CHƯƠNG 1

Trắc nghiệm:

1. Câu hỏi 1: Phần mềm bao gồm các loại nào dưới đây?

A. Phần mềm hệ thống

B. Phần mềm ứng dụng

C. Phần mềm nhúng

D. Cả A, B và C

2. Câu hỏi 2: Công nghệ phần mềm là gì?

A. Việc viết mã nguồn cho phần mềm

B. Phát triển phần mềm mà không có lỗi

C. Ứng dụng các phương pháp khoa học để phát triển phần mềm

D. Chỉ bảo trì phần mềm

3. Câu hỏi 3: Quy trình phát triển phần mềm gồm mấy giai đoạn chính?

A. 3

B. 4

C. 5

D. 6

4. Câu hỏi 4: Hoạt động nào dưới đây thuộc quy trình bảo trì phần mềm?

A. Lập kế hoạch

B. Triển khai phần mềm

C. Cập nhật phần mềm để phù hợp với thay đổi môi trường

D. Phân tích yêu cầu

5. Câu hỏi 5: Chi phí bảo trì phần mềm chiếm bao nhiêu phần trăm tổng chi phí vòng đời phần mềm?

A. 10%

B. 30%

C. 60%

D. 90

6. Câu hỏi 6: Nguyên nhân chính gây ra việc vượt chi phí khi phát triển phần mềm là gì?

A. Thiếu nhân lực

B. Không xác định rõ yêu cầu

C. Thay đổi công nghệ

D. Cả A và C

7. Câu hỏi 7: Yêu cầu nào dưới đây không phải là yêu cầu phi chức năng?

A. Hiệu suất xử lý

B. Tính bảo mật

C. Khả năng mở rộng

D. Chức năng đăng nhập

8. Câu hỏi 8: Khi nào phần mềm được coi là hoàn thành?

A. Khi hoàn thành việc viết mã nguồn

B. Khi được bàn giao cho khách hàng và không còn lỗi

C. Khi được triển khai trên hệ thống của khách hàng

D. Khi được khách hàng chấp nhận và đưa vào sử dụng

9. Câu hỏi 9: Vấn đề phổ biến nào thường gặp khi phát triển phần mềm?

A. Thiếu công cụ hỗ trợ

B. Vượt chi phí, trễ thời hạn và lỗi sau khi bàn giao

C. Không có đội kiểm thử

D. Tất cả đều đúng

10. Câu hỏi 10: Phần mềm có thể được chia thành bao nhiêu loại chính?

A. 2

B. 3

C. 4

D. 5

CÂU HỎI NGẮN

1. Phần mềm là gì ?

- Phần mềm là tập hợp các hướng dẫn, chương trình được viết để máy tính thực thi nhằm thực hiện các chức năng hoặc nhiệm vụ cụ thể.

2. Công nghệ phần mềm là gì?

- Công nghệ phần mềm là lĩnh vực nghiên cứu, phát triển và áp dụng các phương pháp có hệ thống để xây dựng phần mềm chất lượng cao.

3. Các loại phần mềm chính là gì?

- Phần mềm hệ thống: Hệ điều hành, trình điều khiển thiết bị.
- Phần mềm ứng dụng: Các chương trình phục vụ công việc hoặc nhu cầu của người dùng cuối như Microsoft Office, trình duyệt web.
- Phần mềm nhúng: Phần mềm điều khiển các thiết bị phần cứng như máy giặt, điều hòa.

4.Tại sao công nghệ phần mềm lại quan trọng?

- Tăng cường hiệu suất: Giúp tự động hóa quy trình, giảm thiểu lỗi và tiết kiệm thời gian.
- Khả năng mở rộng: Dễ dàng mở rộng và thích ứng với nhu cầu thay đổi của doanh nghiệp.
- Kết nối: Tạo ra các ứng dụng và nền tảng kết nối con người và thông tin.
- Đổi mới: Khuyến khích sự sáng tạo và phát triển sản phẩm mới.
- Cạnh tranh: Giúp doanh nghiệp duy trì lợi thế cạnh tranh trong thị trường.

5. Quy trình phát triển phần mềm gồm những giai đoạn nào?

- Lấy yêu cầu: Thu thập và phân tích các yêu cầu từ khách hàng.
- Thiết kế: Lên kế hoạch và cấu trúc hệ thống phần mềm.
- Lập trình: Chuyển đổi thiết kế thành mã nguồn thực thi.
- Kiểm thử: Đảm bảo phần mềm hoạt động đúng chức năng.
- Triển khai: Cài đặt và bàn giao phần mềm cho khách hàng.
- Bảo trì: Khắc phục lỗi và nâng cấp phần mềm sau khi triển khai.

