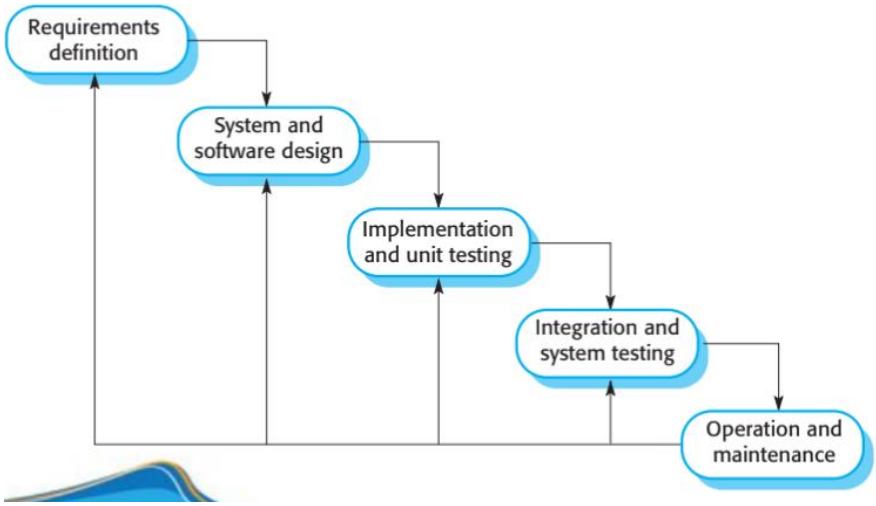
NHẬP MÔN CÔNG NGHỆ PHẦN MỀM
TÓM TẮT LÝ THUYẾT

1. **Phần mềm**
    a. **Phần mềm là gì?**
    Phần mềm bao gồm các chương trình máy tính và tài liệu liên quan. Luôn gắn với một hệ thống cụ thể.
    b. **Vai trò**
    Ảnh hưởng đến tất cả các khía cạnh của cuộc sống vì ngày càng nhiều hệ thống được điều khiển bằng phần mềm.
    c. **Phân loại sản phẩm phần mềm**
        i. **Sản phẩm dùng chung**
        Những hệ thống độc lập được chào bán trên thị trường và bất cứ ai cũng có thể mua chúng. Người phát triển hệ thống điều khiển đặc tả của hệ thống. Sự thay đổi phần mềm là do người phát triển quyết định.
        ii. **Sản phẩm đặt hàng**
        Phần mềm được phát triển cho một khách hàng cụ thể để đáp ứng nhu cầu của họ. Đặc tả về những gì phần mềm phải có là do khách hàng quyết định. Quyết định về sự thay đổi phần mềm là do khách hàng yêu cầu.
    d. **Tiêu chí của một phần mềm tốt**
        - **Tính bảo trì được**: phần mềm phải cải tiến được để đáp ứng những thay đổi về nhu cầu của khách hàng.
        - **Tính tin cậy được**: gồm các đặc tính: độ tin cậy, an toàn và bảo mật.
        - **Tính hiệu quả:** phần mềm không không nên sử dụng lãng phí các tài nguyên hệ thống
        - **Tính chấp nhận được**: phần mềm phải được chấp nhận bởi người dùng
2. **Mô hình quy trình phần mềm**
    a. **Quy trình phần mềm là gì**? Cái gì, ai, khi nào, tại sao?
    Một chuỗi các hoạt động để tạo ra một sản phẩm phần mềm. Bao gồm 4 hoạt động cơ bản chung nhất cho tất cả các quy trình phần mềm:
        - Đặc tả: định nghĩa phần mềm sẽ sản xuất và các ràng buộc về mặt chức năng của phần mềm
        - Phát triển: phần mềm được thiết kế và lập trình
        - Thẩm định: phần mềm được kiểm tra để đảm bảo là nó đáp ứng được yêu cầu người dùng
        - Cải tiến: phần mềm sẽ được thay đổi để đáp ứng sự thay đổi yêu cầu của người dùng và yêu cầu của thị trường
    b. **Mô hình quy trình phần mềm là gì?**
    Mô hình quy trình phần mềm là biểu diễn trừu tượng của một quy trình. Nó biểu diễn mô tả của quy trình từ 1 góc nhìn nào đó.
    c. **Các mô hình**
        i. Mô hình thác nước: cách tiếp cận truyền thống



1. Tuần tự
2. Đảm bảo hoàn thành một giai đoạn trước khi thực hiện giai đoạn tiếp theo
3. Phù hợp với các dự án có yêu cầu ổn định: loại bỏ sự lãng phí của dự án.
4. Các giai đoạn: xác định yêu cầu, thiết kế và phân tích, cài đặt, kiểm thử, triển khai, bảo trì và vận hành.
5. **Ưu điểm**:
    - Quy trình rõ ràng
    - Giúp cho việc phối hợp trong công việc dễ dàng hơn
6. **Nhược điểm:**
    - Khó khăn trong việc thích nghi với sự thay đổi sau khi quy trình đã vào guồng
    - Không linh động trong việc phân chia dự án thành những giai đoạn tách biệt

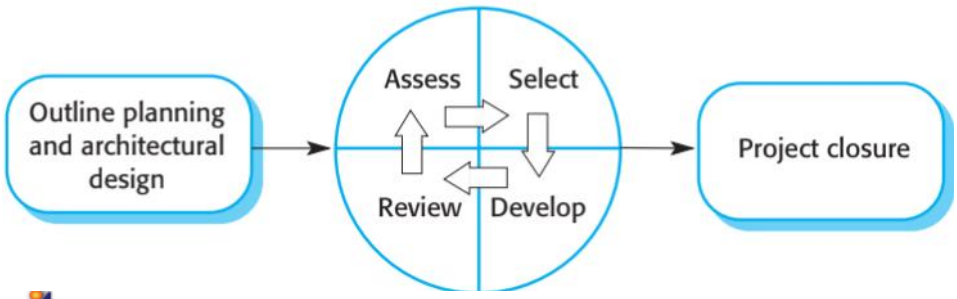
ii. RUP: cách tiếp cận truyền thống

1. Lặp đi lặp lại: làm mọi việc lặp đi lặp lại, tăng dần
    - Lặp trong từng pha
    - Lặp qua các pha
2. Các giai đoạn:
    - Khởi động (Inception): thiết lập business case cho hệ thống
    - Phát triển (Elaboration): nghiên cứu các vấn đề và phát triển kiến trúc hệ thống
    - Xây dựng (Construction): thiết kế hệ thống, lập trình và kiểm thử
    - Chuyển tiếp(Transition): triển khai hệ thống
3. Lặp lại = chu kỳ
    a. Một lần lặp lại giống như một chu kỳ của mô hình thác nước
4. Quy trình chi tiết để thực hiện mọi việc
5. Vai trò chi tiết của các thành viên trong nhóm
6. Nhiều tài liệu cần viết
7. Phát triển theo hướng Use-case

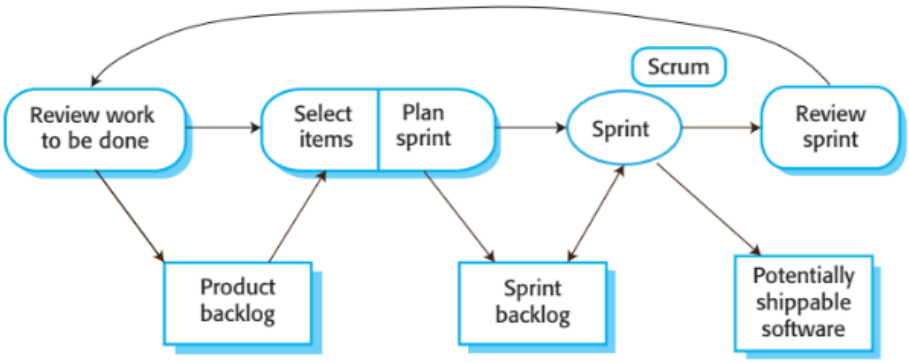
iii. Phương pháp linh hoạt(Agile)

1. Bốn mệnh đề giá trị của phương pháp Agile
    - Các cá nhân và tương tác hơn là quy trình và công cụ
    - Phần mềm hoạt động được hơn là tài liệu đầy đủ
    - Sự cộng tác của khách hàng hơn là thương lượng hợp đồng
    - trả lời nhanh sự thay đổi hơn là làm theo kế hoạch
2. Scrum

Quy trình Scrum



Scrum sprint cycle

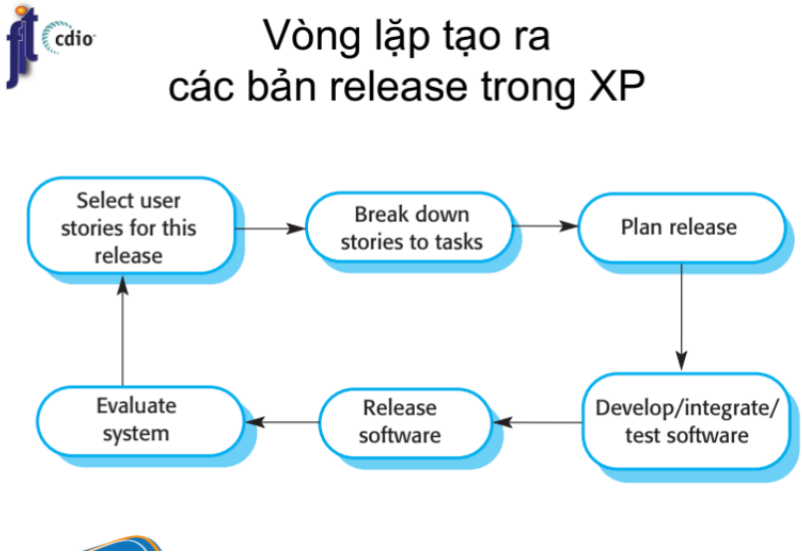


    a. Chia Sprint thường từ 2-4 tuần, lặp lại theo khung thời gian
    b. Chu trình Sprint: điểm bắt đầu cho kế hoạch là product backlog, là danh sách các công việc phải làm trong dự án.
        - Pha chọn: cả nhóm phát triển dự án làm việc với khách hàng để chọn ra các đặc tính và chức năng được cài đặt trong sprint.
        - Pha phát triển: Một khi các chức năng được lựa chọn, nhóm tự tổ chức để phát triển phần mềm.

