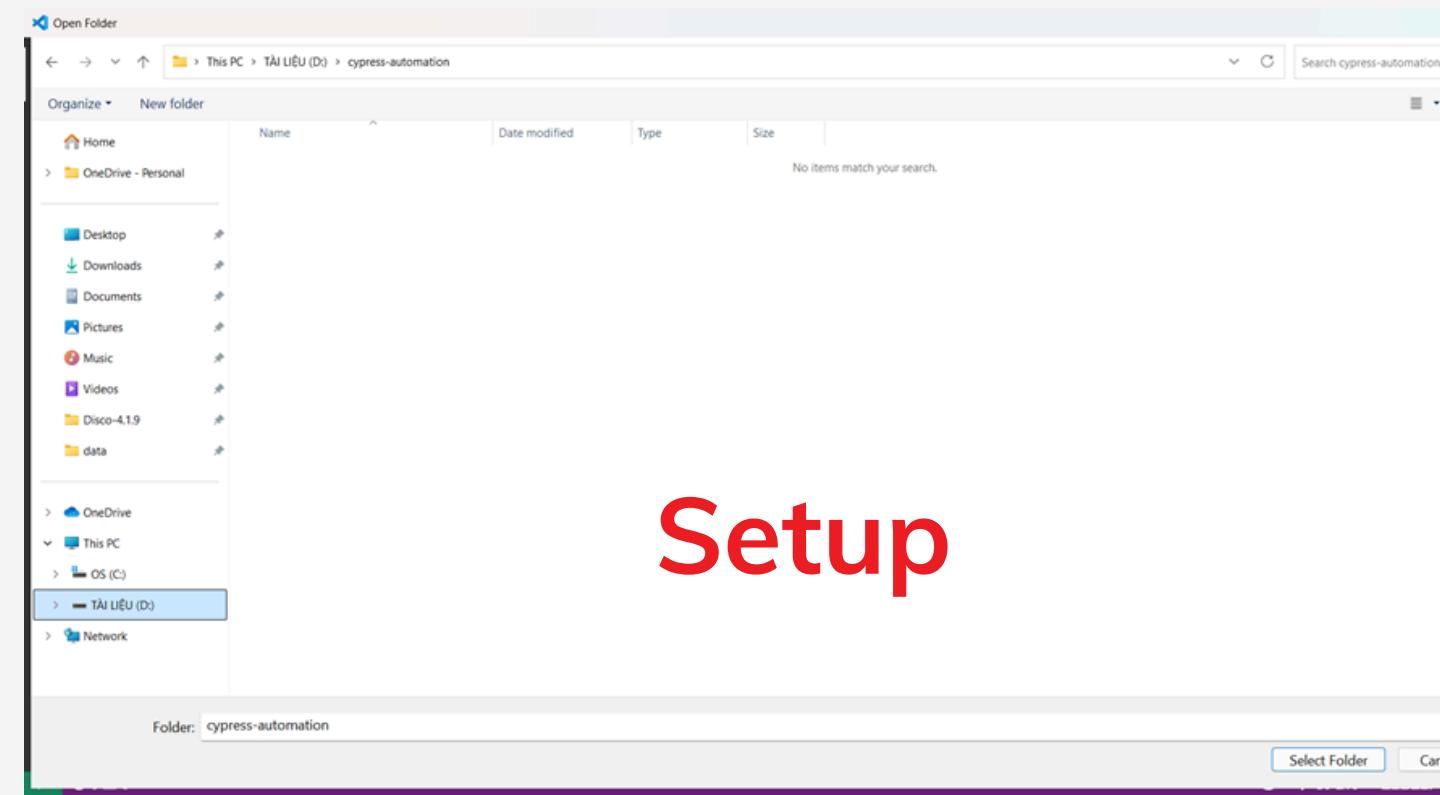
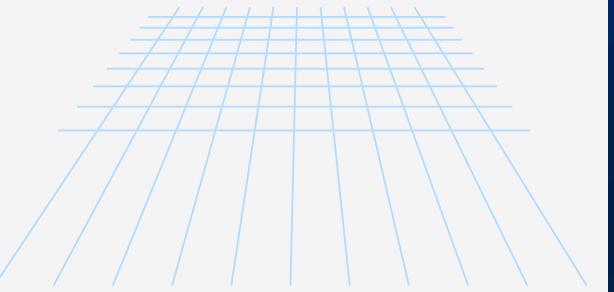
- Node.js phiên bản 12 trở lên.
- Hệ điều hành: Windows, macOS, hoặc Linux.
- Trình duyệt hỗ trợ: Chrome, Firefox, Edge, Electron.



