<p>{{sub.username}} =>{{sub.password}} =>{{sub.fullname}}=>{{sub.email}}=>{{sub.gender}}=>{{sub.birthday}}=>{{sub.schoolfee}}=>{{sub.marks}}</p>

## BACK BOX TEST

**1.1.1. Định nghĩa** Kiểm tra hộp đen (Black box testing) là một phương pháp kiểm thử phần mềm mà việc kiểm tra các chức năng của một ứng dụng không cần quan tâm vào cấu trúc nội bộ hoặc hoạt động của nó.

**1.1.2. Đối tượng được kiểm thử** Là thành phần phần mền (TPPM) có thể là 1 hàm chức năng, 1 modul chức năng, 1 phân hệ chức năng...

**1.1.3. Phương pháp thử nghiệm:** Dựa vào chức năng Kiểm thử hộp đen (Black box test) có thể được áp dụng hầu như đến mọi cấp độ của kiểm thử phần mềm:

* Kiểm thử đơn vị (Unit test)
* Kiểm thử tích hợp (Intergration test)
* Kiểm thử hệ thống (System test)
* Kiểm thử chấp nhận (Acceptance test).

Tuy nhiên, Black box test được sử dụng thích hợp nhất trong kiểm thử hệ thống (System test) và Kiểm thử chấp nhận (Acceptance test)

**1.1.4. Đặc điểm**

* Là chiến lược kiểm thử TPPM dựa vào thông tin duy nhất là các đặc tả về yêu cầu chức năng của TPPM tương ứng.
* Người kiểm thử không cần thiết phải có kiến thức về việc mã hoá, cấu trúc bên trong của TPPM, cũng như không yêu cầu phải biết lâp trình phần mềm.
* Việc kiểm thử được tiến hành dựa vào việc kiểm thử TPPM làm được gì, có phù hợp với yêu cầu của người dùng hay không. Các tester nhập số liệu vào phần mềm và chỉ cần xem kết quả của phần mềm và các mục tiêu kiểm tra.
* Mức test này thường yêu cầu các tester phải viết test case đầy đủ trước khi test; khi test, đơn giản chỉ cần thực hiện theo các bước mô tả trong test case thao tác và nhập data vào, sau đó xem kết quả trả về hoặc hành vi của phần mềm, rồi so sánh với kết quả mong đọi được viết trong testcase

## WHITE BOX TEST

**1.2.1. Định nghĩa** Kiểm thử hộp trắng (While box test) là phương pháp thử nghiệm phần mềm, trong đó các thiết kế, cấu trúc giải thuật bên trong, và việc thực hiện các công việc đều được biết đến

**1.2.2. Đối tượng kiểm thử** Là 1 thành phần của phần mềm (1 chức năng, 1 module chức năng, 1 phân hệ chức năng....)

**1.2.3. Phương pháp thử nghiệm:** Dựa vào thuật giải Kiểm thử hộp trắng dựa vào thuật giải cụ thể, vào cấu trúc dữ liệu bên trong của ₫ơn vị phần mềm cần kiểm thử ₫ể xác ₫ịnh ₫ơn vị phần mềm ₫ó có thực hiện ₫úng không.

* Với những TPPM quá lớn sẽ tốn rất nhiều thời gian và công sức để kiểm thử nếu như dùng kiểm thử tích hợp (Integration test) hay kiểm thử chức năng (Functional test)).
* Kỹ thuật white box test thích hợp dùng để kiểm thử đơn vị (Unit test)

**1.2.4. Đặc điểm**

* Là chiến lược kiểm thử TPPM dựa vào giải thuật, cấu trúc bên trong chức năng của TPPM tương ứng.
* Người kiểm thử phải có kiến thức nhất định về việc mã hoá, cấu trúc bên trong của chức năng, biết lâp trình phần mềm.
* Việc kiểm thử được tiến hành dựa vào việc kiểm xem giải thuật, mã lệnh đã làm có đúng không.
* Mức test này thường yêu cầu các tester phải viết test case đầy đủ các nhánh trong code; khi test, sẽ set điều kiện và data để chạy vào đủ tất cả các nhánh trong giải thuật, đảm bảo thực hiện đầy đủ.