- Pha duyệt: công việc đã thực hiện được duyệt lại và chuyển cho stakeholder. Chu trình sprint tiếp theo lại bắt đầu

c. Backlog(tồn đọng các công việc chưa hoàn thành, còn sót lại hoặc đang chờ giải quyết) (tồn đọng sản phẩm, tồn đọng sprint, tồn đọng trở ngại)
d. Vai trò
    - Product owner: Một cá nhân (có thể một nhóm nhỏ) chịu trách nhiệm nhận diện các đặc tính, chức năng của hệ thống, xếp độ ưu tiên, và review lại các product backlog để đảm bảo rằng dự án đáp ứng được các yêu cầu nghiệp vụ. Product Owner có thể là khách hang nhưng cũng có thể là một product manager trong một công ty phần mềm hoặc các đại diện stakeholder khác.
    - Scrum Master: ScrumMaster chịu trách nhiệm đảm bảo rằng quy trình Scrum được tuân thủ và chỉ dẫn nhóm sử dụng Scrum một cách hiệu quả. Người này chịu trách nhiệm là "giao diện" của nhóm, giao tiếp với phần còn lại của công ty để đảm bảo rằng nhóm Scrum không bị chi phối bởi yếu tố bên ngoài.
e. Lợi ích của Scrum
    - Sản phẩm được chia thành các phần nhỏ dễ hiểu và dễ quản lí
    - Các yêu cầu không ổn định sẽ không làm chậm trễ tiến độ
    - Toàn đội thấy được mọi thứ
    - Khách hàng nhận từng phần đúng hạn và gởi phản hồi về sản phẩm
    - Niềm tin giữa khách hàng và đội phát triển tăng lên và một văn hoá tích cực được tạo ra
f. Các hoạt động
    i. Scrum hàng ngày
    ii. Đánh giá sprint
    iii. kế hoạch sprint
    iv. Lập kế hoạch phát hành

3. XP
    a. Triết học: biến những thực hành tốt thành cực đoan
        - Các phiên bản mới có thể được xây dựng vài lần mỗi ngày;
        - Các phần được phân phối đến khách hàng hai tuần một lần;
        - Tất cả các test phải được chạy ở mỗi phiên bản và phiên bản đó chỉ được chấp nhận nếu các test đều thành công.



b. Practice chính
   i. Lập trình cặp
   ii. Sử dụng các user story để đặc tả
   iii. Sử dụng phương pháp TDD
   iv. Refactor mã nguồn
   v. phát triển theo hướng test

3. Quản lý dự án phần mềm

a. Mục tiêu

- phần mềm được phân phối đúng hạn, đúng lịch trình
- theo các yêu cầu của tổ chức phát triển và mua phần mềm
- tạo ra được phần mềm

b. Vai trò

i. PM

1. Nhiệm vụ:
- Xác định và quản lý yêu cầu dự án.
- Lập kế hoạch và quản lý tiến độ dự án.
- Quản lý nguồn lực và ngân sách.
- Quản lý rủi ro và vấn đề trong dự án.
- Tương tác với khách hàng và các bên liên quan khác.

2. Thành phẩm
- Kế hoạch dự án.
- Báo cáo tiến độ.
- Bảng phân công công việc.
- Biểu đồ Gantt.
- Kế hoạch đề xuất: Proposal

ii. BA

1. Nhiệm vụ:
- Nắm bắt yêu cầu từ khách hàng và người dùng.
- Phân tích và biên tập yêu cầu phần mềm.
- Xác định các quy trình kinh doanh và luồng công việc.
- Phân tích và thiết kế cơ sở dữ liệu.
- Đảm bảo rằng yêu cầu được hiểu đúng và đầy đủ cho đội phát triển phần mềm

2. Thành phẩm
- Biểu đồ Use case.
- Mô hình dữ liệu (Data model).
- Sơ đồ luồng công việc (Workflow diagram)
- Tài liệu phân tích yêu cầu

iii. Dev

1. Nhiệm vụ:
- Xây dựng và triển khai mã nguồn.
- Thực hiện kiểm thử đơn vị (unit testing).
- Tích hợp các thành phần phần mềm.
- Gỡ lỗi và sửa lỗi trong mã nguồn.
- Triển khai và cấu hình hệ thống.

2. Thành phẩm
- Mã nguồn và tài liệu hướng dẫn cài đặt (Installation guide).
- Tài liệu kỹ thuật (Technical documentation).
- Tài liệu triển khai (Deployment documentation).
- Tài liệu cấu hình hệ thống (System configuration documentation).

iv. Tester

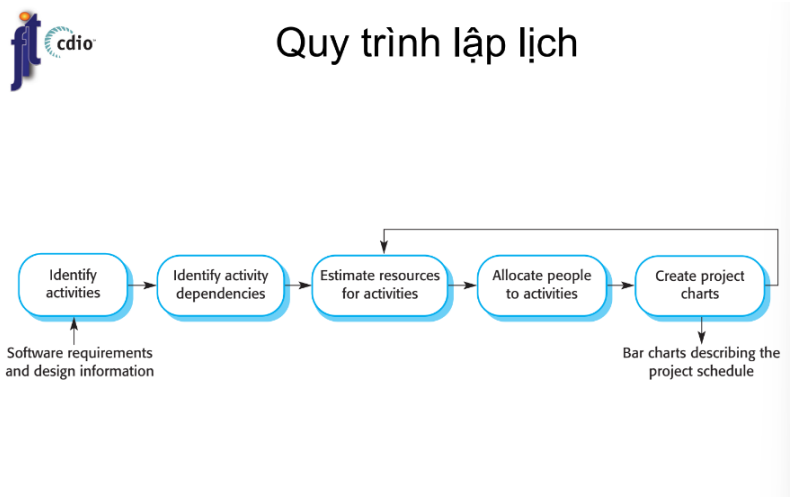
1. Nhiệm vụ:
- Xác định và lập kế hoạch kiểm thử.
- Tạo kịch bản kiểm thử và thiết kế bộ dữ liệu kiểm thử.
- Thực hiện kiểm thử hệ thống, kiểm thử chấp nhận và kiểm thử tích hợp.
- Ghi lại và báo cáo lỗi.
- Đảm bảo chất lượng phần mềm.

2. Thành phẩm
- Kế hoạch kiểm thử (Test plan).
- Kịch bản kiểm thử (Test script).
- Báo cáo kết quả kiểm thử (Test report).
- Báo cáo lỗi (Bug report).
- Tài liệu kiểm thử (Test documentation).

c. Các hoạt động

i. Lập kế hoạch: lên kế hoạch, ước lượng, lên lịch trình phát triển dự án và gán công việc cho các thành viên

1. Bao gồm: Phân rã công việc thành từng phần và gán công việc, Dự đoán các vấn đề phát sinh và dự kiến giải pháp cho chúng
2. Có các giai đoạn: đề xuất, khởi động dự án, định kỳ suốt dự án
3. Kế hoạch lập lịch:
   - Là quá trình đưa ra quyết định các công việc trong dự án được tổ chức thành từng tác vụ như thế nào, các tác vụ này được thực thi như thế nào và khi nào.
   - Ước lượng về thời gian và nguồn lực cần thiết để hoàn thành một tác vụ, nỗ lực cần thiết và ai sẽ thực hiện nó



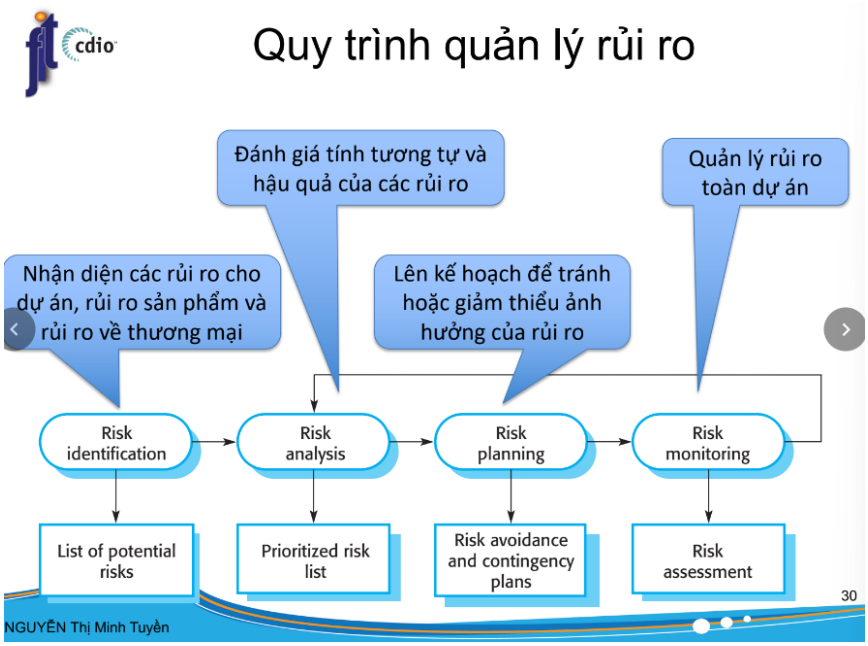
ii. Xây dựng đội nhóm
    1. tạo ra một nhóm gắn kết, cân bằng kỹ năng và cá tính
    2. tổ chức nhóm sao cho các thành viên làm việc cùng nhau một cách hiệu quả nhất.

iii. Quản lý con người: là một nhân tố quan trọng cho thành công của dự án
    1. Các nhân tố: tính nhất quán(công bằng), sự tôn trọng, tính bao hàm (quan tâm quan điểm của all thành viên), tính trung thực
    2. Thúc đẩy con người: tổ chức công việc và môi trường để khuyến khích con người làm việc hiệu quả.

iv. Kiểm soát và giám sát:
    1. Báo cáo: Người quản trị dự án thường chịu trách nhiệm viết báo cáo về tiến độ dự án cho khách hàng và cho người quản lý của công ty phát triển phần mềm.
    2. Giải quyết vấn đề

v. Sự hợp tác của khách hàng

vi. Quản lý rủi ro
    1. Bao gồm: nhận diện các rủi ro và lên kế hoạch để giảm thiểu ảnh hưởng của rủi ro lên dự án
    2. Các loại rủi ro: rủi ro dự án (ảnh hưởng lịch trình và nguồn lực), rủi ro sản phẩm (ảnh hưởng đến chất lượng hoặc hiệu năng của phần mềm), rủi ro thương mại (ảnh hưởng đến phát triển hoặc mua phần mềm)



3. Các rủi ro thường gặp: Rủi ro về công nghệ. Rủi ro về con người, Rủi ro về tổ chức, Rủi ro về công cụ, Rủi ro về yêu cầu, Rủi ro về việc ước lượng

4. Yêu cầu phần mềm & kỹ thuật yêu cầu

a. Yêu cầu phần mềm

i. Yêu cầu của người dùng (khái niệm vận hành) & yêu cầu hệ thống

1. Yêu cầu người sử dụng

- Những phát biểu (bằng ngôn ngữ tự nhiên kết hợp với các biểu đồ) về các dịch vụ mà hệ thống cung cấp và những ràng buộc về hoạt động của nó.