6. Khía cạnh kinh tế của công nghệ phần mềm là gì?

- Phần mềm là yếu tố cốt lõi trong nhiều ngành công nghiệp như tài chính, y tế, giáo dục.
- Sự phát triển của phần mềm giúp tăng năng suất lao động và tối ưu hóa chi phí vận hành doanh nghiệp.
- Các dự án phần mềm lớn thường có chi phí rất cao, do đó việc áp dụng công nghệ phần mềm giúp kiểm soát chi phí tốt hơn.

7. Khía cạnh công nghệ của công nghệ phần mềm là gì?

- Sự phát triển nhanh chóng của công nghệ yêu cầu phần mềm phải được cải tiến liên tục để đáp ứng nhu cầu mới.
- Phần mềm giúp kết nối các hệ thống phức tạp và xử lý khối lượng dữ liệu lớn.

8. Khía cạnh bảo trì của công nghệ phần mềm là gì?

- Sau khi phần mềm được triển khai, bảo trì là hoạt động cần thiết để đảm bảo phần mềm hoạt động ổn định và đáp ứng các yêu cầu thay đổi của người dùng.
- Bảo trì phần mềm chiếm khoảng 60% tổng chi phí vòng đời của một hệ thống phần mềm.

9. Các nguyên nhân chính gây trễ thời hạn khi phát triển phần mềm là gì?

- Yêu cầu thay đổi liên tục từ phía khách hàng.
- Thiếu nhân lực hoặc sự phối hợp kém giữa các thành viên trong nhóm phát triển.

10. Bảo trì phần mềm bao gồm những hoạt động nào?

- Khắc phục lỗi và nâng cấp phần mềm sau khi triển khai.

THẢO LUẬN NHÓM

Câu 1: Phân biệt phần mềm hệ thống và phần mềm ứng dụng.

Tiêu chí	Phần mềm hệ thống	Phần mềm ứng dụng
Khái niệm	Phần mềm hỗ trợ quản lý và điều hành phần cứng máy tính, tạo môi trường cho các phần mềm khác hoạt động.	Phần mềm được thiết kế để thực hiện các công việc cụ thể phục vụ nhu cầu của người dùng.
Mục đích	Điều khiển, quản lý tài nguyên hệ thống, hỗ trợ phần mềm khác chạy.	Cung cấp chức năng đáp ứng nhu cầu sử dụng của người dùng.
Ví dụ	Hệ điều hành (Windows, Linux, macOS), trình điều khiển (Drivers), BIOS, phần mềm tiện ích hệ thống.	Microsoft Office, trình duyệt web (Chrome, Firefox), phần mềm đồ họa (Photoshop), ứng dụng di động (Facebook, Zalo).
Cách hoạt động	Chạy nền, hoạt động liên tục để đảm bảo máy tính vận hành ổn định.	Chỉ chạy khi người dùng khởi động và sử dụng.
Mức độ tương tác với người dùng	Ít tương tác trực tiếp, chủ yếu hoạt động phía sau để hỗ trợ hệ thống.	Có giao diện thân thiện, người dùng thao tác trực tiếp.
Tính cần thiết	Bắt buộc có để máy tính hoạt động.	Không bắt buộc, tùy vào nhu cầu sử dụng của người dùng.

Câu 2: Thảo luận về vai trò của công nghệ phần mềm trong lĩnh vực tài chính.

1. Tự động hóa giao dịch tài chính

- Các hệ thống phần mềm cho phép thực hiện giao dịch tự động như chứng khoán, ngoại hối, tiền điện tử mà không cần can thiệp thủ công.
- Thuật toán giao dịch (Algorithmic Trading) giúp phân tích dữ liệu nhanh chóng và đưa ra quyết định mua/bán trong mili-giây.

Ví dụ: Hệ thống giao dịch thuật toán của Bloomberg, MetaTrader 4/5.

---

2. Quản lý tài chính cá nhân và doanh nghiệp

- Phần mềm giúp cá nhân và doanh nghiệp quản lý thu nhập, chi tiêu, đầu tư một cách hiệu quả.
- Hỗ trợ lập kế hoạch tài chính, dự báo xu hướng và đưa ra quyết định đầu tư tốt hơn.

Ví dụ: MISA, QuickBooks, Mint, YNAB (You Need A Budget).

