### **Phần 1: Lý thuyết**

#### **1. Error Guessing**

**a. Error guessing là gì? Giải thích cách sử dụng kinh nghiệm thực tế để dự đoán các lỗi tiềm ẩn trong hệ thống.**

Error guessing là kỹ thuật kiểm thử dựa trên kinh nghiệm của tester để dự đoán các khu vực hoặc chức năng dễ xảy ra lỗi trong hệ thống. Tester sử dụng kiến thức về các lỗi thường gặp trong các dự án tương tự, hành vi người dùng, hoặc các trường hợp đặc biệt để xác định điểm yếu. Cách tiếp cận này không dựa trên tài liệu hay kịch bản cố định mà dựa vào trực giác và kinh nghiệm thực tế, ví dụ: dự đoán lỗi ở các trường nhập liệu, tính toán, hoặc tích hợp.

**b. Ví dụ áp dụng error guessing trong kiểm thử tính năng thanh toán của ứng dụng thương mại điện tử.**

Trong một ứng dụng thương mại điện tử, tester có thể dự đoán lỗi ở tính năng thanh toán như sau:

* **Lỗi tiềm ẩn**: Người dùng nhập sai định dạng số thẻ tín dụng nhưng hệ thống vẫn cho phép thanh toán.
* **Cách áp dụng**: Tester nhập số thẻ tín dụng không hợp lệ (ví dụ: thiếu số, ký tự đặc biệt) dựa trên kinh nghiệm rằng các hệ thống thường không kiểm tra chặt chẽ định dạng đầu vào.

#### **2. Exploratory Testing**

**a. Exploratory testing là gì? Giải thích tại sao phương pháp này hiệu quả.**

Exploratory testing là phương pháp kiểm thử mà tester khám phá hệ thống một cách tự do, không dựa vào kịch bản cố định. Tester thiết kế và thực hiện kiểm thử đồng thời, dựa trên hiểu biết về hệ thống, kinh nghiệm, và phản hồi tức thời. Phương pháp này hiệu quả vì:

* Khám phá lỗi không dự đoán được mà kịch bản cố định có thể bỏ qua.
* Linh hoạt, thích nghi nhanh với thay đổi của hệ thống.
* Tăng tính sáng tạo và khả năng phát hiện lỗi ở các khu vực phức tạp.

**b. Ví dụ tình huống sử dụng exploratory testing.**

Trong kiểm thử giao diện người dùng của một ứng dụng di động (ví dụ: ứng dụng nhắn tin):

* Tester khám phá giao diện bằng cách thử thay Policing tích cực các tính năng như gửi tin nhắn, chuyển đổi giao diện sáng/tối, hoặc nhập văn bản dài bất thường.
* **Mục tiêu**: Phát hiện lỗi giao diện hoặc logic, ví dụ: nút gửi không hoạt động khi nhập ký tự đặc biệt hoặc giao diện bị vỡ khi phóng to.

#### **3. User Story và Tiêu chí chấp nhận**

**a. User story là gì? Tại sao hỗ trợ kiểm thử và hợp tác đội nhóm?**

User story là mô tả ngắn gọn về một tính năng từ góc nhìn người dùng, thường theo định dạng: “Là một [vai trò], tôi muốn [tính năng] để [mục tiêu].” Tiêu chí chấp nhận là các điều kiện cụ thể để xác định tính năng hoàn thành. User story và tiêu chí chấp nhận giúp:

* Làm rõ yêu cầu, đảm bảo tester, developer và product owner hiểu chung mục tiêu.
* Hỗ trợ kiểm thử bằng cách cung cấp tiêu chí cụ thể để thiết kế test case.
* Thúc đẩy hợp tác nhờ tạo nền tảng giao tiếp rõ ràng giữa các bên.

**b. Ví dụ user story với tiêu chí chấp nhận.**

**User story**: Là một người dùng mới, tôi muốn đăng ký tài khoản để truy cập các tính năng của ứng dụng.

**Tiêu chí chấp nhận**:

1. Người dùng có thể nhập email, mật khẩu và xác nhận mật khẩu để đăng ký.
2. Hệ thống gửi email xác nhận với liên kết kích hoạt tài khoản.
3. Hệ thống báo lỗi nếu email đã tồn tại hoặc mật khẩu không đáp ứng yêu cầu (tối thiểu 8 ký tự, có chữ hoa, chữ thường, số).