2. Yêu cầu hệ thống

- Một tài liệu có cấu trúc mô tả chi tiết chức năng của hệ thống, các dịch vụ và ràng buộc về hoạt động của hệ thống.
- Định nghĩa chính xác cái gì cần được cài đặt. Có thể là một phần của hợp đồng giữa khách hàng và người nhận thầu.

a. Use-case
b. Câu chuyện người dùng

ii. Yêu cầu chức năng và yêu cầu phi chức năng

- Yêu cầu chức năng: Mô tả chức năng và dịch vụ hệ thống cung cấp. Cần đảm bảo 2 nguyên tắc : hoàn chỉnh (định nghĩa tất cả dịch vụ người dùng yêu cầu) và nhất hoán (không có mâu thuẫn trong mô tả yêu cầu).
- Yêu cầu phi chức năng: Xác định những thuộc tính và ràng buộc của hệ thống (độ tin cậy, thời gian trả lời và yêu cầu về mặt lưu trữ, ...). Có thể quan trọng hơn yêu cầu chức năng. Ảnh hưởng đến cấu trúc của hệ thống hơn là component riêng lẻ.

iii. Yêu cầu về miền (chức năng và phi chức năng)

b. Kỹ thuật yêu cầu

i. Thu thập các yêu cầu
Kỹ sư phần mềm làm việc với các stakeholder
để tìm ra : miền ứng dụng, các dịch vụ hệ thống cung cấp, các ràng buộc để vận hành hệ thống.

1. Kỹ thuật
    a. Phỏng vấn
    b. Khảo sát
    c. Quan sát
    d. Ghi chú

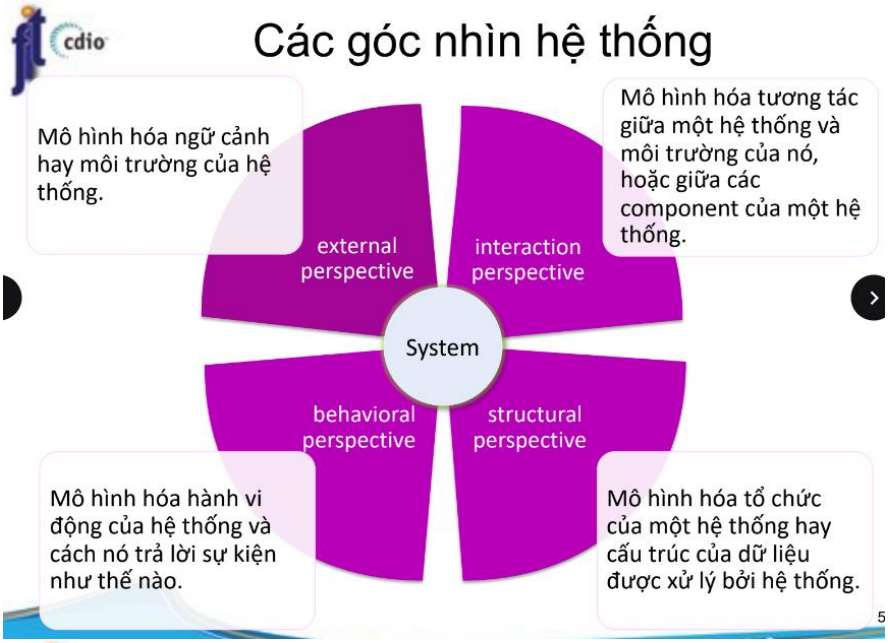
ii. Phân tích yêu cầu và tài liệu
1. Mô hình use-case
    - Use-case là kỹ thuật dựa vào kịch bản bằng ngôn ngữ UML chỉ ra các tác nhân trong một tương tác, mô tả chính tương tác đó.
    - Một tập các use case mô tả tất cả các tương tác có thể với hệ thống.

iii. Xác thực yêu cầu
Chứng tỏ rằng yêu cầu định nghĩa được hệ thống mà khách hàng cần.
1. Điều tra: Phân tích một cách có hệ thống các yêu cầu (không dùng công cụ tự động).

1. TDD – Test-driven development (test generation) - Phát triển dựa trên thử nghiệm (tạo thử nghiệm): Phát triển các test cho các yêu cầu để kiểm tra khả năng test được hay không.
2. Tạo nguyên mẫu (PoC): Sử dụng một mô hình chạy được của hệ thống để kiểm tra các yêu cầu.

iv. Quản lý yêu cầu
- Là quy trình quản trị sự thay đổi yêu cầu trong suốt quá trình công nghệ yêu cầu và phát triển hệ thống.
- Các yêu cầu mới phát sinh khi hệ thống đang được phát triển và cả khi nó được đưa vào sử dụng.
- Cần theo dõi những yêu cầu đơn lẻ và duy trì mối liên hệ giữa các yêu cầu phụ thuộc nhau.
- Cần thiết lập một quy trình hình thức cho những đề nghị thay đổi và tạo mối liên hệ giữa yêu cầu này với các yêu cầu hệ thống.
- Kế hoạch quản lý: Định danh yêu cầu, quy trình quản lý sự thay đổi (đánh giá sự ảnh hưởng và chi phí của thay đổi), các chính sách lần vết, công cụ hỗ trợ

v. Câu hỏi: tại sao chúng ta cần thực hiện xác nhận yêu cầu? Tại sao việc kiểm tra/xem xét yêu cầu lại hiệu quả?



5. Phân tích và thiết kế phần mềm

a. Kiến trúc phần mềm

i. Câu hỏi: tại sao chúng ta cần chi tiết/hình thành kiến trúc cho phần mềm? Thiết kế kiến trúc liên quan đến việc hiểu một hệ thống được tổ chức như thế nào và thiết kế toàn bộ kiến trúc của hệ thống đó. Là cầu nối giữa yêu cầu phần mềm và thiết kế, giúp nhận diện các component chính của hệ thống và cách thức giao tiếp giữa các component

ii. Khả năng mở rộng ->chiều ngang và chiều dọc

iii. Yêu cầu quan trọng về kiến trúc

1. Hiệu năng: Định vị các chức năng quan trọng trong một số ít component và giảm thiểu giao tiếp. Những component này được triển khai trên cùng một máy tính.
2. Bảo mật: Sử dụng kiến trúc phân tầng với các phần quan trọng được đặt ở các lớp trong cùng.
3. An toàn (Safety): Định vị các thao tác liên quan đến an toàn trong một số ít các hệ thống con.
4. Tính thường trực: Thiết kế sẵn các component dư thừa sao cho có thể thay thế hoặc cập nhật các component mà không phải dừng hệ thống, nghĩa là đảm bảo cho hệ thống hoạt động liên tục.
5. Tính dễ bảo trì: Sử dụng các component nhỏ, chi tiết, có thể thay thế được.

b. Thiết kế kiến trúc (thiết kế kiến trúc)

i. Căn cứ vào yêu cầu hình thành kiến trúc
ii. Câu hỏi: kiến trúc ảnh hưởng như thế nào đến hiệu suất của phần mềm?
iii. Liên kết lỏng lẻo và liên kết chặt chẽ
    1. Tight Coupling (phụ thuộc chặt chẽ): Khi một lớp phụ thuộc chặt chẽ vào lớp khác, nghĩa là bất kỳ thay đổi nào trong lớp này cũng có thể ảnh hưởng đến lớp kia. Điều này làm giảm khả năng maintenance, tái sử dụng (reusable) và mở rộng (extendable) của mã nguồn (source code).
    2. Loose Coupling (phụ thuộc lỏng lẻo): Khi một lớp phụ thuộc lỏng lẻo vào lớp khác, nghĩa là nó chỉ phụ thuộc vào một phần nhỏ của lớp kia, thường là thông qua interface hoặc lớp trừu tượng (abstraction). Điều này giúp mã nguồn dễ dàng mở rộng (extendable), tái sử dụng (reusability) và bảo dưỡng (maintenance) hơn.

c. Thiết kế chi tiết (thiết kế cấp thấp)

i. Thiết kế cấp cao và thiết kế chi tiết?
    1. Thiết kế kiến trúc (còn được gọi là thiết kế mức cao): là phát triển mức kiến trúc cao nhất và đưa ra cách tổ chức phần mềm và chỉ ra các thành phần khác nhau trong phần mềm
    2. Thiết kế chi tiết: chỉ ra chi tiết và đầy đủ về thành phần tạo điều kiện xây dựng phần mềm trong pha sau đó
ii. Sơ đồ lớp: Được sử dụng khi phát triển một mô hình hệ thống hướng đối tượng, để chỉ ra các lớp trong một hệ thống và mối liên hệ giữa các lớp đó.
iii. Sơ đồ trình tự: Được sử dụng để mô hình hóa tương tác giữa các actor và các đối tượng trong một hệ thống. Chỉ ra một chuỗi tuần tự các tương tác xảy ra trong một use case cụ thể nào đó hoặc một trường hợp của use case.

d. Một số mô hình kiến trúc mẫu

i. Model-View-Controller (MVC)

Mô hình MVC

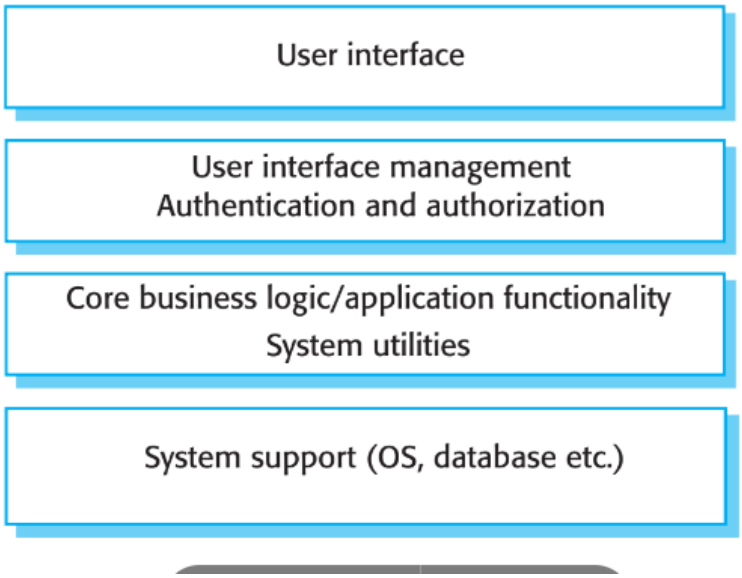
Tên	Mô hình MVC (Model-View-Controller)
Mô tả	Tách riêng phần biểu diễn và phần tương tác ra khỏi dữ liệu hệ thống. Ba component tương tác với nhau. • **Model component**: quản lý dữ liệu hệ thống và các thao tác trên dữ trên dữ liệu đó. • **View component**: định nghĩa và quản lý cách dữ liệu được biểu diễn tới người dùng như thế nào. • **Controller component**: Quản lý tương tác người dùng (ví dụ như ấn phím, nhấp chuột, ...) và chuyển các tương tác này tới View và Model.
Sử dụng khi nào	• Khi có nhiều cách biểu diễn và tương tác với dữ liệu. • Khi chưa biết được các yêu cầu tương lai cho tương tác và biểu diễn dữ liệu.
Ưu điểm	Cho phép dữ liệu thay đổi độc lập với hiển thị và ngược lại. Hỗ trợ biểu diễn theo nhiều cách khác nhau trên cùng một dữ liệu.
Nhược điểm	Có thể chứa code bổ sung và code sẽ phức tạp hơn khi mô hình dữ liệu và mô hình tương tác đơn giản.

