**Thu thập yêu cầu** (Collect Requirements) là quy trình xác định, lập tài liệu, và quản lý nhu cầu và yêu cầu của các bên liên quan để đáp ứng các mục tiêu. Lợi ích chính của quy trình này là cung cấp cơ sở cho việc xác định ra phạm vi sản phẩm (product scope) và phạm vi dự án (project scope) nhằm hiểu rõ các yêu cầu để từ đó tạo ra một sản phẩm hoặc giải pháp đáp ứng được nhu cầu thực tế

Các phương thức thu thập yêu cầu phổ biến:

1. **Phỏng vấn cá nhân:** Trực tiếp nói chuyện với từng bên liên quan để hiểu rõ nhu cầu và mong muốn của họ.
2. **Họp nhóm (workshop):** Tổ chức các buổi thảo luận để thu thập yêu cầu từ nhiều người cùng lúc, đồng thời xây dựng sự đồng thuận.
3. **Quan sát:** Theo dõi cách người dùng thực hiện công việc hoặc quy trình để phát hiện ra các yêu cầu tiềm ẩn.
4. **Khảo sát/bảng câu hỏi:** Sử dụng câu hỏi viết hoặc trắc nghiệm để thu thập ý kiến từ một nhóm lớn người dùng.
5. **Phân tích tài liệu:** Xem xét các tài liệu hiện có như báo cáo, hợp đồng, hoặc hướng dẫn sử dụng để tìm ra các yêu cầu.
6. **Xây dựng mẫu thử (prototype):** Tạo phiên bản thử nghiệm của sản phẩm để nhận phản hồi từ người dùng.

**User story** là một công cụ trong phát triển phần mềm, đặc biệt trong phương pháp Agile, để mô tả ngắn gọn và dễ hiểu về một nhu cầu hoặc tính năng mà người dùng mong muốn từ sản phẩm. Nó tập trung vào việc xác định giá trị mà tính năng mang lại cho người dùng, giúp nhóm phát triển hiểu rõ hơn về mục tiêu của sản phẩm.

Các yếu tố cần đảm bảo của một User Story:

1. **Vai trò (Role):** Đối tượng sử dụng tính năng là ai? Ví dụ: người dùng, quản trị viên, khách hàng.
2. **Hành động (Action):** Người dùng muốn làm gì hoặc cần gì từ sản phẩm.
3. **Mục đích (Purpose):** Lý do hoặc mục tiêu mà người dùng muốn đạt được khi sử dụng tính năng.

Đặc điểm của một User Story tốt (INVEST):

* **Independent (Độc lập):** Story nên không quá phụ thuộc vào các story khác.
* **Negotiable (Có thể thảo luận):** Cần được điều chỉnh và hoàn thiện.
* **Valuable (Có giá trị):** Mang lại giá trị thực tế cho người dùng.
* **Estimable (Ước tính được):** Nhóm có thể đánh giá khối lượng công việc.
* **Small (Nhỏ gọn):** Đủ nhỏ để triển khai trong thời gian ngắn.
* **Testable (Kiểm tra được):** Có thể xác minh tính năng sau khi hoàn thành.

**Ví dụ:**

Là một người mua sắm trực tuyến, tôi muốn có chức năng lọc sản phẩm theo giá để chọn được sản phẩm tốt nhất trong ngân sách của tôi.