---

3. Ngân hàng số và Fintech

- Công nghệ phần mềm giúp ngân hàng chuyển đổi số, cung cấp dịch vụ ngân hàng trực tuyến, ví điện tử, thanh toán điện tử nhanh chóng và tiện lợi.
- Ứng dụng AI & Big Data giúp cá nhân hóa dịch vụ, hỗ trợ khách hàng tốt hơn.

Ví dụ: Mobile Banking (Vietcombank, BIDV), ví điện tử (MoMo, ZaloPay, PayPal).

---

4. Bảo mật và phòng chống gian lận

- Công nghệ mã hóa, bảo mật sinh trắc học (vân tay, nhận diện khuôn mặt) bảo vệ thông tin tài chính.
- AI và Machine Learning được ứng dụng để phát hiện gian lận trong giao dịch tài chính.

Ví dụ: Hệ thống phát hiện gian lận của Mastercard, Visa.

---

5. Phân tích dữ liệu tài chính và dự báo thị trường

- Các phần mềm phân tích tài chính giúp đánh giá rủi ro, tối ưu hóa danh mục đầu tư và dự báo xu hướng thị trường.
- Công nghệ Big Data và AI giúp xử lý lượng dữ liệu khổng lồ để đưa ra quyết định chính xác hơn.

Ví dụ: Bloomberg Terminal, Reuters Eikon.

---

6. Hỗ trợ tuân thủ quy định tài chính (RegTech)

- Công nghệ giúp các tổ chức tài chính tuân thủ quy định pháp luật, giám sát giao dịch, giảm rủi ro bị phạt do vi phạm quy định.
- Blockchain giúp minh bạch hóa giao dịch và giảm gian lận.

Ví dụ: Hệ thống KYC (Know Your Customer), AML (Anti-Money Laundering).

Câu 3: Nêu các thách thức thường gặp trong bảo trì phần mềm.

Các thách thức là:

1. **Hiểu mã nguồn phức tạp**:
    - Khi phần mềm phát triển qua nhiều năm, mã nguồn có thể trở nên phức tạp và khó hiểu, đặc biệt nếu thiếu tài liệu hoặc người phát triển ban đầu không còn tham gia.
2. **Thiếu tài liệu**:
    - Tài liệu không đầy đủ hoặc lỗi thời khiến việc hiểu và sửa đổi mã nguồn trở nên khó khăn.
3. **Quản lý phiên bản**:
    - Theo dõi và quản lý các phiên bản khác nhau của phần mềm, đảm bảo rằng các thay đổi không gây ra xung đột hoặc lỗi.
4. **Tương thích**:
    - Đảm bảo phần mềm hoạt động tốt trên các nền tảng, hệ điều hành và thiết bị khác nhau, đặc biệt khi có cập nhật từ nhà cung cấp.
5. **Bảo mật**:
    - Liên tục cập nhật và vá lỗ hổng bảo mật để bảo vệ phần mềm khỏi các mối đe dọa mới.
6. **Chi phí và thời gian**:
    - Bảo trì thường tốn kém và mất nhiều thời gian, đặc biệt khi cần sửa chữa lớn hoặc tối ưu hóa.
7. **Yêu cầu thay đổi**:
    - Yêu cầu người dùng thay đổi theo thời gian, đòi hỏi phần mềm phải được cập nhật liên tục, gây áp lực lên đội phát triển.
8. **Kiểm thử**:
    - Đảm bảo các thay đổi không gây ra lỗi mới đòi hỏi quy trình kiểm thử nghiêm ngặt, tốn thời gian và công sức.
9. **Quản lý rủi ro**:
    - Mọi thay đổi đều tiềm ẩn rủi ro, cần được quản lý cẩn thận để tránh ảnh hưởng đến hệ thống.
10. **Kỹ năng và kiến thức**:
    - Đội ngũ bảo trì cần có kỹ năng và kiến thức cập nhật để xử lý các vấn đề phức tạp và công nghệ mới.

11. **Tích hợp hệ thống**:
    - Khi phần mềm cần tích hợp với hệ thống khác, việc đảm bảo tương thích và hoạt động trơn tru có thể là thách thức lớn.
12. **Quản lý lỗi**:
    - Theo dõi, ưu tiên và khắc phục lỗi từ người dùng cuối đòi hỏi quy trình quản lý hiệu quả.

Câu 4: Vì sao phần mềm thương mại điện tử cần được bảo trì thường xuyên?