ii. Kiến trúc phân tầng

Mô hình kiến trúc phân tầng

Tên	Kiến trúc phân tầng
Mô tả	Tổ chức hệ thống thành các tầng, mỗi tầng chứa các chức năng liên quan đến nhau. Một tầng cung cấp các dịch vụ cho tầng trên của nó vì vậy các tầng thấp nhất biểu diễn các dịch vụ lõi được sử dụng trong toàn bộ hệ thống.
Sử dụng khi nào	• Khi xây dựng các tính năng mới dựa trên những hệ thống có sẵn; • Khi việc phát triển được dàn trải trên nhiều nhóm khác nhau và mỗi nhóm chịu trách nhiệm về chức năng của một tầng; • Khi có yêu cầu về bảo mật ở nhiều mức độ.
Ưu điểm	• Cho phép thay thế các phần miễn là interface được duy trì. • Các chức năng dư thừa (ví dụ như phân quyền) có thể được cung cấp ở mỗi tầng để tăng độ tin cậy của hệ thống.
Nhược điểm	• Thực tế: cung cấp một sự phân chia rõ rệt giữa các tầng thường rất khó khăn và tầng cao hơn có thể tương tác trực tiếp với tầng thấp hơn hơn là thông qua một tầng bên dưới nó. • Hiệu năng cũng có thể là một vấn đề vì có nhiều mức diễn giải của một yêu cầu dịch vụ khi nó được thực hiện tại mỗi tầng.

Một kiến trúc phân tầng tổng quát

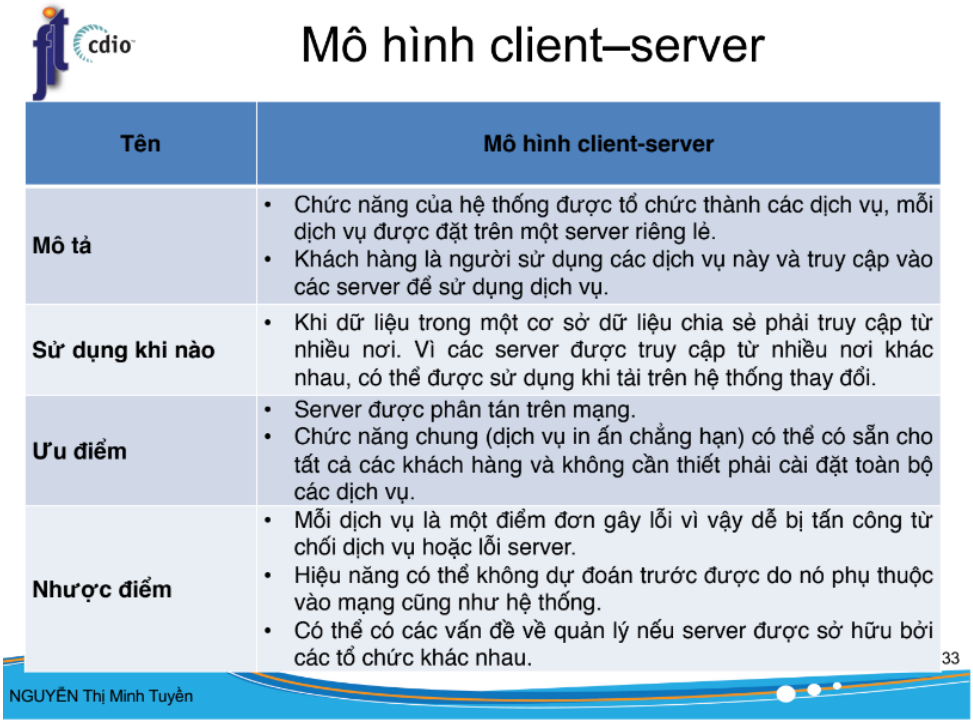


iii. Repository

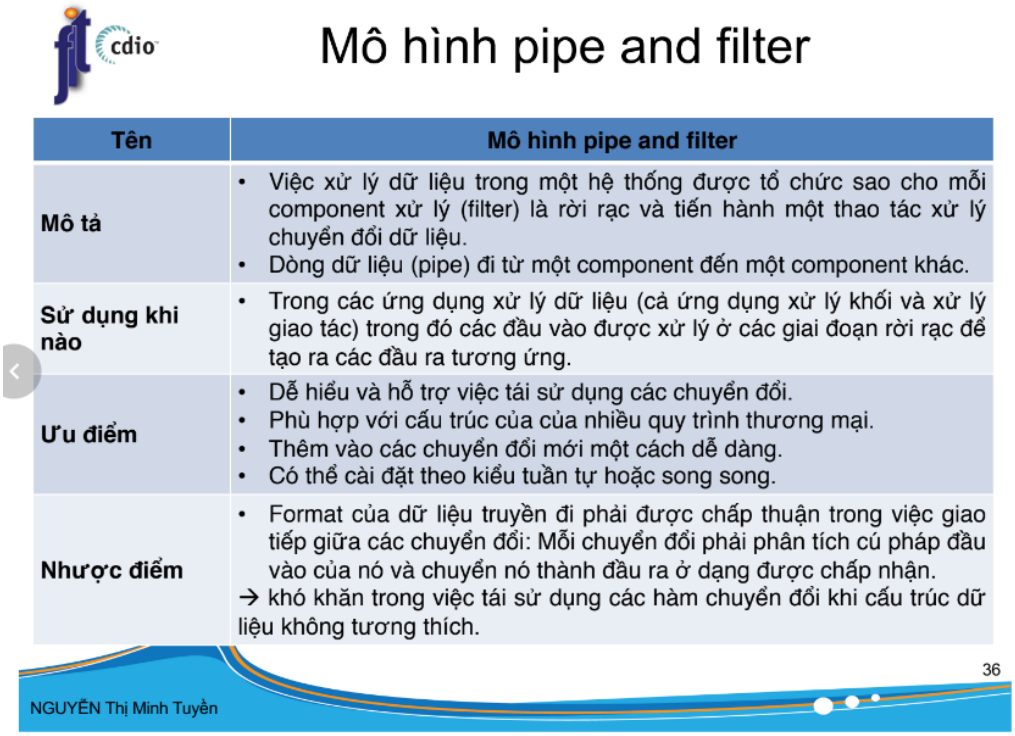
Mô hình Repository

Tên	Mô hình Repository
Mô tả	• Tất cả các dữ liệu trong hệ thống được quản lý ở một kho trung tâm, kho này được truy cập bởi tất cả các component của hệ thống. • Các component không tương tác trực tiếp với nhau, chỉ thông qua kho chung thôi.
Sử dụng khi nào	• Khi ta có một hệ thống trong đó một lượng lớn thông tin sinh ra phải được lưu trữ trong một thời gian dài. • Sử dụng trong các hệ thống hướng dữ liệu
Ưu điểm	• Các component có thể độc lập với nhau – chúng không cần biết sự tồn tại của các component khác. Các thay đổi xảy ra ở một component không ảnh hưởng tới các component khác. • Tất cả các dữ liệu có thể được quản lý một cách nhất quán (ví dụ như backup dữ liệu được thực hiện đồng thời) vì tất cả dữ liệu được lưu trữ ở cùng một nơi.
Nhược điểm	• Các vấn đề xảy ra trên kho chung ảnh hưởng đến toàn hệ thống. • Có thể không hiệu quả trong việc tổ chức các giao tiếp thông qua kho. • Phân tán kho trên nhiều máy tính có thể khó khăn.

iv. Client-server



v. Pipe and filter



6. Kiểm định và thẩm định phần mềm

a. Kiểm định (verification) và thẩm định (validation)

- Kiểm định: Chỉ ra các tình huống trong đó các hành vi của phần mềm không đúng, không như mong đợi hoặc không tương thích với **đặc tả**. ("Are we building the product right". Phần mềm phải tương thích với đặc tả.)

- Thẩm định: Chỉ ra cho người phát triển và khách hàng rằng phần mềm thỏa mãn các yêu cầu đưa ra bởi **khách hàng**. ("Are we

building the right product". Phần mềm phải thỏa mãn được những gì người dùng thật sự yêu cầu.)

i. Tại sao chúng ta cần cả hai?
Nhằm mục đích chỉ ra rằng: Một hệ thống tuân theo đặc tả của nó và thoả mãn yêu cầu của khách hàng hệ thống

b. Kỹ thuật

i. Kiểm tra tĩnh (Software inspection)
- Liên quan đến việc phân tích các biểu diễn tĩnh của hệ thống để tìm ra lỗi (static verification).

1. Đánh giá (hiện vật, sản phẩm công việc)
2. Phân tích (phân tích mã)

ii. Kiểm thử (Software testing)
- Liên quan đến việc thực hiện và quan sát hành vi của sản phẩm (dynamic verification).
- Hệ thống được thực thi với dữ liệu kiểm thử và quan sát hành vi hoạt động của hệ thống.

1. Kiểm tra
2. Phân tích

iii. Phát triển hướng thử nghiệm

Test-driven development (TDD)
- Là phương pháp trong đó việc phát triển mã nguồn và kiểm thử đan xen nhau.
- Các test được viết trước khi lập trình và phải "pass"các test là yếu tố quan trọng.
- Phát triển mã nguồn theo kiểu tăng dần, song song với việc kiểm thử cho từng phần đó.
- Không thể chuyển sang cài đặt phần tiếp theo cho đến khi mã nguồn đang phát triển "pass" tất cả các test của nó.

Lợi ích:
- Bao phủ mã nguồn
- Kiểm thử hồi quy (Regression testing)
- Đơn giản hóa việc sửa lỗi
- Là tài liệu hệ thống

iv. Phát triển theo mô hình

v. …

7. Kiểm thử phần mềm

a. Mức độ kiểm thử

i. Kiểm thử đơn vị
- Là quy trình kiểm thử từng component riêng lẻ.
- Là quy trình kiểm thử tìm lỗi.
- Các đơn vị có thể là:

+ Các hàm hay phương thức đơn lẻ trong một đối tượng.
+ Các lớp đối tượng chứa vài thuộc tính và phương thức.
+ Các component với các giao diện được định nghĩa sẵn để truy cập vào các tính năng của chúng.