1. Use Case là gì?

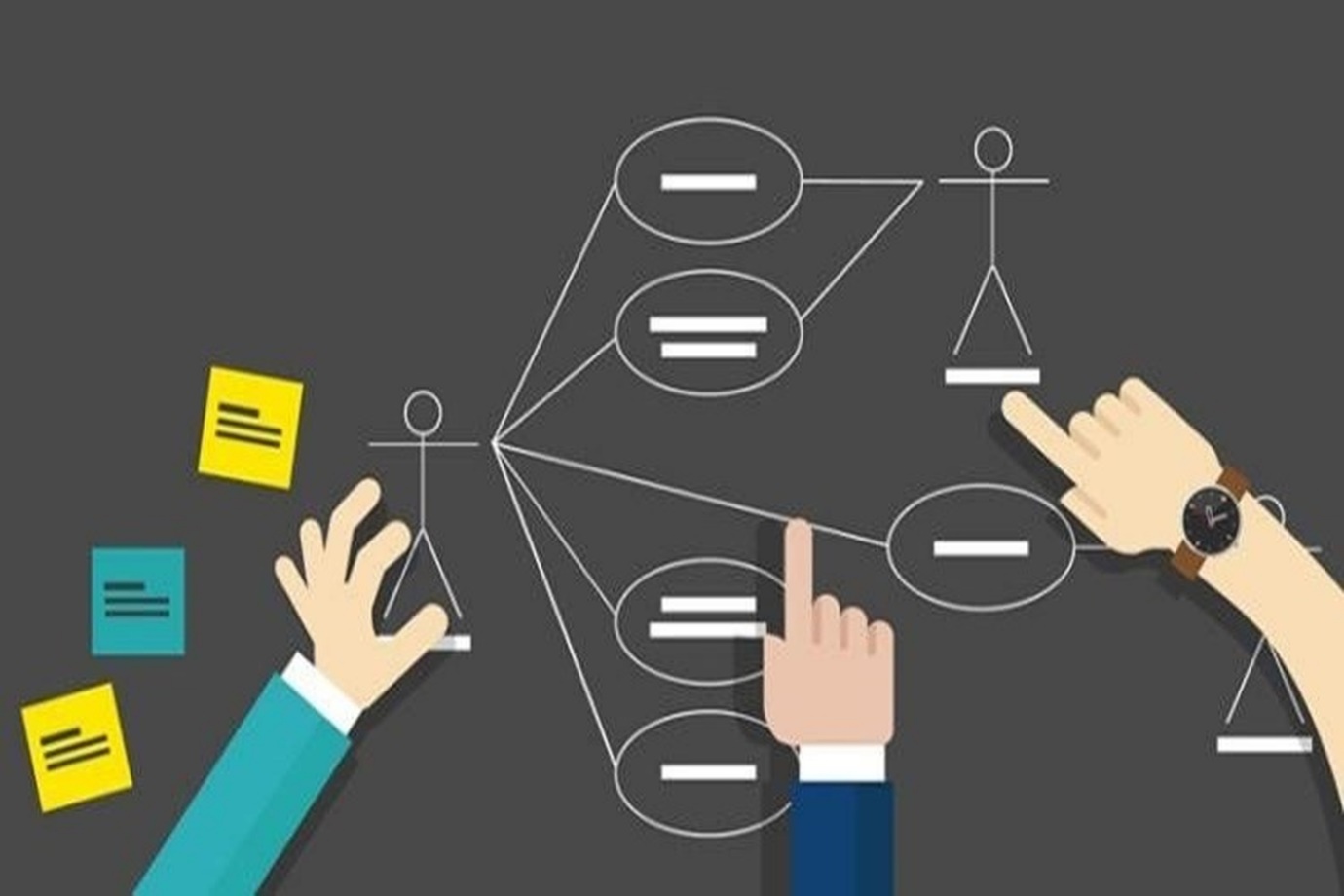
**Use Case** là một **mô tả chi tiết** về cách người dùng (actor) tương tác với hệ thống để đạt được một mục tiêu cụ thể. Nó giúp đội phát triển hiểu rõ **chức năng của hệ thống** từ góc nhìn người dùng.