PHẦN III. THỰC HÀNH VỚI CYPRESS

3.2 Cài đặt môi trường cho Cypress

1. Tải và cài đặt Node.js từ trang chủ.
2. Cài đặt VS Code.
3. Tạo và thiết lập thư mục dự án Cypress (khởi tạo package.json).



Setup

A screenshot of the VS Code interface. On the left, the Explorer sidebar shows a project folder 'CYPRESS-AUTOMATION' containing a 'package.json' file. On the right, the terminal window displays the command 'npm init' being run in the directory 'D:\cypress-automation'. The terminal output shows the creation of a package.json file with the following content:

```
Microsoft Windows [Version 10.0.26100.4061]
(c) Microsoft Corporation. All rights reserved.

D:\cypress-automation>npm -i init
npm warn Expanding --i to --iwr. This will stop working in the next major version of npm.
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See 'npm help init' for definitive documentation on these fields
and exactly what they do.

Use 'npm install <pkg>' afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (cypress-automation) cypress-automation
version: (1.0.0)
description:
entry point: (index.js)
test command:
git repository:
keywords:
author:
license: (ISC)
type: (commonjs)
About to write to D:\cypress-automation\package.json.

{
  "name": "cypress-automation",
  "version": "1.0.0"
}
```

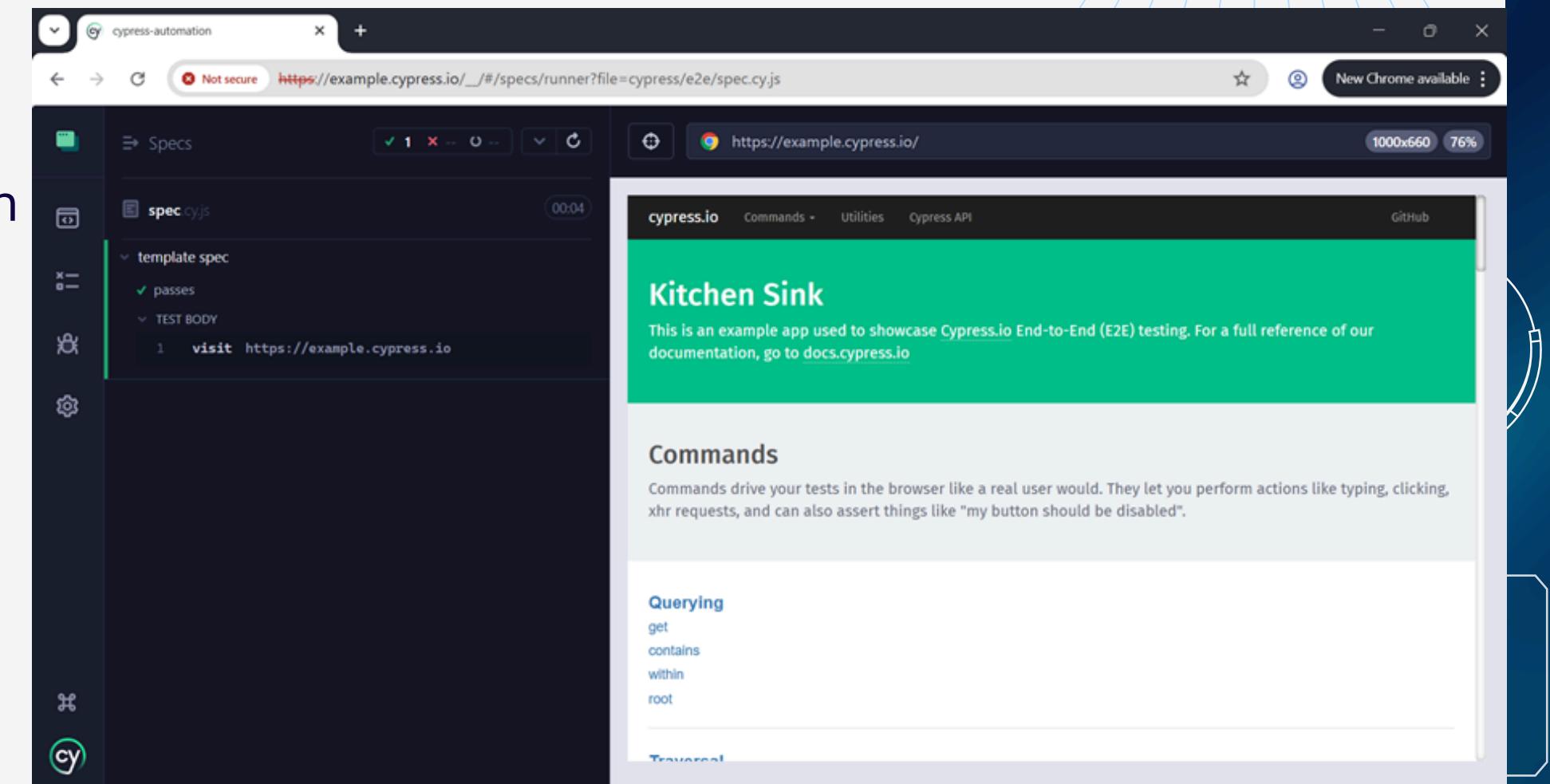
Cấu hình

PHẦN III. THỰC HÀNH VỚI CYPRESS

3.3 Cài đặt Cypress

1. Dùng lệnh `npm install cypress --save-dev` để cài đặt Cypress vào dự án.
2. Khởi động Cypress bằng lệnh `npx cypress open`.
3. Chọn loại kiểm thử (E2E Testing hoặc Component Testing).
4. Cypress sẽ tự động tạo các thư mục cấu hình cần thiết.
5. Chọn trình duyệt để chạy kiểm thử.
6. Tạo file test mới (`.cy.js`).
7. Viết test script trong file này.
8. Chạy file kiểm thử và xem kết quả trực quan trong Cypress Test Runner.