- Kiểm thử lớp đối tượng
  - Để kiểm thử bao phủ một lớp đối tượng:
  - Kiểm thử tất cả các thuộc tính liên quan.
  - Thiết lập và kiểm thử giá trị của tất cả các thuộc tính.
  - Thực thi đối tượng với tất cả các trạng thái có thể.
  - Tính kế thừa làm cho việc thiết kế các kiểm thử lớp đối tượng trở nên khó khăn vì thông tin cần kiểm thử không được định vị.
- Kiểm thử tự động
  - Nếu có thể, nên tự động hóa việc kiểm thử đơn vị để các test được chạy và kiểm tra mà không cần sự can thiệp của con người.
  - Sử dụng các framework hỗ trợ kiểm thử tự động (JUnit chẳng hạn) để viết và chạy chương trình test.

ii. Thử nghiệm hệ thống

- Kiểm thử hệ thống tập trung vào tương tác => có thể sử dụng biểu đồ use case cơ sở để kiểm thử hệ thống.
- Biểu đồ tuần tự cũng có thể được sử dụng.

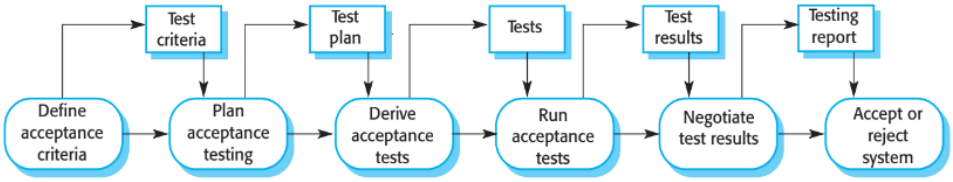
1. Tích hợp
   - Tích hợp các component để tạo ra một phiên bản của hệ thống và sau đó kiểm thử hệ thống được tích hợp.
   - Tập trung vào việc kiểm thử tương tác giữa các component.
   - Kiểm tra rằng các component tương thích với nhau, tương tác đúng và chuyển đúng dữ liệu, đúng thời điểm thông qua giao diện của chúng.
   - Trong suốt quá trình kiểm thử hệ thống, các component sử dụng lại được tích hợp với các component đang phát triển. Hệ thống hoàn chỉnh được kiểm thử.
   - Các component được phát triển bởi các thành viên khác nhau được tích hợp lại trong giai đoạn này.
   - Trong một số công ty, kiểm thử hệ thống có thể được thực hiện bởi một nhóm độc lập không tham gia vào việc thiết kế và cài đặt.
2. Phát hành thử nghiệm

iii. Kiểm thử chấp nhận

Acceptance testing

- Khách hàng kiểm thử hệ thống để quyết định xem hệ thống này có được chấp nhận để triển khai đến môi trường làm việc của khách hàng hay không.

Quy trình acceptance testing



b. Các loại kiểm thử

i. Kiểm thử chức năng
ii. Phi chức năng
    1. Hiệu suất
    2. Trọng tải
    3. Khả năng sử dụng
    4. Bảo vệ

c. Kỹ thuật

i. Kiểm thử hồi quy
    - Là việc kiểm thử hệ thống để kiểm tra rằng sự thay đổi không phá vỡ việc cài đặt mã nguồn trước đó.
    - Kiểm thử hồi quy bằng tay rất tốn kém.
    - Kiểm thử hồi quy tự động đơn giản và trực tiếp. Tất cả các test đều được thực thi lại mỗi khi có sự thay đổi trong chương trình.
    - Các test phải được thực thi thành công trước khi chấp nhận một thay đổi.

ii. Kiểm thử Ad-hoc
iii. Kiểm thử khói
iv. ....

d. Các khái niệm

i. Test case
ii. Các bước kiểm thử
iii. Dữ liệu kiểm thử
iv. Kết quả kiểm thử
v. Khiếm khuyết
vi. Kiểm tra phạm vi
    1. Phạm vi mã nguồn
    2. Phạm vi chức năng
    3. Phạm vi giao diện người dùng
    4. Phạm vi đường dẫn

e. Kỹ thuật

i. Dựa trên yêu cầu test cases
ii. phân vùng tương đương test cases
    1. VD. Tạo các test cases để kiểm tra chức năng đăng ký với tên người dùng có ít nhất 8 ký tự và một ký tự đặc biệt.
iii. Kiểm thử theo chỉ dẫn
    1. Đưa ra một đoạn mã, viết các trường hợp kiểm thử để bao gồm tất cả các chỉ dẫn
iv. Thử nghiệm đặc biệt
v. Kiểm tra khói

8. Tự động hóa thử nghiệm

a. Mục tiêu
b. Các phương pháp tự động hóa thử nghiệm
c. Cấp độ
    i. Kiểm tra đơn vị
    ii. Thử nghiệm tích hợp
    iii. Thử nghiệm hệ thống
d. Công cụ
    i. Selenium
    ii. Appium
    iii. Katalon

9. Thiết kế giao diện người dùng

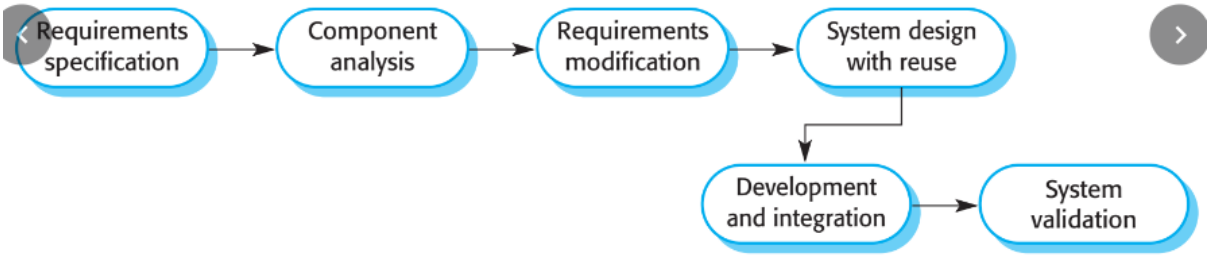
a. Nguyên tắc thiết kế giao diện người dùng

Phải xem xét nhu cầu, kinh nghiệm và khả năng của người dùng hệ thống. Nhận thức được các hạn chế về vật lý và tinh thần của người dùng và chấp nhận rằng ai cũng có thể nhầm lẫn. Các nguyên tắc chính trong thiết kế UI đóng vai trò nền tảng cho thiết kế giao diện dù không phải tất cả các nguyên lý có thể áp dụng cho tất cả các thiết kế.

i. Thân thiện với người dùng: Sử dụng các thuật ngữ và khái niệm hướng người dùng.
ii. Nhất quán: Hệ thống nên hiển thị một cách nhất quán.
iii. Ít bất ngờ: Nếu một lệnh được thực hiện theo cách thông thường, người dùng có thể dự đoán được thao tác của các lệnh tương tự.
iv. Có thể khôi phục được: Khi gặp lỗi: Hệ thống nên cung cấp một số cơ chế khôi phục tình trạng hoạt động bình thường trước khi gặp lỗi.
v. Hướng dẫn người dùng: Cung cấp một số hướng dẫn người dùng
vi. Đa dạng người dùng: Cung cấp tiện ích tương tác cho các loại người dùng khác nhau.

10. Tái sử dụng phần mềm

CNPM theo hướng tái sử dụng



a. Đặc điểm

- Dựa vào việc tái sử dụng một cách có hệ thống
- Hệ thống được tích hợp từ những thành phần có sẵn hoặc từ các hệ thống COTS (Commercial-off-the-shelf).

b. Các giai đoạn của quy trình

i. Phân tích component;
ii. Bổ sung yêu cầu;
iii. Thiết kế hệ thống với việc tái sử dụng;
iv. Phát triển và tích hợp.

Hiện nay, việc tái sử dụng là phương pháp chuẩn cho việc xây dựng nhiều hệ thống thương mại.

Các câu hỏi lý thuyết từng gặp:

Câu 1:

a) Hãy liệt kê các công việc mà trưởng dự án (Project Manager) phải thực hiện trong quá trình phát triển phần mềm.

b) Hãy liệt kê các thành phẩm (artifact) được tạo ra bởi trưởng dự án trong quá trình phát triển phần mềm.

a) Các công việc:

- Xác định và quản lý yêu cầu dự án.
- Lập kế hoạch và quản lý tiến độ dự án.
- Quản lý nguồn lực và ngân sách.
- Quản lý rủi ro và vấn đề trong dự án.
- Tương tác với khách hàng và các bên liên quan khác.

b) Các thành phẩm

- Kế hoạch dự án
- Kế hoạch đề xuất Proposal
- Báo cáo tiến độ.
- Biểu đồ Gantt và bảng phân công công việc

Câu 2:

a) Hãy phân biệt yêu cầu chức năng và yêu cầu phi chức năng.

- Yêu cầu chức năng: Cho biết hệ thống có thể thực hiện được những việc gì, cách hệ thống xử lý các đầu vào cụ thể
- Yêu cầu phi chức năng: Là những ràng buộc đối với dịch vụ hay chức năng nào đó của hệ thống. Thường là những ràng buộc về mặt thời gian phản hồi, về độ tin cậy … Yêu cầu phi chức năng thường được áp dụng cho cả hệ thống chứ không chỉ một chức năng nào đó

b) Các yêu cầu phi chức năng ảnh hưởng như thế nào đến việc thiết kế kiến trúc (software architecture) của một dự án phần mềm? Nêu ít nhất 03 ví dụ thể hiện sự ảnh hưởng này.