Các cấp độ kiểm thử

* **Kiểm thử** đơn vị (Unit Testing)
* **Kiểm thử** tích hợp (Integration Testing)
* **Kiểm thử** hệ thống (System Testing)
* **Kiểm thử** chấp nhận (Acceptance Testing)
* **Các loại kiểm thử**
* **Kiểm thử** cài đặt (Installation testing) ...
* **Kiểm thử** “khói” (Smoke test) ...
* **Kiểm thử** tương thích (Compatability test) ...
* **Kiểm thử** hồi quy (Regression test) ...
* **Kiểm thử** nghiệm thu (Acceptance test) ...
* **Kiểm thử** hiệu năng.
* **Các thành phần có trong test cas**
* **Test case ID**: chỉ đơn giản là ID để mình dễ gọi tên nó thôi. Thu thường hay đặt theo kiểu.
* *Ví dụ như test case của chức năng “Sign Up” thì sẽ là SU\_001.*
* – **Test case summary**: đây là phần tóm tắt, có thể gọi là tên của test case. Phần này cũng khá quan trọng, giúp cho người đọc nhận diện được mục đích của test case là gì. Một cái tên tốt là cái tên mà chỉ cần đọc vào là biết được mình phải làm gì.
* *Ví dụ : “Verify that user can sign up with valid data successfully”*
* – **Test case description**: phần này để mô tả chi tiết test case dùng để làm gì. Vì phần tên thường không thể mô tả đầy đủ được test case, nên chúng ta sẽ mô tả rõ ở đây. Nhưng theo kinh nghiệm của mình thì phần này thường hao hao giống phần “Test case summary”, nên nếu thời gian hạn hẹp, bạn có thể lượt bỏ phần này.
* –**Pre- conditions**: điều kiện, đây là phần Thu nghĩ không thể thiếu. Không thể cứ mở phần mềm ra là test được liền, bạn cần test data, test data phải thoải một số điều kiện nào đó. Bên cạnh đó là môi trường test, bạn cần phải ở màn hình nào để chuẩn bị để test…. Nói chung là không thế thiếu nhé.
* *Ví như bạn kiểm tra case sau: “Verify sign in function when there’s no internet connection”*
* *=> Thế thì Proconditon là : “1. There’s no internet connection on device. 2. User is in sign in screen”*
* – **Test data**: Có những test case sẽ cần test data, có những cái không cần. Với những hệ thống phức tạp thì thường sẽ có, bạn phải dùng một account nhất định thì bạn mới có function đó để test hoặc với một số case nhất định như valid data chẳng hạn, mình nên đưa test data để dễ dàng test hơn.
* – **Reproduce Steps**: mô tả lại từng step để thực hiện test case. Phần này là phần cực kì quan trọng và chiếm nhiều thời gian khi viết test case.
* Một test case tốt khi phần này được viết chi tiết và rõ ràng. Nhớ rằng “Đừng assume” bất cứ điều gì khi viết nhé, hãy nghĩ rằng bạn là một người không biết gì về phần mềm và đây là lần đầu tiên bạn thao tác nhé. Có thể thì test case của bạn mới rõ được.
* – **Expected result**: tương ứng với một “reproduce steps” bạn viết thì sẽ có một expected result. Không nhất thiết là step nào cũng phải có, nhưng nên là có bạn nhé.
* *Ví dụ. Trong test case “Verify that user can sign up successfully”. Bạn có steps “Press “Sign In” button” thì tương ứng phải phải có expected result “Sign in successfully. User go to home screen”*
* – **Observed/Actual result**: kết quả thực tế. Thường thì phần này bạn sẽ thêm vào khi bạn run test case nhé. Nó giúp bạn biết được thực tế phần mềm đang có gì, bị lỗi gì hay nó làm “work like expected”
* – **Test case status:** hiện trạng của phần mềm, nó là kết quả sau khi bạn “run” một test case. Thường có 4 status: Pass/Fail/Blocked/Skipped
* – **Bug ID**: nếu test case bị failed thì bạn sẽ tạo một ticket bug và thêm bug ID vào cột này nhen. (Thường team sẽ có một phần mềm để quản lý task và bug)
* –**Environment**: môi trường test. Quá trình phát triển phần mềm thường sẽ có nhiều môi trường như Dev, Staging, UAT, Production. Khi bạn run test case, thì nhớ ghi rõ là bạn test trên môi trường nào nhé.
* – **Notes/Comment**: lưu ý hay phản hồi gì đó bạn muốn ghi lại cho người khác dễ dàng thực hiện hơn khi đọc

Co bao nhieu lai kiem thu

**. Kỹ thuật phân vùng tương đương (Equivalence Partitioning)**

**2. Kỹ thuật phân tích giá trị biên (Boundary-value Analysis)**

**Đoán lỗi (Error Guessing)**