KQ bài mẫu cypress



PHẦN III. THỰC HÀNH VỚI CYPRESS

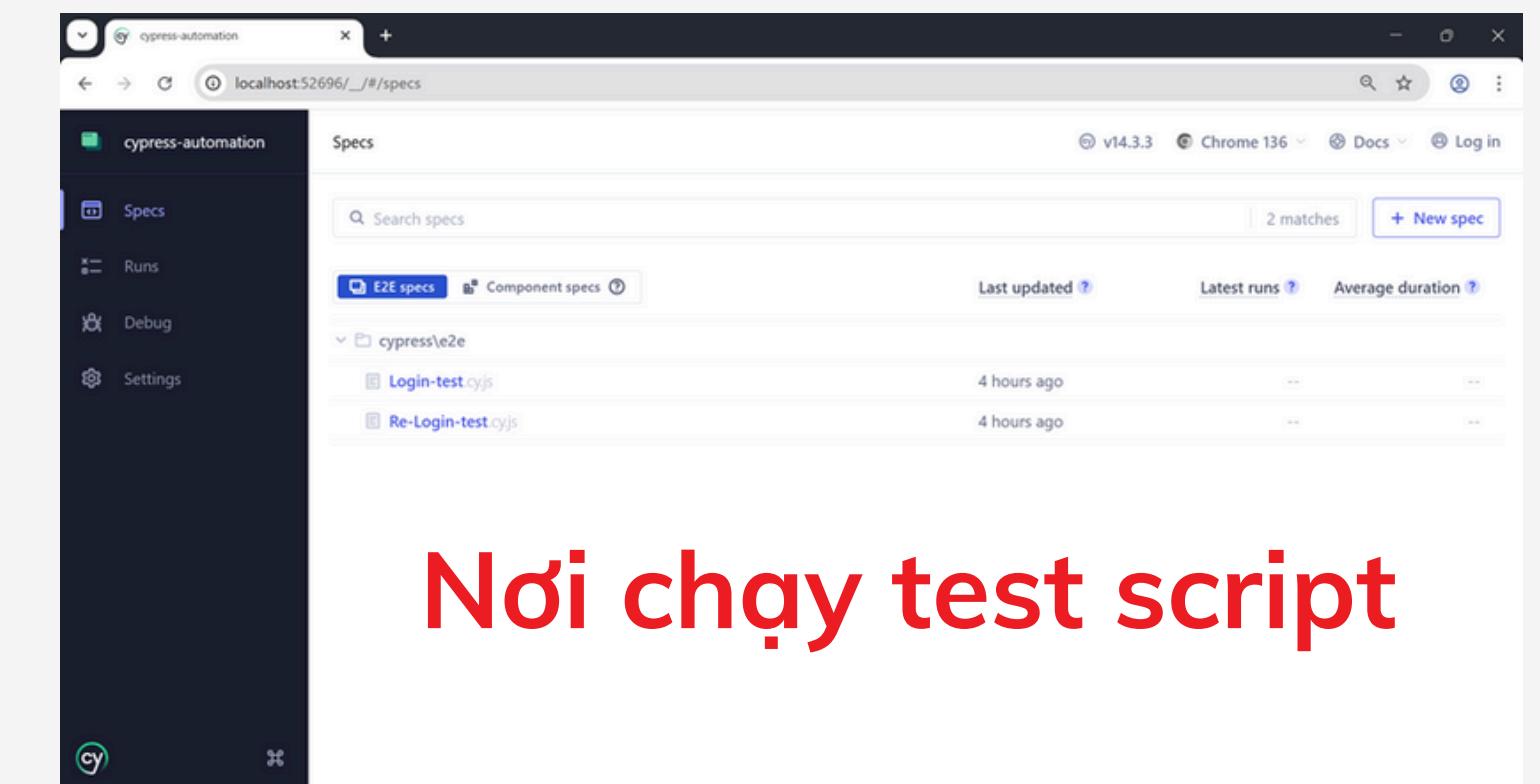
3.4 Thực hành với Cypress

1. Viết test script:

- Kiểm tra đăng nhập sai thông tin (hiển thị thông báo "Invalid login...").
- Kiểm tra đăng nhập đúng thông tin (chuyển đến trang <https://lcms.ntt.edu.vn/my/>).