- Các yêu cầu về độ bảo mật, tính an toàn, tính dễ bảo trì, .. tác động đến việc chọn lựa kiến trúc của hệ thống
- Ví dụ:
    - Hệ thống yêu cầu độ bảo mật cao → Kiến trúc phân tầng là lựa chọn hợp lý
    - Tất cả dữ liệu yêu cầu sự nhất quán → Kiến trúc Repository
    - Các chức năng có sẵn cho tất cả khách hàng và không cần cài đặt toàn bộ → Kiến trúc client - server

Câu 3:

a) Có ý kiến cho rằng tái sử dụng là cách tiếp cận tốt nhất để phát triển phần mềm. Bạn đồng ý với ý kiến này ở mức độ nào ? Hãy viết một đoạn văn (ít nhất 150 từ) để thể hiện quan điểm của bạn.
b) Có ý kiến cho rằng mô hình thác nước là một mô hình đã lỗi thời và không nên sử dụng nó khi phát triển phần mềm. Bạn đồng ý với ý kiến này ở mức độ nào ? Hãy viết một đoạn văn (ít nhất 150 từ) để thể hiện quan điểm của bạn.

a) Em chỉ đồng ý một phần với ý kiến trên. Tái sử dụng đúng là một hướng tiếp cận rất tốt bởi ưu điểm của nó. Tái sử dụng các component giúp dự án giảm chi phí, thời gian và công sức trong việc thiết kế, phát triển và kiểm thử tất cả mã nguồn. Tái sử dụng giúp chúng ta có thể tạo ra được phần mềm chuẩn, giảm thiểu được rủi ro về lỗ hỗng bảo mật hay lỗi vì chúng đã được kiểm tra và xử lí một cách kĩ càng trước khi đưa vào sử dụng rộng rãi. Từ đó phần mềm sẽ được phát triển với hiệu suất tốt hơn và tăng được khả năng cạnh tranh. Nhưng không thể nói Tái sử dụng là "tốt nhất", vì khi sử dụng quá nhiều component, ta sẽ tốn công sức tìm hiểu chúng và nghiên cứu về cách thức tích hợp chúng lại với nhau. Ngoài ra, chúng ta còn có thể bị phụ thuộc vào rất nhiều bên liên quan khi các component cập nhật, xảy ra lỗi hay bị ngưng hỗ trợ và ta phải tốn chi phí cho việc khắc phục. Tóm lại, việc tái sử dụng là rất hữu ích cho việc phát triển phần mềm, nhưng ta không thể hoàn toàn áp dụng tái sử dụng ở mọi trường hợp mà phải sử dụng cho hợp lý.

b) Mô hình thác nước là một trong những mô hình đã được áp dụng để phát triển phần mềm. Em không đồng ý với ý kiến "Có ý kiến cho rằng mô hình thác nước là một mô hình đã lỗi thời và không nên sử dụng nó khi phát triển phần mềm." Mô hình vẫn còn mang trong mình những ưu điểm có giá trị. Mô hình thác nước tách quá trình thành các giai đoạn rõ ràng như yêu cầu, thiết kế, phát triển và kiểm thử, qua luôn đảm bảo giai đoạn này kết thúc trước khi bước vào giai đoạn kia. Nó giúp cho dự án có kế hoạch, quy trình rõ ràng và đảm bảo được tính nhất quán và kiểm soát trong quá trình phát triển. Mô hình này rất thích hợp để áp dụng vào các dự án mà yêu cầu được hiểu rõ và ổn định trong suốt thời gian phát triển. Trong các hệ thống lớn được phát triển ở nhiều địa điểm khác nhau, mô hình này giúp phân chia rõ ràng giữa các bộ phận và giúp cho việc phối hợp trong công việc dễ dàng hơn. Tuy nhiên, đối với một số dự án như hệ thống thương mại có thường có yêu cầu dễ thay đổi thì mô hình này không thể thích hợp bằng các mô hình phát triển linh hoạt vì khó khăn và không linh động trong việc đáp ứng của mô hình với các thay đổi đó. Tóm lại, dù không phải là lỗi thời nhưng mô hình thác nước nên được áp dụng một cách hợp lý trong các dự án phù hợp.

Câu 4:
a) Hãy liệt kê các công việc mà developer và tester phải thực hiện trong quá trình phát triển phần mềm.
b) Hãy liệt kê các thành phẩm (artifact) được tạo ra bởi developer và tester trong quá trình phát triển phần mềm.

Developer

a) Nhiệm vụ:

- Xây dựng và triển khai mã nguồn.
- Thực hiện kiểm thử đơn vị (unit testing).
- Tích hợp các thành phần phần mềm.
- Gỡ lỗi và sửa lỗi trong mã nguồn.
- Triển khai và cấu hình hệ thống.

b) Thành phẩm

- Mã nguồn và tài liệu hướng dẫn cài đặt (Installation guide).
- Tài liệu kỹ thuật (Technical documentation).
- Tài liệu triển khai (Deployment documentation).
- Tài liệu cấu hình hệ thống (System configuration documentation).
- Báo cáo lỗi cho kiểm thử đơn vị

Tester

a) Nhiệm vụ:

- Xác định và lập kế hoạch kiểm thử.
- Tạo kịch bản kiểm thử và thiết kế bộ dữ liệu kiểm thử.
- Thực hiện kiểm thử hệ thống, kiểm thử chấp nhận và kiểm thử tích hợp.
- Ghi lại và báo cáo lỗi.
- Đảm bảo chất lượng phần mềm.

b) Thành phẩm

- Kế hoạch kiểm thử (Test plan).
- Kịch bản kiểm thử (Test case).
- Báo cáo kết quả kiểm thử (Test report).
- Báo cáo lỗi (Bug report).
- Tài liệu kiểm thử (Test documentation).

Câu 5:

a) Hãy phân biệt verification và validation trong kiểm thử phần mềm. Vì sao một dự án phần mềm phải thực hiện cả verification và validation trong suốt quá trình phát triển

b) Hãy liệt kê những điểm khác nhau cơ bản giữa các quy trình phát triển phần mềm truyền thống (đại diện là Waterfall) với các quy trình phát triển linh hoạt (Agile, đại diện là Scrum)

a) Phân biệt;

- Kiểm định: Chỉ ra các tình huống trong đó các hành vi của phần mềm không đúng, không như mong đợi hoặc không tương thích với **đặc tả**. ("Are we building the product right". Phần mềm phải tương thích với đặc tả.)
- Thẩm định: Chỉ ra cho người phát triển và khách hàng rằng phần mềm thỏa mãn các yêu cầu đưa ra bởi **khách hàng**. ("Are we building the right product". Phần mềm phải thỏa mãn được những gì người dùng thật sự yêu cầu.)

Tại sao chúng ta cần cả hai?

- Nhằm mục đích chỉ ra rằng: Một hệ thống tuân theo đặc tả của nó và thoả mãn yêu cầu của khách hàng, mục đích của phần mềm. mong đợi của người dùng và môi trường thương mại

b) Phân biệt:

Quy trình hoạch định sẵn	Quy trình linh hoạt
Tất cả các hoạt động được lên kế hoạch trước và tiến độ thực hiện được đánh giá dựa vào kế hoạch này	Kế hoạch được phát triển dần dần và dễ dàng thay đổi quy trình để đáp ứng sự thay đổi yêu cầu của khách hàng
Các pha đặc tả và phát triển phân biệt và tách rời nhau	Các pha đặc tả, phát triển và thẩm định đan xen nhau.
Quy trình rõ ràng	Quy trình không rõ ràng
Không linh động, khó khăn trong việc thích nghi với sự thay đổi sau khi quy trình đã vào guồng.	Giảm được chi phí khi đáp ứng sự thay đổi yêu cầu của khách hàng, dễ dàng lấy phản hồi của khách hàng
Khách hàng chỉ được phân phối sản phẩm khi phần mềm khi các pha được thực hiện xong	Phân phối phiên bản mới thường xuyên để đánh giá và có sự tham gia của stakeholder.
Đầu ra ở mỗi giai đoạn đã được lên kế hoạch trước	Đầu ra từ quy trình phát triển được quyết định thông qua việc thương lượng trong suốt quá trình phát triển phần mềm.

Câu 6:

a) Hãy phân biệt quy trình hoạch định sẵn(plan-driven process) và quy trình linh hoạt(agile process)

b) Hãy chọn một quy trình hoạch định sẵn mà em đã học: mô tả quy trình, nêu ưu nhược điểm và khi nào nên sử dụng quy trình đó.

a) Phân biệt:

Quy trình hoạch định sẵn	Quy trình linh hoạt
Tất cả các hoạt động được lên kế hoạch trước và tiến độ thực hiện được đánh giá dựa vào kế hoạch này	Kế hoạch được phát triển dần dần và dễ dàng thay đổi quy trình để đáp ứng sự thay đổi yêu cầu của khách hàng
Các pha đặc tả và phát triển phân biệt và tách rời nhau.	Các pha đặc tả, phát triển và thẩm định đan xen nhau.
Quy trình rõ ràng	Quy trình không rõ ràng
Không linh động, khó khăn trong việc thích nghi với sự thay đổi sau khi quy trình đã vào guồng.	Giảm được chi phí khi đáp ứng sự thay đổi yêu cầu của khách hàng, dễ dàng lấy phản hồi của khách hàng
Khách hàng chỉ được phân phối sản phẩm khi phần mềm khi các pha được thực hiện xong	Phân phối phiên bản mới thường xuyên để đánh giá và có sự tham gia của stakeholder.
Đầu ra ở mỗi giai đoạn đã được lên kế hoạch trước	Đầu ra từ quy trình phát triển được quyết định thông qua việc thương lượng trong suốt quá trình phát triển phần mềm.

b) Waterfall:

- Mô tả: Bao gồm các giai đoạn: xác định yêu cầu, thiết kế và phân tích, cài đặt, kiểm thử, triển khai, bảo trì và vận hành được thực hiện tuần tự. Đảm bảo hoàn thành một giai đoạn trước khi thực hiện giai đoạn tiếp theo
- Ưu
  - Quy trình rõ ràng -> quản lí dễ theo dõi tiến độ công việc
  - Giúp cho việc phối hợp trong công việc dễ dàng hơn
- Nhược
  - Khó khăn trong việc thích nghi với sự thay đổi sau khi quy trình đã vào guồng
  - Không linh động trong việc phân chia dự án thành những giai đoạn tách biệt -> khó khăn trong việc đáp ứng sự thay đổi yêu cầu người dùng
- Khi nào nên dùng
  - Mô hình này chỉ hợp lý khi yêu cầu được hiểu rõ và ít thay đổi trong suốt quá trình phát triển;
  - Hệ thống thương mại thường có yêu cầu không ổn định -> mô hình thác nước không phù hợp.
  - Mô hình này được sử dụng trong các hệ thống lớn trong đó hệ thống được phát triển tại nhiều địa điểm khác nhau

Câu 7: Mục đích của kiểm thử phần mềm(software testing) là gì? Hãy liệt kê và **mô tả** các pha trong quy trình kiểm thử phần mềm.

Mục tiêu:

- Chỉ ra rằng một chương trình thực hiện đúng như mong đợi và

- Tìm ra được lỗi của chương trình trước khi đưa vào sử dụng.
- Có thể chỉ ra sự có mặt của lỗi, không chỉ ra được chương trình không có lỗi.

Các pha:

1. Requirement analysis - Phân tích yêu cầu
2. Test planning - Lập kế hoạch kiểm thử
3. Test case development - Thiết kế kịch bản kiểm thử
4. Test environment set up - Thiết lập môi trường kiểm thử
5. Test execution - Thực hiện kiểm thử
6. Test cycle closure - Đóng chu trình kiểm thử

1/ Phân tích yêu cầu

Đọc hiểu, nghiên cứu, phân tích cái tài liệu đặc tả, từ đó nắm yêu cầu mà dự án đưa ra bao gồm kiểm thử chức năng/phi chức năng nào.

