# Các đặc điểm cốt lõi của một unit test tốt

* Bảo vệ chống hồi quy (Protection against regressions)
* Chống chịu với refactoring (Resistance to refactoring)
* Phản hồi nhanh (Fast feedback)
* Dễ bảo trì (Maintainability)

## Phân tích bốn trụ cột của một unit test tốt

### The first pillar: Protection against regressions

Regression là lỗi phần mềm. Đó là khi một tính năng ngừng hoạt động như mong đợi sau khi có thay đổi trong code, thường là sau khi triển khai chức năng mới.

Vì vậy, Unit test cần có cơ chế “Protection against regressions” để duy trì sự phát triển lâu dài của dự án. Để đánh giá mức độ Protection against regressions của Unit Test, cần xem xét 3 vấn đề sau:

* The amount of code that is executed during the test: Càng nhiều mã được kiểm tra, khả năng phát hiện lỗi hồi quy càng cao.
* The complexity of that code: Code có logic phức tạp quan trọng hơn so với code đơn giản. Các lỗi trong phần logic quan trọng sẽ gây tác động lớn hơn.
* The code’s domain significance: Các phần mã liên quan đến nghiệp vụ quan trọng (business-critical functionality) cần được kiểm thử kỹ hơn so với mã không có ý nghĩa nhiều đối với hoạt động của phần mềm.

Cần kiểm thử cả mã của bên thứ ba. Không chỉ kiểm thử mã do bạn viết, mà cả các thư viện, framework, hoặc hệ thống bên ngoài cũng cần được kiểm tra

**To maximize the metric of protection against regressions, the test needs to aim at exercising as much code as possible.**

### The second pillar: Resistance to refactoring

Thể hiện ở việc một unit test có thể duy trì tính đúng đắn sau khi refactoring code.

Nếu unit test fail dù chức năng vẫn hoạt động tốt, nó được gọi là “false positive”.

Tác hại của “false positive”:

* Làm giảm hiệu quả của hệ thống kiểm thử: Khi gặp quá nhiều cảnh báo giả, lập trình viên dần mất niềm tin vào test suite.
* Khiến developer e ngại refactoring: Nếu mỗi lần chỉnh sửa mã đều làm hỏng nhiều bài kiểm thử, họ sẽ hạn chế thay đổi mã nguồn, khiến mã ngày càng khó bảo trì.

### Fast feedback

Test chạy nhanh làm cho việc chạy test diễn ra nhanh và thường xuyên hơn, không làm gián đoạn công việc​

### Maintainability

Cần quan tâm đến 2 vấn đề:

* How hard it is to understand the test
* How hard it is to run the test

## In search of an ideal test

Một unit test lý tưởng phải đạt điểm tối đa trong bốn tiêu chí:

* Protection against regressions
* Resistance to refactoring
* Fast feedback
* Maintainability

Công thức đánh giá giá trị của một test được tác giả đưa ra như sau:

Value estimate=[0..1]∗[0..1]∗[0..1]∗[0..1]

Nếu một yếu tố có giá trị bằng 0, giá trị của test sẽ trở thành 0​

Ba tiêu chí đầu tiên (protection against regressions, resistance to refactoring, fast feedback) có mâu thuẫn với nhau.

Vì vậy, không thể tạo ra một unit test hoàn hảo, nhưng có thể tối ưu hóa sao cho không có tiêu chí nào bị giảm quá nhiều.