📌 **Cấu trúc của một Use Case bao gồm:**

* **Tên Usecase**: Mô tả hành động chính.
* **Actor** (Người tham gia): Ai là người thực hiện hành động? Actor có thể là người dùng, hệ thống khác hoặc thiết bị.
* **Mô tả ngắn** (Description): Giải thích Use Case thực hiện điều gì.
* **Điều kiện tiên quyết** (Pre-conditions): Các điều kiện cần phải được đáp ứng trước khi use case có thể bắt đầu.
* **Điều kiện sau** (Post-conditions): Kết quả sau khi Use Case hoàn tất.
* **Luồng chính** (Main Flow): Chuỗi các bước chính mà actor thực hiện để đạt được mục tiêu.
* **Luồng thay thế** (Alternative Flow): Các trường hợp khác có thể xảy ra.

**Ví dụ**

|  |  |
| --- | --- |
| Usecase name | Đăng nhập |
| Description | Người dùng đăng nhập vào hệ thống để thực hiện các chức năng tương ứng, admin đăng nhập để thực hiện quản lý |
| Actor(s) | Khách hàng, Admin |
| Pre-Condition(s) | Khách hàng đã có tài khoản (Đã đăng kí)  Tài khoản được phân quyền (Đối với admin) |
| Post-Condition(s) | Người dùng/Admin đăng nhập thành công (Thông báo) |
| Basic Flow | 1. Người dùng chọn chức năng “Đăng nhập” trên trang web 2. Hệ thống hiển thị form đăng nhập 3. Người dùng nhập tài khoản, mật khẩu và nhấn đăng nhập 4. Hệ thống kiểm tra xem tên đăng nhập đã tồn tại chưa    1. Nếu đã tồn tại, kiểm tra mật khẩu       1. Nếu mật khẩu trùng khớp, thông báo thành công và chuyển đến trang chủ |
| Alternative Flow | 1. Người dùng chọn phương thức đăng nhập bằng Gmail 2. Hệ thống chuyển sang màn hình đăng nhập của google 3. Người dùng nhập/chọn tài khoản google và thực hiện đăng nhập 4. Google xác thực thông tin đăng nhập thành công |
| Exception Flow | 4.1.2. Nếu mật khẩu sai, báo lỗi và yêu cầu đăng nhập lại  4.2. Nêu tên đăng nhập không tồn tại, báo lỗi và yêu cầu đăng nhập lại |



**1. Actor (Tác nhân)**

* Là người dùng hoặc hệ thống bên ngoài tương tác với hệ thống mà biểu đồ mô tả.
* Biểu diễn bằng một hình người (stick figure).
* Có hai loại:
  + **Primary Actor (Tác nhân chính):** Người sử dụng trực tiếp hệ thống.
  + **Secondary Actor (Tác nhân phụ):** Các hệ thống hoặc con người hỗ trợ, gián tiếp tương tác.

**2. Use Case (Trường hợp sử dụng)**

* Là một chức năng cụ thể mà hệ thống cung cấp.
* Biểu diễn bằng hình elip và chứa tên chức năng.
* Ví dụ: "Đăng nhập", "Thanh toán hóa đơn".

**3. System Boundary (Ranh giới hệ thống)**

* Là đường viền hình chữ nhật bao quanh các Use Case để thể hiện phạm vi của hệ thống.
* Các Actor nằm bên ngoài hệ thống nhưng tương tác với các Use Case bên trong ranh giới.

**4. Relationships (Mối quan hệ)**

* Biểu diễn cách các Actor và Use Case liên kết với nhau:
  + **Association:** Đường thẳng nối Actor với Use Case mà nó tương tác.
  + **Include:** Mối quan hệ "bao gồm" (include) khi một Use Case cần một Use Case khác để thực hiện.
  + **Extend:** Mối quan hệ "mở rộng" (extend) được sử dụng khi một Use Case mở rộng chức năng của Use Case khác.
  + **Generalization:** Kế thừa quan hệ giữa các Actor hoặc Use Case.

**5. Notes (Ghi chú)**

* Biểu diễn thông tin bổ sung hoặc ghi chú về các yếu tố trong biểu đồ.
* Được biểu diễn bằng một hình chữ nhật nhỏ với góc gấp.