2/ Lập kế hoạch kiểm thử

Xác định các yếu tố quan trọng sau:

- Phạm vi dự án: dự án thực hiện trong bao lâu, bao gồm những công việc gì, từ đó đưa ra lịch trình phù hợp
- Phương pháp tiếp cận
- Nguồn lực
- Lên kế hoạch thiết kế công việc test:
    + Liệt kê các chức năng cần kiểm thử.
    + Để thực hiện test chức năng này thì cần làm những công việc gì, trong thời gian bao lâu, cái nào thực hiện trước, cái nào thực hiện sau, ai là người thực hiện.
    + Xác định điều kiện bắt đầu: xác định những điều kiện tối thiểu để bắt đầu hoạt động kiểm thử cho từng chức năng.
    + Xác định điều kiện kết thúc : khi có những điều kiện nào thì sẽ kết thúc việc kiểm thử.

3/ Thiết kế kịch bản kiểm thử

- Xem lại các tài liệu để xác nhận lại các chức năng nào cần test.
- Viết test case cho tất cả các trường hợp có thể xảy ra
- Chuẩn bị dữ liệu kiểm thử
- Review test case

4/ Thiết lập môi trường kiểm thử

Chuẩn bị 1 vài test case xem thử môi trường cài đặt có sẵn sàng cho việc kiểm thử chưa

5/ Thực hiện kiểm thử

Thực hiện các test case như thiết kế và mức độ ưu tiên đã đưa ra trên môi trường đã được cài đặt.

6/ Đóng chu trình kiểm thử

Báo cáo kết quả về việc thực thi test case, đánh giá các tiêu chí hoàn thành như phạm vi kiểm tra, chất lượng, chi phí, thời gian, mục tiêu kinh doanh quan trọng.

Câu 8: Hãy trình bày các hoạt động ở giai đoạn thu thập yêu cầu trong một dự án phần mềm. Làm cách nào để thu thập yêu cầu có chất lượng đảm bảo tính chính xác, đầy đủ, và nhất quán?

- Các hoạt động:

- Nhận diện yêu cầu: Tương tác với các stakeholder để tìm ra các yêu cầu của họ.
- Tổ chức và phân loại yêu cầu: Phân tích các yêu cầu có liên quan với nhau và tổ chức chúng thành các nhóm.
- Đặc độ ưu tiên và thương lượng: Đặt độ ưu tiên cho yêu cầu và giải quyết xung đột

- Đặc tả yêu cầu: Yêu cầu được viết thành tài liệu
- Cách thu thập: Ta cần áp dụng các kỹ thuật thu thập yêu cầu như phỏng vấn, quan sát, lấy mô tả câu chuyện người dùng, làm bản mẫu, mô hình hoá yêu cầu. Sau các bước phân tích, ta cần duyệt lại với stakeholder để thẩm định yêu cầu

Câu 9: Hãy mô tả mô hình V-model trong quá trình phát triển phần mềm. Phân biệt khác nhau giữa kiểm tra động(dynamic verification) và kiểm tra tĩnh(static verification)

a) Mô hình V-Model là mô hình phát triển mở rộng từ mô hình Waterfall. Các bài kiểm tra và kiểm thử được phát triển đồng thời và song song với từng giai đoạn phát triển. Điều này đảm bảo rằng từng bước phát triển có một bài kiểm tra hoặc kiểm thử tương ứng, giúp đảm bảo tích hợp và kiểm tra hiệu quả. Mô hình V-Model thích hợp cho các dự án đòi hỏi tính chính xác và đáng tin cậy trong kiểm thử. Nó tạo mối liên kết mạnh mẽ giữa các giai đoạn phát triển và kiểm thử, giúp đảm bảo rằng dự án được kiểm tra toàn diện và chất lượng sản phẩm đạt được.

b)

static verification	dynamic verification
Liên quan đến việc phân tích các biểu diễn tĩnh của hệ thống để tìm ra lỗi.	Liên quan đến việc thực hiện và quan sát hành vi của sản phẩm với dữ liệu kiểm thử
Không yêu cầu chạy chương trình, có thể được áp dụng cho các hoạt động trước khi cài đặt	Cần phải có chương trình để chạy
Không cần quan tâm đến các tương tác của lỗi	Một lỗi có thể bị che giấu bởi các lỗi khác

Câu 10: Hãy trình bày những nguyên tắc cơ bản khi thiết kế giao diện người dùng. Hãy liệt kê những cách thức thu thập yêu cầu liên quan đến giao diện và trải nghiệm người dùng. Làm thể nào kiểm tra những yêu cầu thu thập được là phù hợp với mong đợi của khách hàng.

a) Các nguyên tắc cơ bản khi thiết kế giao diện người dùng:

- **Thân thiện với người dùng**: Sử dụng các thuật ngữ và khái niệm hướng người dùng.
- **Nhất quán**: Hệ thống nên hiển thị một cách nhất quán.
- **Ít bất ngờ:** Nếu một lệnh được thực hiện theo cách thông thường, người dùng có thể dự đoán được thao tác của các lệnh tương tự.
- **Có thể khôi phục được:** Khi gặp lỗi: Hệ thống nên cung cấp một số cơ chế khôi phục tình trạng hoạt động bình thường trước khi gặp lỗi.
- **Hướng dẫn người dùng:** Cung cấp một số hướng dẫn người dùng
- **Đa dạng người dùng:** Cung cấp tiện ích tương tác cho các loại người dùng khác nhau.

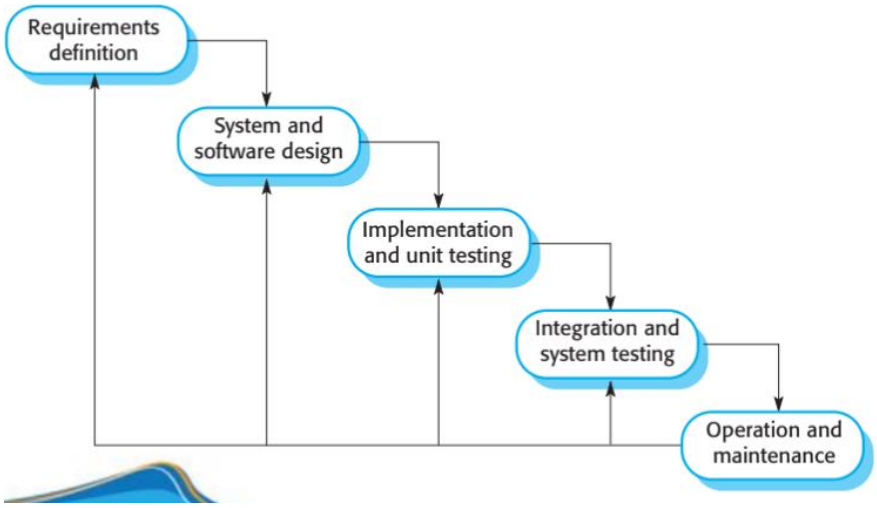
b) Các cách thức để thu thập yêu cầu liên quan đến giao diện và trải nghiệm người dùng:

- Phỏng vấn
- Stories và scenarios: Lấy mô tả về cách mà hệ thống có thể được sử dụng cho một tác vụ cụ thể.

c) Làm thể nào kiểm tra những yêu cầu thu thập được là phù hợp với mong đợi của khách hàng

- Phân tích yêu cầu và mô hình hoá bằng use-case, sơ đồ trình tự
- Tạo bản mẫu để thẩm định yêu cầu
- Lấy ý kiến đánh giá cho những cái trên từ khách hàng thông qua quá trình duyệt yêu cầu để có thể có thể chỉnh sửa cho phù hợp với mong đợi của khách hàng

Câu 11:
a) Hãy vẽ mô hình minh hoạ quy trình phát triển phần mềm thác nước (Waterfall).
b) Hãy mô tả các vai trò tham gia cùng các hoạt động diễn ra trong từng pha của quy trình phát triển phần mềm thác nước.
a)



b)
Phân tích yêu cầu:

- PM: lên kế hoạch đề xuất, kế hoạch triển khai để thoả thuận với khách hàng
- BA: làm việc với các stakeholder để thu thập, phân tích và thẩm định yêu cầu

Thiết kế:

- BA: đảm bảo yêu cầu được hiểu chính xác bởi các Designer
- Designer: từ yêu cầu thiết kế các chức năng cụ thể, thiết kế kiến trúc hệ thống, thiết kế chi tiết các component cần thiết, thiết kế giao diện

Cài đặt:

- Developer: viết mã nguồn, tích hợp các component, cấu hình hệ thống theo thiết kế, kiểm thử đơn vị

Kiểm thử:

- Tester: thực hiện kiểm định và thẩm định các chức năng, hệ thống của phần mềm, báo cáo lỗi cho dev.
- Developer: thực hiện tìm và sửa tất cả lỗi được báo trước khi đưa vào triển khai

Triển khai và bảo trì:

- Devloper: viết các tài liệu triển khai và cấu hình, tạo ra các phiên bảo release, tạo các bản vá chữa lỗi phát sinh khi sử dụng
- PM: đảm bảo việc triển khai diễn ra đúng thời hạn và xử lí các vấn đề khó khăn phát sinh.

Câu 12:
a) Hãy trình bày ưu khuyết điểm của các phương pháp phát triển phần mềm truyền thống so với các phương pháp phát triển nhanh (Agile).
b) Hãy trình bày những nguyên nhân khiến cho các phương pháp phát triển phần mềm truyền thống dễ dẫn đến thất bại hơn so với các phương pháp phát triển nhanh.

a) .
    - Ưu điểm
        - Quy trình rõ ràng, kế hoạch dự án được lên kế hoạch từ trước giúp quản lí tiến độ được tốt hơn
        - Các giai đoạn tách biệt nhau nên có thể phối hợp một cách dễ dàng hơn và phát triển ở nhiều vị trí địa lí khác nhau
    - Khuyết điểm
        - Không phù hợp với các dự án có yêu cầu không ổn định, quy trình không thích nghi được với nhiều sự thay đổi khi đã vào guồng
        - Không linh động trong việc phân chia các giai đoạn nên không đáp ứng được những thay đổi của của khách hàng
b) Nguyên nhân:
    - Chi phí bỏ ra là quá cao cho việc áp dụng các thay đổi của khách hàng
    - Tốn nhiều thời gian để lên kế hoạch và hoạch định sẵn tất cả những gì cần làm
    - Khách hàng chỉ được tiếp xúc với phần mềm khi pha cuối cùng kết thúc nên có thể không đáp ứng được nhu cầu của khách hàng
    - Các rủi ro và các lỗi không được kiểm tra liên tục nên dễ tốn chi phí để giải quyết các vấn đề phát sinh