### Extreme case #1: End-to-end tests

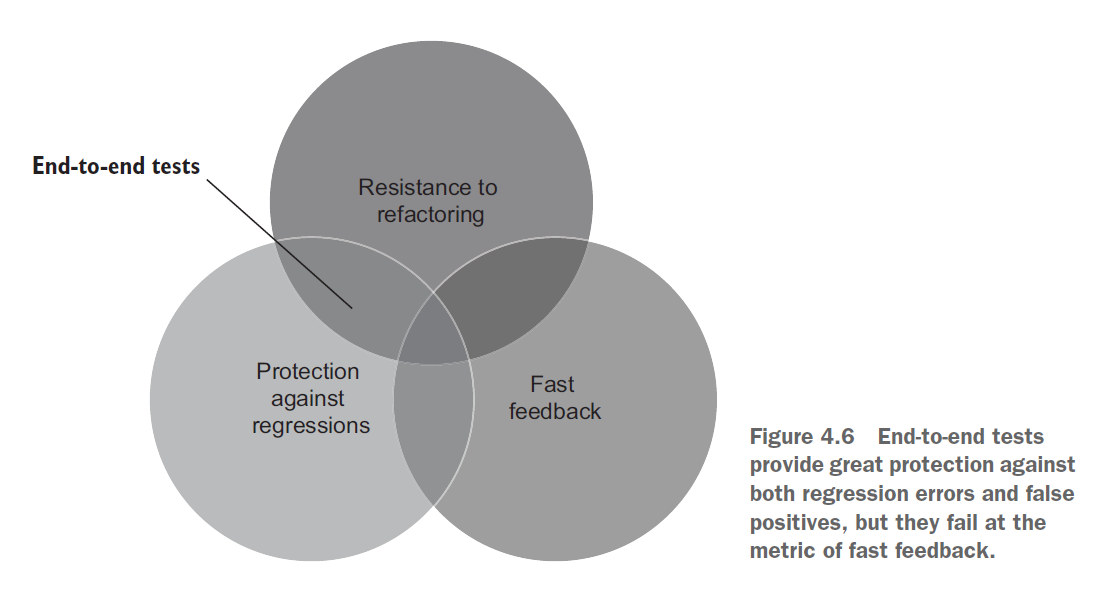
End-to-end (E2E) tests kiểm tra toàn bộ hệ thống

**Ưu điểm:**,

* Protection against regressions mạnh, vì chúng kiểm tra toàn bộ ứng dụng.
* Có **khả năng chống chịu với refactoring cao**, vì chúng chỉ quan tâm đến kết quả cuối cùng.

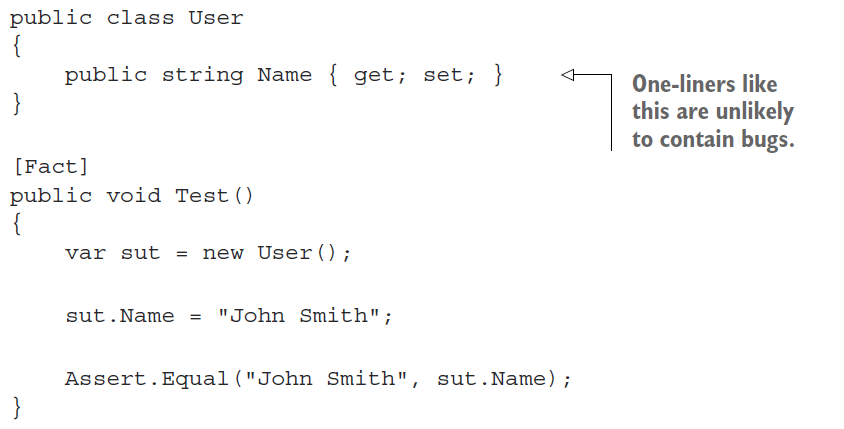
**Nhược điểm:**

* **Chạy chậm**, khiến feedback loop bị trì hoãn.
* **Khó bảo trì**, vì phụ thuộc vào nhiều yếu tố bên ngoài như cơ sở dữ liệu, API​.