Vì:

1. **Đảm bảo tính bảo mật**:
    - Các trang thương mại điện tử xử lý thông tin nhạy cảm như thông tin thanh toán và cá nhân của khách hàng. Bảo trì thường xuyên giúp vá lỗ hổng bảo mật và bảo vệ dữ liệu khỏi các mối đe dọa như tấn công mạng và phần mềm độc hại.
2. **Cải thiện trải nghiệm người dùng**:
    - Người dùng mong đợi trải nghiệm mượt mà và thân thiện. Bảo trì giúp tối ưu hóa giao diện, sửa lỗi và cải thiện hiệu suất, từ đó nâng cao sự hài lòng của khách hàng.
3. **Cập nhật công nghệ mới**:
    - Công nghệ phát triển nhanh chóng, và việc cập nhật các công nghệ mới giúp phần mềm hoạt động hiệu quả hơn, tận dụng các tính năng và công cụ mới nhất.
4. **Duy trì tính tương thích**:
    - Phần mềm cần tương thích với các hệ điều hành, trình duyệt và thiết bị mới. Bảo trì thường xuyên đảm bảo phần mềm hoạt động tốt trên mọi nền tảng.
5. **Xử lý lỗi và sự cố**:
    - Lỗi và sự cố có thể xảy ra bất cứ lúc nào, gây ảnh hưởng đến hoạt động kinh doanh. Bảo trì giúp phát hiện và khắc phục sớm các vấn đề này.
6. **Đáp ứng yêu cầu kinh doanh thay đổi**:
    - Yêu cầu kinh doanh và thị trường thay đổi liên tục. Bảo trì giúp cập nhật và điều chỉnh phần mềm để đáp ứng các yêu cầu mới, như thêm tính năng mới hoặc thay đổi quy trình.
7. **Tối ưu hóa hiệu suất**:
    - Theo thời gian, phần mềm có thể chậm lại hoặc gặp vấn đề về hiệu suất. Bảo trì giúp tối ưu hóa mã nguồn và cải thiện tốc độ xử lý.
8. **Tuân thủ quy định**:
    - Các quy định về bảo mật và quyền riêng tư thay đổi thường xuyên. Bảo trì giúp đảm bảo phần mềm tuân thủ các quy định mới nhất, tránh rủi ro pháp lý.
9. **Duy trì tính cạnh tranh**:
    - Thị trường thương mại điện tử cạnh tranh khốc liệt. Bảo trì thường xuyên giúp cải thiện và cập nhật phần mềm, giữ vững lợi thế cạnh tranh.

10. **Quản lý dữ liệu hiệu quả**:
    - Dữ liệu khách hàng và sản phẩm tăng lên theo thời gian. Bảo trì giúp quản lý và lưu trữ dữ liệu hiệu quả, đảm bảo hệ thống hoạt động ổn định.

5. Phân tích những vấn đề khi yêu cầu khách hàng liên tục thay đổi trong quá trình phát triển phần mềm.

Những vấn đề khi yêu cầu khách hàng liên tục thay đổi trong quá trình phát triển phần mềm:

- Tăng chi phí (mỗi lần thay đổi đều cần thời gian và nguồn lực để phát triển, dẫn đến việc gia tăng chi phí phát triển)
- Trễ tiến độ
- Giảm chất lượng sản phẩm
- Khó khăn trong giao tiếp
- Mất định hướng, khó khăn trong quản lí dự án

6. So sánh chi phí phát triển và chi phí bảo trì phần mềm.

Chi phí phát triển phần mềm: là chi phí để thiết kế, triển khai và phát triển ban đầu. Bao gồm:

- Lương nhân viên
- Công cụ và công nghệ
- Thời gian

Chi phí bảo trì phần mềm: là chi phí phát sinh sau khi phần mềm đã được triển khai, bao gồm bảo trì, cập nhật và sửa lỗi. Bao gồm :

- Bảo trì kỹ thuật
- Cải tiến và nâng cấp
- Hỗ trợ người dùng

7. Phân biệt các loại yêu cầu trong phát triển phần mềm (chức năng và phi chức năng).

Yêu cầu chức năng

- Định nghĩa: Yêu cầu chức năng mô tả các hành vi, chức năng và tính năng mà phần mềm cần phải thực hiện để đáp ứng nhu cầu của người dùng.

- Tập trung vào "cái gì": Xác định những gì hệ thống phải làm, tức là các chức năng mà người dùng có thể thực hiện.