#### **4. Acceptance Test-Driven Development (ATDD)**

**a. ATDD là gì? Cách cải thiện hiệu quả làm việc nhóm.**

ATDD là phương pháp phát triển phần mềm mà các tiêu chí chấp nhận được định nghĩa trước khi phát triển, dưới dạng các bài kiểm thử tự động. Tester, developer và product owner hợp tác để viết các tiêu chí này. ATDD cải thiện hiệu quả nhóm vì:

* Đảm bảo tất cả thành viên hiểu rõ yêu cầu từ đầu.
* Giảm lỗi do hiểu lầm yêu cầu.
* Tăng tốc phát triển nhờ kiểm thử tự động hóa.

**b. Cách áp dụng ATDD và lợi ích.**

Trong dự án phát triển tính năng tìm kiếm sản phẩm:

* **Quy trình**:
  1. Product owner, developer và tester họp để định nghĩa tiêu chí chấp nhận (ví dụ: tìm kiếm trả về kết quả trong 2 giây, hỗ trợ tìm kiếm theo từ khóa và danh mục).
  2. Tester viết các kịch bản kiểm thử tự động dựa trên tiêu chí.
  3. Developer phát triển tính năng dựa trên các bài kiểm thử này.
  4. Kiểm thử tự động chạy để xác nhận tính năng hoàn thành.
* **Lợi ích**: Giảm thiểu lỗi, tăng tính minh bạch, và đảm bảo sản phẩm đáp ứng yêu cầu người dùng.

### **Phần 2: Bài tập thực hành**

#### **1. Áp dụng Error Guessing**

**Ứng dụng đặt vé xe buýt trực tuyến – Tính năng chọn ghế ngồi**

**a. Dự đoán 3 lỗi tiềm ẩn và lý do**

1. **Lỗi: Ghế đã được đặt nhưng vẫn hiển thị khả dụng.**
   * **Lý do**: Hệ thống không đồng bộ trạng thái ghế trong thời gian thực, đặc biệt khi nhiều người dùng đặt cùng lúc.
2. **Lỗi: Chọn nhiều ghế vượt quá số lượng vé.**
   * **Lý do**: Hệ thống không kiểm tra số ghế được chọn so với số vé đã mua.
3. **Lỗi: Giao diện hiển thị sai vị trí ghế.**
   * **Lý do**: Lỗi ánh xạ dữ liệu ghế trên giao diện, đặc biệt với các xe có bố trí ghế không chuẩn.

**b. Thiết kế 3 test case**

1. **Test case: Kiểm tra trạng thái ghế đã đặt**
   * **Mô tả**: Kiểm tra xem ghế đã được đặt có hiển thị trạng thái không khả dụng không.
   * **Dữ liệu đầu vào/hành động**: Hai người dùng đồng thời chọn cùng một ghế, người dùng thứ hai thử đặt ghế đó.
   * **Kết quả mong đợi**: Hệ thống báo ghế đã được đặt, không cho phép chọn.
2. **Test case: Kiểm tra số lượng ghế được chọn**
   * **Mô tả**: Kiểm tra giới hạn số ghế chọn so với số vé mua.
   * **Dữ liệu đầu vào/hành động**: Mua 2 vé nhưng chọn 3 ghế.
   * **Kết quả mong đợi**: Hệ thống báo lỗi, yêu cầu chọn đúng số lượng ghế.
3. **Test case: Kiểm tra hiển thị vị trí ghế**
   * **Mô tả**: Kiểm tra giao diện hiển thị đúng vị trí ghế.
   * **Dữ liệu đầu vào/hành động**: Chọn ghế số 5A trên sơ đồ ghế, kiểm tra thông tin hiển thị.
   * **Kết quả mong đợi**: Thông tin ghế 5A được hiển thị chính xác (vị trí, số ghế).

#### **2. Thực hiện Exploratory Testing**

**Ứng dụng đặt vé xe buýt – Tính năng chọn ghế ngồi**

**a. Kế hoạch exploratory testing**

* **Mục tiêu kiểm thử**: Phát hiện lỗi giao diện và logic trong tính năng chọn ghế ngồi.
* **Khu vực tập trung**:
  + Giao diện sơ đồ ghế (hiển thị, tương tác).
  + Phản hồi khi chọn ghế đã đặt hoặc ghế không hợp lệ.
  + Tích hợp với số lượng vé và thông tin thanh toán.
* **Thời gian dự kiến**: 30 phút.