2. Chạy Cypress và mở Test Runner.

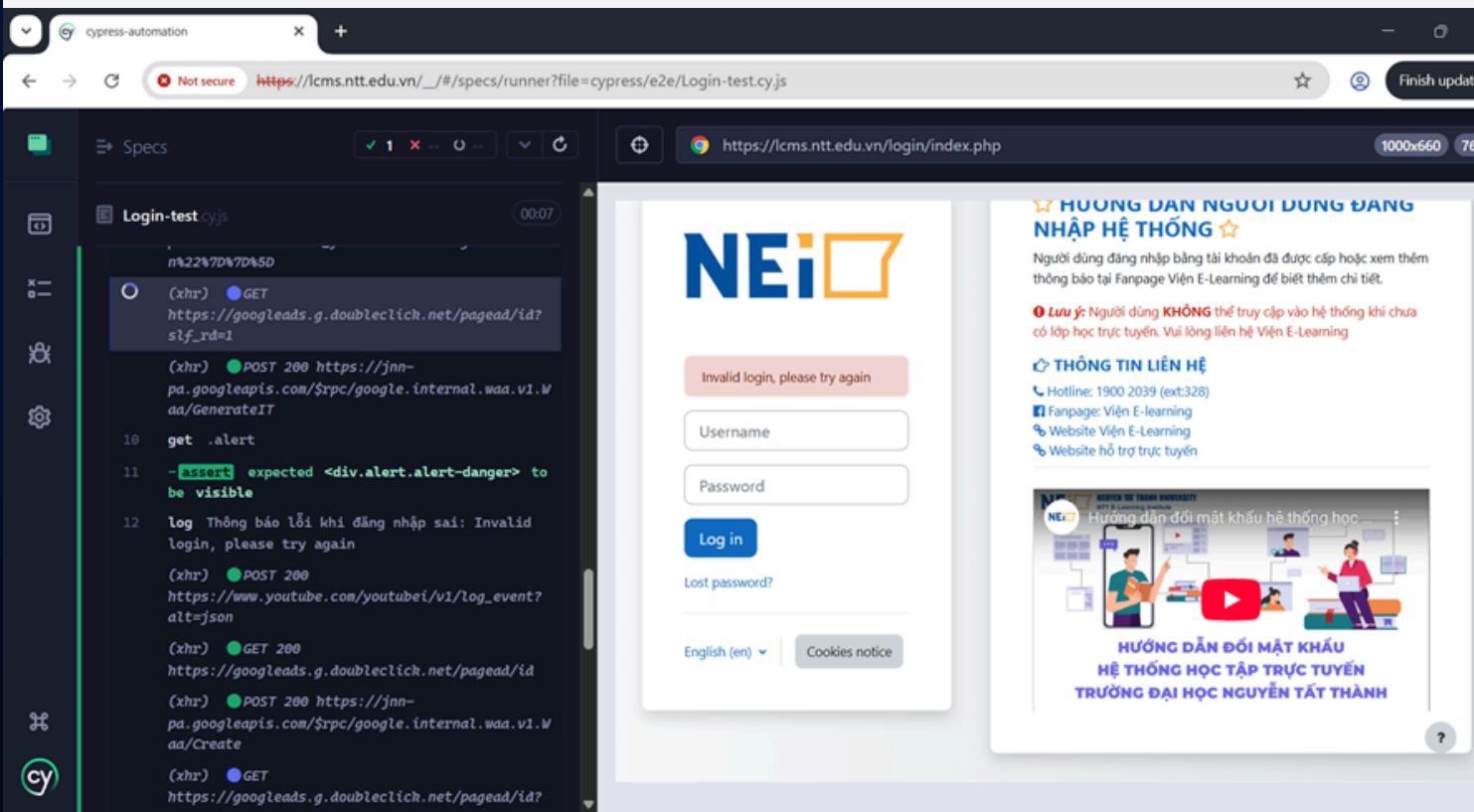
```
cypress > e2e > Login-test.cy.js
cypress > e2e > Login-test.cy.js > ...
describe("Login test", () => {
  it("hiển thị thông báo lỗi khi sai mật khẩu", () => {
    cy.visit("https://lcms.ntt.edu.vn/login/index.php");
    cy.get("#username").should("be.visible");
    cy.get("#username").type("Ten-dang-nhap");
    cy.get("#password").type("Mat-khau");
    cy.get("button[type='submit']").click();
    cy.get(".alert")
      .should("be.visible")
      .then($alert) => {
        const alertText = $alert.text();
        cy.log(`Thông báo lỗi khi đăng nhập sai: ${alertText}`);
      });
  });
})
```