CÂU HỎI ĐÁNH GIÁ QUỐC DÂN: Hãy liệt kê tên và các công việc chính của từng thành viên **khác** trong nhóm đồ án của bạn. Hãy mô tả cách thức làm việc, giao tiếp giữa các thành viên trong nhóm trong quá trình thực hiện đồ án

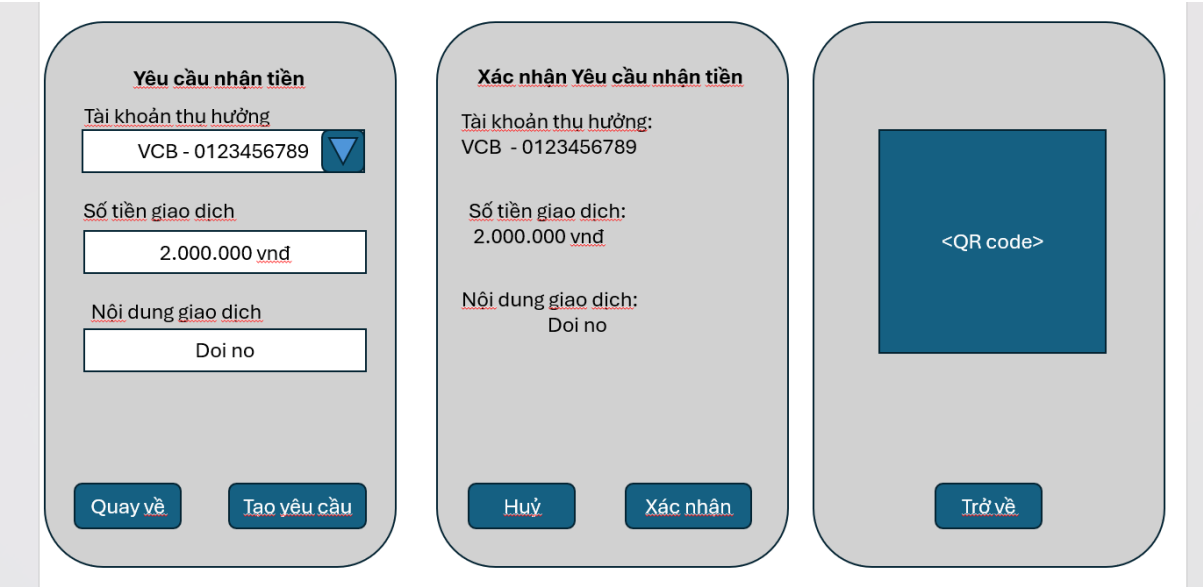
a) Công việc của các thành viên (bỏ phần của mình ra)

- Lưu Chấn Huy
  - Phát biểu bài toán của dự án
  - Danh sách các stakeholder
  - Đặc tả cơ sở dữ liệu
  - Phát triển tầng Business của ứng dụng
  - Viết test case và kiểm thử cho 1 chức năng
- Bùi Hoàng Duy
  - Để xuất kiến trúc tổng thể phần cứng, phần mềm
  - Lập danh sách các yêu cầu
  - Thiết kế mô hình quan niệm và sơ đồ dữ liệu
  - Phát triển xây dựng cơ sở dữ liệu và dữ liệu mẫu
  - Viết test case và kiểm thử cho 1 chức năng
- Lê Quang Khánh
  - Lập kế hoạch phát triển
  - Sơ đồ Use Case và đặc tả
  - Thiết kế kiến trúc và sơ đồ lớp
  - Phát triển tầng DAL và xây dựng DTO cho ứng dụng
  - Lập Test Plan, viết test case và kiểm thử cho 1 chức năng
- Biện Công Khanh
  - Lập kế hoạch phát triển
  - Làm bản mẫu cho giao diện
  - Thiết kế giao diện
  - Phát triển xây dựng giao diện
  - Viết test case và kiểm thử cho 1 chức năng
- Ngô Thành Nhân
  - Lập kế hoạch nhân sự và chi phí
  - Sơ đồ Use Case và đặc tả
  - Thiết kế kiến trúc và sơ đồ lớp
  - Phát triển tầng Business của ứng dụng
  - Viết test case và kiểm thử cho 1 chức năng

b) Cách thức làm việc và giao tiếp: chia đồ án thành từng giai đoạn của quy trình Waterfall, nhận phân công công việc ở mỗi đầu giai đoạn trên Google Sheet, sử dụng Discord để giao tiếp và họp mặt online, dùng github để lưu trữ và chia sẻ mã nguồn giữa các thành viên.

Ôn phần bài tập
Câu1c

Use case ID	**U001**
Tên Use Case	Tạo một yêu cầu nhận tiền
Tóm tắt	Người dùng khởi tạo giao dịch nhận tiền từ người khác
Tác nhân	Người dùng hệ thống
Điều kiện tiên quyết	Người dùng đã đăng nhập, tài khoản liên kết với ít nhất một tài khoản ngân hàng
Kết quả	Ứng dụng hiển thị mã QR để người nhận đưa cho người gửi
Kịch bản chính	1. Người dùng chọn chức năng Yêu cầu nhận tiền 2. Hệ thống hiển thị các ô thông tin 3. Người dùng chọn ngân hàng và điền đầy đủ thông tin 4. Người dùng nhấn nút Tạo yêu cầu 5. Hệ thống kiểm tra thông tin hợp lệ, hiển thị lại toàn bộ thông tin đã điền 6. Người dùng ấn nút xác nhận để tạo yêu cầu 7. Hệ thống hiển thị mã QR đã khởi tạo
Kịch bản phụ	3b. Người dùng ấn nút quay về, hệ thống quay lại màn hình chính 5b. Thông tin không hợp lệ, hệ thống báo lỗi và quay lại bước 5 6b. Người dùng nhấn hủy bỏ, quay lại bước 3
Ràng buộc phi chức năng	Tốc độ tạo QR nhanh chóng(tối đa 5s) Số tiền không được vượt quá 20000000



CâuII.3

Test case	**Tạo yêu cầu nhận tiền với thông tin hợp lệ**
Mô tả	Người dùng nhập đầy đủ thông tin để tạo thành công yêu cầu
Input Data	1. Tài khoản thụ hưởng: “VCB - 0123456789” 2. Số tiền: “2000000” 3. Nội dung: “Doi no”
Expected Output	Xuất hiện mã QR trên màn hình
Test steps	1. Chọn tài khoản thụ hưởng từ Combobox 2. Nhập thông tin về số tiền và nội dung giao dịch 3. Chọn nút tạo yêu cầu 4. Chọn nút xác nhận

Test case	**Tạo yêu cầu nhận tiền với thông tin chưa nhập đủ**
Mô tả	Người dùng nhập không đủ thông tin để tạo thành công yêu cầu
Input Data	1. Tài khoản thụ hưởng: “VCB - 0123456789” 2. Số tiền: “” 3. Nội dung: “Doi no”
Expected Output	Xuất hiện thông báo thiếu thông tin trên màn hình
Test steps	1. Chọn tài khoản thụ hưởng từ Combobox 2. Nhập thông tin về số tiền và nội dung giao dịch 3. Chọn nút tạo yêu cầu

Test case	**Tạo yêu cầu nhận tiền với thông tin số tiền không hợp lệ(số liền quá lớn)**
Mô tả	Người dùng nhập thông tin số tiền quá lớn vượt mức cho phép
Input Data	4. Tài khoản thụ hưởng: “VCB - 0123456789” 5. Số tiền: “200.000.000” 6. Nội dung: “Doi no”

Expected Output	Xuất hiện thông báo không được giao dịch quá 20000000 trên màn hình
Test steps	4. Chọn tài khoản thụ hưởng từ Combobox 5. Nhập thông tin về số tiền và nội dung giao dịch 6. Chọn nút tạo yêu cầu

Test case	**Tạo yêu cầu nhận tiền và không xác nhận thông tin**
Mô tả	Người dùng nhập đầy đủ thông tin hợp lệ nhưng không xác nhận thông tin
Input Data	4. Tài khoản thụ hưởng: "VCB - 0123456789" 5. Số tiền: "2000000" 6. Nội dung: "Doi no"
Expected Output	Ứng dụng quay lại màn hình nhập thông tin
Test steps	5. Chọn tài khoản thụ hưởng từ Combobox 6. Nhập thông tin về số tiền và nội dung giao dịch 7. Chọn nút tạo yêu cầu 8. Chọn nút huỷ

Câu II.1:

a) Các mong muốn:

- Là người tổ chức, tôi muốn tạo cuộc họp: tạo mã và mật khẩu cho phòng hop
- Là người tổ chức cuộc họp, tôi mong muốn ứng dụng cho phép tôi quản lí người tham gia: tắt mic, mời ra khỏi phòng, mời vào phòng
- Là người tổ chức, tôi muốn chia phòng họp thành các nhóm họp riêng
- Là người tham gia, tôi muốn bật mic để phát biểu ý kiến
- Là người tham gia, tôi muốn giao tiếp với mọi người qua kênh chat
- Là người tham gia, tôi muốn dùng filter và chỉnh camera quay hình của mình
- Là người tham gia, tôi có thể tham gia phòng họp thông qua mã và mật khẩu phòng họp
- Là người tham gia, tôi muốn chia sẻ file cho các thành viên trong phòng họp

Phi chức năng

- Là người tổ chức, tôi muốn phòng họp chứa được ít nhất 200 người cùng lúc
- Là người tham gia, tôi muốn tốc độ tin nhắn được gửi tới người khác phải nhanh (<2s)
- Là người tổ chức, tôi muốn nội dung của cuộc họp phải được bảo mật.

b) Use case:

- Người tổ chức
  - Tổ chức cuộc họp
  - Quản lí thành viên tham gia
  - **Chat**
  - **bật tắt mic**
  - **chỉnh camera**
  - **Chia sẻ file**
  - Chia phòng họp
- Người tham gia
  - **Chat**
  - **bật tắt mic**
  - **chỉnh camera**
  - **Chia sẻ file**
  - Tham gia cuộc họp bằng mã và mật khẩu

c)

Use case ID	**U001**
Tên Use Case	Tổ chức cuộc họp
Tóm tắt	Người dùng tạo cuộc họp cho người khác tham gia
Tác nhân	Người tổ chức
Điều kiện tiên quyết	Người dùng đã tạo tài khoản
Kết quả	Ứng dùng chuyển sang giao diện cuộc họp, cung cấp mã và mật khẩu
Kịch bản chính	1. Người dùng chọn nút tạo cuộc họp 2. Hệ thống hiển thị mã và ô để người dùng nhập mật theo ý muốn 3. Người dùng nhập mật khẩu 4. Người dùng nhấn nút tạo

	5. Hệ thống chuyển sang giao diện cuộc họp
Kịch bản phụ	4b. Người dùng nhấn huỷ để quay lại bước 1
Ràng buộc phi chức năng	Tốc độ tạo cuộc họp nhanh (<5s)