**b. Dự đoán lỗi và cách ghi nhận**

1. **Lỗi: Sơ đồ ghế không tải lại sau khi chọn ghế.**
   * **Cách phát hiện**: Thử chọn ghế, hủy chọn, và kiểm tra xem sơ đồ cập nhật đúng không.
   * **Cách ghi nhận**: Ghi chú lỗi (ví dụ: “Sơ đồ ghế không cập nhật trạng thái sau khi hủy chọn”), kèm ảnh chụp màn hình và bước tái hiện.
2. **Lỗi: Hệ thống cho phép chọn ghế đã đặt bởi người dùng khác.**
   * **Cách phát hiện**: Mở hai phiên người dùng, thử chọn cùng một ghế.
   * **Cách ghi nhận**: Ghi chú lỗi (ví dụ: “Hệ thống không khóa ghế khi người dùng khác đang chọn”), kèm video hoặc log hệ thống.

#### **3. Viết User Story và Áp dụng ATDD**

**Ứng dụng quản lý công việc – Tính năng thêm công việc mới**

**a. User story và tiêu chí chấp nhận**

**User story**: Là một người dùng, tôi muốn thêm một công việc mới vào danh sách công việc để quản lý công việc hàng ngày.

**Tiêu chí chấp nhận**:

1. Người dùng có thể nhập tiêu đề, mô tả, ngày hết hạn và mức độ ưu tiên.
2. Hệ thống lưu công việc và hiển thị trong danh sách công việc.
3. Hệ thống báo lỗi nếu tiêu đề để trống hoặc ngày hết hạn không hợp lệ.
4. Công việc mới được gắn nhãn mặc định là “Chưa hoàn thành”.

**b. Thiết kế 3 test case dựa trên tiêu chí chấp nhận**

1. **Test case: Thêm công việc hợp lệ**
   * **Mô tả**: Kiểm tra thêm công việc với đầy đủ thông tin.
   * **Dữ liệu đầu vào**: Tiêu đề: “Hoàn thành báo cáo”, Mô tả: “Báo cáo Q3”, Ngày hết hạn: 20/07/2025, Ưu tiên: Cao.
   * **Kết quả mong đợi**: Công việc được thêm vào danh sách, hiển thị đúng thông tin, trạng thái “Chưa hoàn thành”.
2. **Test case: Thêm công việc thiếu tiêu đề**
   * **Mô tả**: Kiểm tra lỗi khi không nhập tiêu đề.
   * **Dữ liệu đầu vào**: Mô tả: “Báo cáo Q3”, Ngày hết hạn: 20/07/2025, Ưu tiên: Cao, Tiêu đề: để trống.
   * **Kết quả mong đợi**: Hệ thống báo lỗi “Tiêu đề không được để trống”.
3. **Test case: Thêm công việc với ngày hết hạn không hợp lệ**
   * **Mô tả**: Kiểm tra lỗi khi nhập ngày hết hạn trong quá khứ.
   * **Dữ liệu đầu vào**: Tiêu đề: “Hoàn thành báo cáo”, Mô tả: “Báo cáo Q3”, Ngày hết hạn: 10/07/2025, Ưu tiên: Cao.
   * **Kết quả mong đợi**: Hệ thống báo lỗi “Ngày hết hạn phải là ngày tương lai”.

**c. Áp dụng ATDD và vai trò các thành viên**

* **Quy trình ATDD**:
  1. **Product owner**: Định nghĩa user story và tiêu chí chấp nhận, đảm bảo tính năng đáp ứng nhu cầu người dùng.
  2. **Tester**: Viết các bài kiểm thử tự động dựa trên tiêu chí chấp nhận (ví dụ: kiểm tra thêm công việc hợp lệ, lỗi tiêu đề trống).
  3. **Developer**: Phát triển tính năng dựa trên các bài kiểm thử, chạy kiểm thử để xác nhận.
  4. **Hợp tác**: Cả nhóm họp để thống nhất tiêu chí, tester cung cấp phản hồi từ kết quả kiểm thử, developer điều chỉnh mã, và product owner xác nhận sản phẩm hoàn thành.
* **Lợi ích**: Tăng tính minh bạch, giảm lỗi do hiểu lầm yêu cầu, và đảm bảo sản phẩm đáp ứng đúng mong đợi.