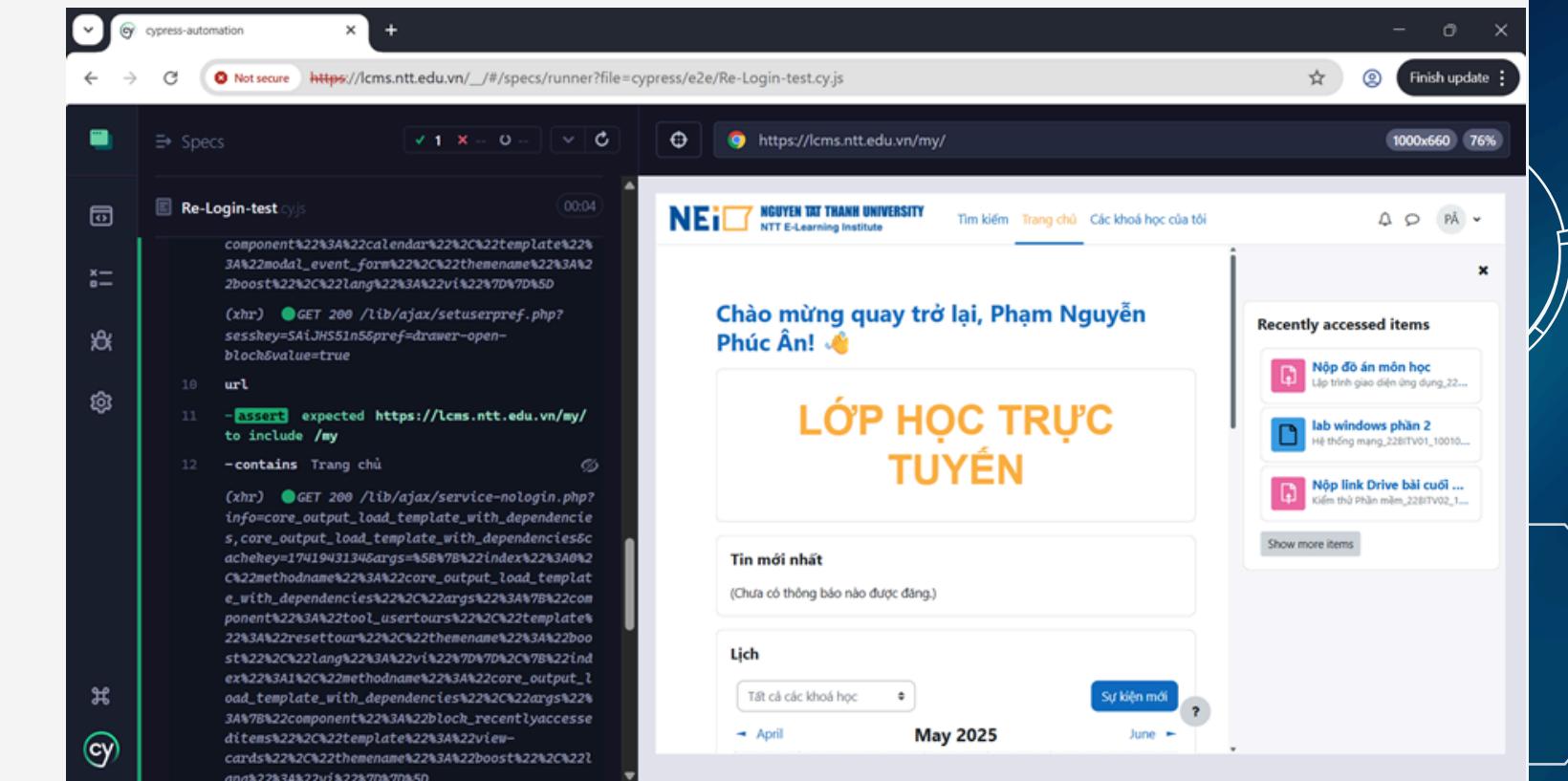
PHẦN III. THỰC HÀNH VỚI CYPRESS

3.4 Thực hành với Cypress

3. Chọn và chạy test script đăng nhập sai:
Cypress hiển thị quá trình kiểm thử và giao diện thông báo lỗi đăng nhập.



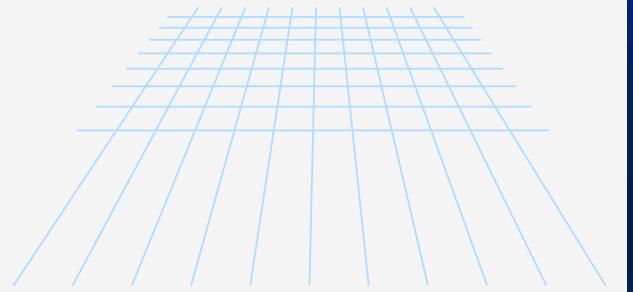
4. Chọn và chạy test script đăng nhập đúng:
Cypress hiển thị quá trình kiểm thử và giao diện sau khi đăng nhập thành công, chuyển đến trang chủ LCMS.



PHẦN IV. KẾT LUẬN

4.1 Ưu điểm

- Miễn phí, mã nguồn mở: Dễ tiếp cận cho mọi đối tượng.
- Đơn giản, dễ cài đặt và viết test script: Cài đặt nhanh, cú pháp trực quan.
- Document đầy đủ, chi tiết, có examples: Dễ học và làm quen.
- Debug dễ dàng: Ghi lại thao tác, "time-travel" để xem trạng thái DOM.
- Không cần wait thủ công: Tự động đợi phần tử xuất hiện.
- Dễ kiểm soát và kiểm tra mạng: Giả lập băng thông, lỗi server.
- Hỗ trợ chụp ảnh lỗi và quay video: Hữu ích cho debug và báo cáo.
- Cung cấp Dashboard service (tùy chọn): Xem tổng quan, báo cáo kết quả test.



PHẦN IV. KẾT LUẬN

4.2 Nhược điểm

- Cộng đồng sử dụng chưa nhiều: Ít tài nguyên tham khảo hơn so với Selenium/Playwright.
- Không hỗ trợ test native mobile app: Chỉ hoạt động trên trình duyệt web.
- Không hỗ trợ tương tác nhiều tab: Chạy trong một tab duy nhất.
- Dashboard service bản free có hạn chế: Cần trả phí để sử dụng đầy đủ tính năng.