Ví dụ:

- Đăng nhập: Hệ thống phải cho phép người dùng đăng nhập bằng tên người dùng và mật khẩu.
- Quản lý đơn hàng: Hệ thống phải cho phép người dùng tạo, xem, sửa và xóa đơn hàng.
- Tìm kiếm sản phẩm: Hệ thống phải cung cấp chức năng tìm kiếm sản phẩm theo tên, loại, hoặc giá.

Yêu cầu phi chức năng

- Định nghĩa: Yêu cầu phi chức năng mô tả các thuộc tính, tiêu chuẩn và điều kiện mà phần mềm cần phải đáp ứng, không liên quan trực tiếp đến chức năng cụ thể.
- Tập trung vào "như thế nào": Xác định cách thức mà hệ thống thực hiện các chức năng, bao gồm hiệu suất, bảo mật, khả năng sử dụng, và tính mở rộng.

Ví dụ:

- Hiệu suất: Hệ thống phải xử lý 1000 yêu cầu mỗi giây.
- Bảo mật: Dữ liệu người dùng phải được mã hóa và bảo vệ khỏi các cuộc tấn công.
- Khả năng sử dụng: Giao diện người dùng phải thân thiện và dễ sử dụng cho người dùng không có kinh nghiệm.

8. Thảo luận về các mô hình quy trình phát triển phần mềm phổ biến.

Mô hình Thác nước (Waterfall Model)

Đặc điểm:

Tuần tự: Các giai đoạn phát triển được thực hiện theo thứ tự: yêu cầu, thiết kế, lập trình, kiểm thử, triển khai và bảo trì.

Kết thúc từng giai đoạn: Mỗi giai đoạn phải hoàn thành trước khi chuyển sang giai đoạn tiếp theo.

Ưu điểm:

Dễ quản lý: Rõ ràng trong từng giai đoạn, dễ theo dõi tiến độ.

Tài liệu đầy đủ: Mỗi giai đoạn có tài liệu rõ ràng, giúp dễ dàng cho việc bảo trì sau này.

Nhược điểm:

Khó thay đổi: Khó khăn trong việc thay đổi yêu cầu sau khi đã bắt đầu phát triển.
Không linh hoạt: Không phù hợp với các dự án có yêu cầu thay đổi liên tục.

Mô hình Agile
Đặc điểm:
Linh hoạt và thích ứng: Tập trung vào việc phát triển phần mềm thông qua các vòng lặp ngắn (iteration).
Phản hồi nhanh: Cho phép người dùng tham gia vào quá trình phát triển và cung cấp phản hồi liên tục.
Ưu điểm:
Phản hồi nhanh chóng: Có thể điều chỉnh yêu cầu dễ dàng dựa trên phản hồi từ khách hàng.
Tăng cường hợp tác: Khuyến khích sự hợp tác giữa các nhóm phát triển và khách hàng.
Nhược điểm:
Khó quản lý: Có thể gây ra khó khăn trong việc quản lý nếu không có quy trình rõ ràng.
Thiếu tài liệu: Có thể dẫn đến việc thiếu tài liệu đầy đủ trong quá trình phát triển.

Mô hình V (V-Model)
Đặc điểm:
Kiểm thử song song: Mô hình này mở rộng mô hình thác nước bằng cách thêm các giai đoạn kiểm thử song song với giai đoạn phát triển.
Tập trung vào kiểm thử: Mỗi giai đoạn phát triển có một giai đoạn kiểm thử tương ứng.
Ưu điểm:
Chất lượng cao: Tăng cường chất lượng sản phẩm thông qua kiểm thử liên tục.
Rõ ràng trong tài liệu: Tài liệu rõ ràng cho từng giai đoạn phát triển và kiểm thử.
Nhược điểm:
Chi phí cao: Có thể tốn kém do yêu cầu nhiều nguồn lực cho kiểm thử.
Khó thay đổi: Cũng gặp khó khăn trong việc thay đổi yêu cầu như mô hình thác nước.

Mô hình Spiral

Đặc điểm:

Kết hợp giữa phát triển và kiểm thử: Kết hợp các yếu tố của mô hình thác nước và Agile, với việc phát triển theo vòng lặp (spiral).

Quản lý rủi ro: Tập trung vào việc xác định và giảm thiểu rủi ro trong từng vòng lặp.

Ưu điểm:

Linh hoạt: Có thể điều chỉnh yêu cầu và thiết kế trong từng vòng lặp.

Quản lý rủi ro tốt: Giúp phát hiện và xử lý rủi ro sớm trong quá trình phát triển.