### Extreme case #2: Trivial tests

Trivial tests là các test quá đơn giản, kiểm tra những phần code mà gần như chắc chắn không thể lỗi

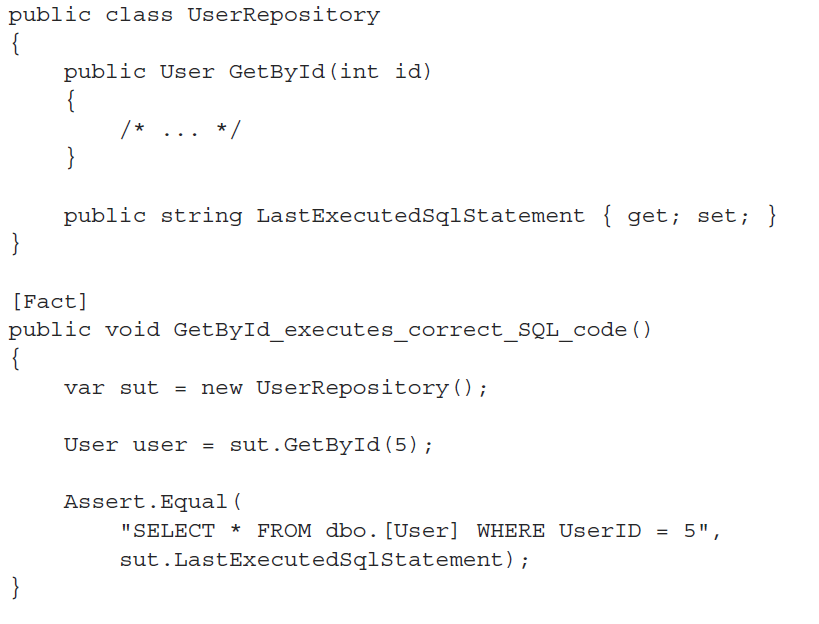


A diagram of a diagram of a test

AI-generated content may be incorrect.​

### Extreme case #3: Brittle tests

Brittle tests là các test dễ bị hỏng khi code thay đổi



Khi câu truy vấn thay đổi, test sẽ fail

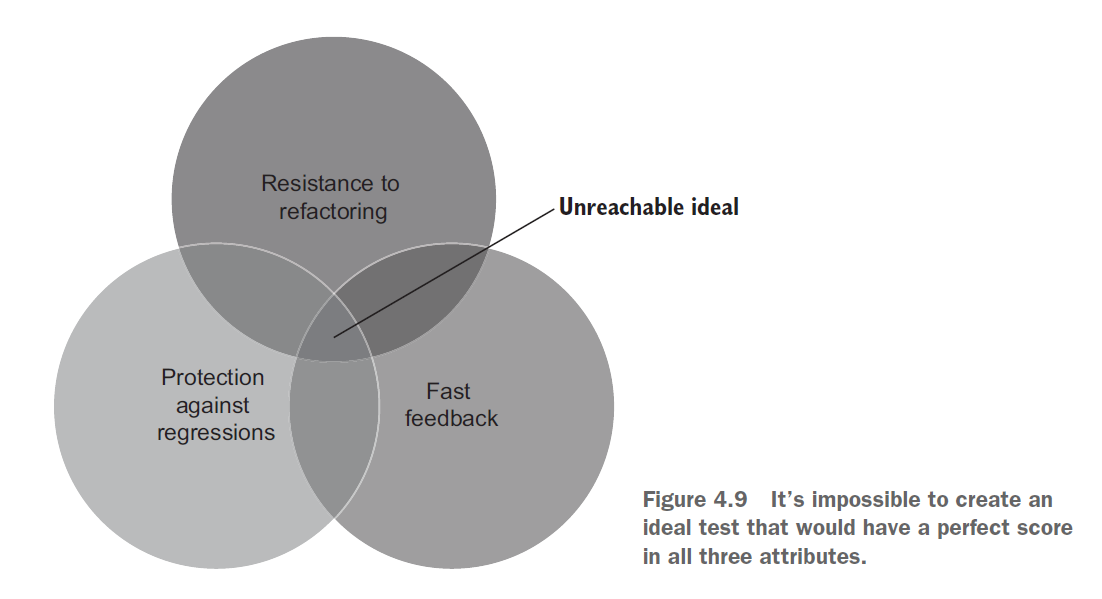
A close-up of a white background

AI-generated content may be incorrect.

A diagram of a test

AI-generated content may be incorrect.

### In search of an ideal test: The results



A diagram of a diagram

AI-generated content may be incorrect.