- Chỉ hỗ trợ Javascript/TypeScript: Không phù hợp nếu dùng ngôn ngữ khác.
- Hỗ trợ ít framework test (Mocha JS): Không hỗ trợ chính thức các framework khác.
- Trình duyệt hỗ trợ còn hạn chế: Không hỗ trợ Safari và trình duyệt mobile thật sự.

PHẦN IV. KẾT LUẬN

4.3 Hướng phát triển

- Tích hợp Cypress vào quy trình CI/CD: Tự động chạy test, phát hiện lỗi sớm.
- Sử dụng Cypress Cloud (Dashboard Service) hiệu quả hơn: Nâng cấp để tận dụng đầy đủ tính năng phân tích.
- Mở rộng phạm vi test: Kiểm thử API, hiệu năng (cơ bản), mô phỏng mạng.
- Xây dựng thư viện test case tái sử dụng: Tạo custom commands, quản lý dữ liệu test tập trung.
- Kết hợp Cypress với công nghệ mới: AI, ML, công cụ báo cáo nâng cao (Allure, Mochawesome).
- Mở rộng kỹ năng và công nghệ kiểm thử: Học thêm Appium (mobile), Playwright (đa trình duyệt), Postman/Newman (API).

PHẦN V. TỔNG KẾT

- Cypress là công cụ kiểm thử tự động hiện đại, nhanh chóng, dễ sử dụng, mang lại trải nghiệm tốt trong kiểm thử giao diện và mô phỏng người dùng.
- Quá trình thực hiện giúp nhóm củng cố lý thuyết và kỹ năng thực hành kiểm thử phần mềm.
- Để tối ưu hiệu quả Cypress cho dự án lớn, cần phát triển bài bản: tích hợp CI/CD, mở rộng loại kiểm thử, xây dựng hệ thống tái sử dụng và học hỏi công nghệ bổ trợ.
- Đây là nền tảng quan trọng để đảm bảo chất lượng phần mềm cho các dự án tương lai.

We
thank
you!