Nhược điểm:

Phức tạp: Mô hình phức tạp và yêu cầu nhiều tài nguyên cho việc quản lý.

Khó khăn trong tài liệu: Có thể khó khăn trong việc duy trì tài liệu đầy đủ.

Mô hình DevOps

Đặc điểm:

Tích hợp phát triển và vận hành: Tích hợp chặt chẽ giữa phát triển phần mềm và hoạt động vận hành.

Tự động hóa: Sử dụng tự động hóa để tăng tốc độ phát triển và triển khai.

Ưu điểm:

Tăng tốc độ phát triển: Giúp giảm thời gian từ phát triển đến triển khai.

Cải thiện chất lượng: Thúc đẩy việc kiểm thử liên tục và phản hồi nhanh chóng.

Nhược điểm:

Yêu cầu văn hóa tổ chức: Cần sự thay đổi trong văn hóa tổ chức để áp dụng thành công.

Khó khăn trong quản lý: Có thể gặp khó khăn trong việc quản lý nếu không có quy trình rõ ràng.

9. Đề xuất giải pháp giảm thiểu lỗi phần mềm sau khi bàn giao.

Đề xuất giải pháp giảm thiểu lỗi phần mềm sau khi bàn giao :

- Kiểm thử toàn diện
- Tài liệu hóa rõ ràng
- Đào tạo và hỗ trợ người dùng
- Quản lý thay đổi hiệu quả
- Phản hồi liên tục

10.Vai trò của đội kiểm thử trong quy trình phát triển phần mềm.

Vai trò của đội kiểm thử trong quy trình phát triển phần mềm :
- Đảm bảo chất lượng sản phẩm
- Tăng cường sự hài lòng của khách hàng
- Giảm thiểu rủi ro
- Cải tiến liên tục
- Thúc đẩy quy trình phát triển

PHẦN TÌNH HUỐNG

Tình huống 1:

Một công ty phát triển phần mềm quản lý tài chính đã hoàn thành dự án và bàn giao cho khách hàng. Tuy nhiên, sau 2 tháng sử dụng, khách hàng phát hiện ra nhiều lỗi phát sinh khi phần mềm xử lý các giao dịch có giá trị lớn. Hãy đề xuất giải pháp xử lý tình huống này.

Trả lời :

Trong tình huống công ty phát triển phần mềm quản lý tài chính bàn giao sản phẩm nhưng sau hai tháng, khách hàng phát hiện nhiều lỗi khi xử lý giao dịch lớn, cần có giải pháp kịp thời. Đầu tiên, công ty nên tổ chức cuộc họp với khách hàng để hiểu rõ các lỗi và phân tích mã nguồn để xác định nguyên nhân. Sau đó, đội ngũ phát triển cần nhanh chóng sửa lỗi và cung cấp bản cập nhật phần mềm.

Tiếp theo, thực hiện kiểm thử hồi quy và kiểm thử hiệu suất để đảm bảo phần mềm hoạt động ổn định. Công ty cũng cần cập nhật tài liệu hướng dẫn và tổ chức buổi đào tạo cho khách hàng về cách sử dụng phần mềm hiệu quả.

Thiết lập hệ thống phản hồi để khách hàng báo cáo lỗi và theo dõi các vấn đề phát sinh cũng là điều cần thiết. Cuối cùng, đánh giá lại quy trình phát triển và cung cấp hỗ trợ kỹ thuật kịp thời sẽ giúp nâng cao chất lượng sản phẩm và khôi phục sự tin tưởng của khách hàng.

Tình huống 2:

Trong quá trình phát triển phần mềm quản lý bệnh viện, khách hàng yêu cầu bổ sung thêm tính năng quản lý kho thuốc khi dự án đã đi vào giai đoạn kiểm thử. Là trưởng nhóm phát triển, bạn sẽ xử lý yêu cầu này như thế nào?.

Trả lời :

Đầu tiên, em sẽ tổ chức cuộc họp với khách hàng để làm rõ yêu cầu bổ sung tính năng quản lý kho thuốc, nhằm hiểu mức độ quan trọng của tính năng này. Sau đó, phối hợp với đội ngũ phát triển để phân tích tác động đến tiến độ dự án.

Nếu tính năng có thể tích hợp mà không ảnh hưởng nhiều đến lịch trình, lập kế hoạch chi tiết cho việc phát triển và kiểm thử. Cần thông báo cho khách hàng về tác động đến thời gian hoàn thành và chi phí.

Sau khi nhận được sự đồng ý, chỉ đạo đội ngũ thực hiện tính năng mới và kiểm thử kỹ lưỡng trước khi triển khai. Cung cấp bản cập nhật phần mềm cho khách hàng và hướng dẫn sử dụng. Thiết lập kênh hỗ trợ sau khi cập nhật để đảm bảo sự hài lòng của khách hàng.

Tình huống 3:

Một nhóm phát triển phần mềm gặp phải vấn đề trễ tiến độ do nhiều thành viên không hiểu rõ yêu cầu của khách hàng. Là trưởng dự án, bạn sẽ làm gì để giải quyết vấn đề này và đảm bảo tiến độ dự án?

- Cần trao đổi lại với khách hảng để làm rõ yêu cầu.
- Chuyển đổi các yêu cầu thành đặc tả kỹ thuật để nhóm phát triển hiểu rõ hơn.
- Cần họp lại phân tích và giải thích để làm rõ yêu cầu của khách hàng và đảm bảo tất cả thành viên đều phải hiểu rõ.
- Xác định các nhiệm vụ cần làm và phân công lại công việc sao cho phù hợp với từng thành viên.

- Xác nhận lại với khách hàng ở từng giai đoạn để đảm bảo dự án đang đi theo đúng yêu cầu của khách hàng.
- Giám sát chặt chẽ từng giai đoạn để đảm bảo đúng tiến độ. Nếu tiến độ bị trễ cho thiếu nhân lực hoặc thiếu kinh nghiệm thì phải thuê thêm nhân lực.
- Phân tích để lường trước các rủi ro để đưa ra các phương án dự phòng để đảm bảo dự án luôn trong tầm kiểm soát và hoàn thành đúng tiến độ.

Tình huống 4:
Sau khi triển khai phần mềm quản lý thư viện, người dùng phản hồi rằng giao diện khó sử dụng và không thân thiện. Đội phát triển cần làm gì để cải thiện trải nghiệm người dùng?

- Cần phải thu thập thêm phản hồi từ người dùng về phần nào cần cải thiện.
- Tham khảo các phần mềm quản lý thư viện khác đã được phát triển và sử dụng rộng rãi để áp dụng vào phần mềm của mình.
- Cải thiện để đảm bảo tính tương thích của phần mềm trên các thiết bị khác nhau.
- Cung cấp tài liệu hướng dẫn chi tiết cho người dùng.
- Thêm chức năng hỗ trợ để có thể hỗ trợ trực tuyến khi người dùng có thắc mắc về phần mềm của mình.
- Đưa ra các giao diện thử nghiệm và tiếp tục thu thập phản hồi từ người dùng để cải thiện trước khi đưa ra phiên bản hoàn chỉnh.

Tình huống 5:
Một dự án phát triển phần mềm đã vượt quá ngân sách dự kiến do thời gian hoàn thành lâu hơn kế hoạch. Là quản lý dự án, bạn sẽ đề xuất những giải pháp nào để hạn chế việc vượt ngân sách trong tương lai?

- Cần phân tích lại để đảm bảo các chức năng là cần thiết, không có chức năng nào là dư thừa.
- Tính toán lại chi phí dựa trên thực tế và lập kế hoạch chi tiết phân phối chi phí cho từ giai đoạn hợp lý hơn.
- Quản lý lại nhân lực đảm bảo mọi người đều đủ khả năng hoàn thành công việc của mình.
- Giám sát chặt chẽ quá trình để kịp thời phát hiện vấn đề để giải quyết ngay.
- Phân tích để dự đoán được các rủi ro để kiểm soát không để các rủi ro xảy ra hoặc nếu rủi ro xảy ra thì phải có phương án dự phòng.

Tình huống 6:
Trong quá trình bảo trì phần mềm quản lý khách sạn, một nhân viên phát hiện ra một lỗi nhỏ không ảnh hưởng lớn đến hoạt động. Tuy nhiên, chi phí để sửa lỗi này khá cao. Bạn sẽ quyết định sửa lỗi hay không? Vì sao?

- Cần phải xem xét, đánh giá lại các vấn đề có thể phát sinh của lỗi này trong tương lai.

- Kiểm thử lại lỗi này xem có gây ra các rủi ro khác mà nhân viên không thể thấy được từ hướng người dùng như các rủi ro về tính bảo mật, tính toàn vẹn của cơ sở dữ liệu,…
- Xem xét lại mức độ ưu tiên của việc sửa lỗi này so với các nhiệm vụ hiện tại.
- Nếu qua các bước phân tích mà vẫn không phát hiện các vấn đề nghiêm trọng thì sẽ không sửa ngay bây giờ mà để chi phí dành cho các việc khác và sẽ sửa lỗi này trong lần bảo trì tiếp theo. Nếu phát hiện các vấn đề nghiêm trọng trong tương lai thì phải chấp nhận sửa ngay dù chi phí cao.

Tình huống 7:

-Cách xử lý tốt nhất khi khách hàng yêu cầu rút ngắn thời gian hoàn thành dự án trong khi đội phát triển đang gặp khó khăn về nhân lực và tài nguyên là cân nhắc giữa khả năng thực tế và yêu cầu của khách hàng.

-Các bước cụ thể để giải quyết tình huống này:

+Đánh giá lại phạm vi và tiến độ dự án:

- Xem xét kỹ lưỡng các nhiệm vụ hiện tại và thời gian hoàn thành dự kiến.
- Xác định các phần có thể được rút ngắn hoặc thực hiện song song.
- Phân tích xem có phần nào trong yêu cầu của khách hàng có thể được lùi lại cho các bản cập nhật sau không (ví dụ: các tính năng không quá quan trọng).

+Trao đổi trực tiếp với khách hàng:

- Giải thích rõ ràng về khó khăn hiện tại (thiếu nhân lực, hạn chế tài nguyên) và lý do vì sao việc rút ngắn tiến độ là thách thức.
- Đưa ra các phương án thay thế, ví dụ:
    - Giảm phạm vi của đợt triển khai đầu tiên và bổ sung sau thông qua bản cập nhật.
    - Tăng chi phí để thuê thêm nhân lực hoặc tài nguyên nếu khách hàng kiên quyết giữ thời hạn.
- Đề xuất một kế hoạch thực tế với các mốc thời gian hợp lý.

+Xem xét tăng cường nhân lực và tài nguyên:

- Thuê thêm nhân viên tạm thời hoặc tìm đối tác hỗ trợ ngoài (outsourcing).
- Tăng ca hoặc làm thêm giờ (với điều kiện nhân viên đồng ý và được trả công xứng đáng).

+Ưu tiên hóa công việc:

- Áp dụng nguyên tắc MoSCoW (Must have, Should have, Could have, Won’t have) để ưu tiên tính năng quan trọng.
- Tập trung nguồn lực vào các nhiệm vụ có tác động lớn nhất tới tiến độ.

+Tăng cường giao tiếp nội bộ:

- Cập nhật thường xuyên tiến độ công việc cho cả đội phát triển.
- Đảm bảo mọi thành viên hiểu rõ vai trò và trách nhiệm của mình.

+Đánh giá và giám sát thường xuyên:

- Thiết lập các cuộc họp định kỳ để theo dõi tiến độ và điều chỉnh kế hoạch khi cần thiết.
- Sử dụng công cụ quản lý dự án (như Jira, Trello) để quản lý công việc hiệu quả hơn.

Tình huống 8:

Trong tình huống một công ty phần mềm nhỏ gặp khó khăn do thay đổi công nghệ liên tục, dẫn đến trì hoãn tiến độ và tăng chi phí, giải pháp tốt nhất là tập trung vào việc ổn định công nghệ và quản lý dự án hiệu quả. Dưới đây là các bước cụ thể để khắc phục:

+Đánh giá lại yêu cầu và năng lực nội bộ

- Rà soát lại yêu cầu dự án: Xác định rõ ràng các tính năng quan trọng và phạm vi công việc (scope).
- Đánh giá năng lực của đội ngũ: Dựa trên kỹ năng hiện có của đội phát triển để chọn công nghệ phù hợp. Việc dùng một công nghệ quen thuộc sẽ giúp tiết kiệm thời gian và chi phí đào tạo.

+ Ổn định công nghệ từ sớm

- Chọn một stack công nghệ phù hợp ngay từ đầu (ví dụ: React Native hoặc Flutter nếu muốn phát triển ứng dụng đa nền tảng).
- Xác định rõ tiêu chí chọn công nghệ, như:
    - Độ phù hợp với yêu cầu kỹ thuật
    - Tính dễ bảo trì
    - Tài nguyên hỗ trợ và cộng đồng sử dụng
- Chỉ thay đổi công nghệ khi thực sự cần thiết và sau khi đã đánh giá tác động của nó đến tiến độ và chi phí.

+ Xây dựng kế hoạch và quy trình phát triển rõ ràng

- Lập kế hoạch dự án (Project Plan): Đặt ra các mốc thời gian rõ ràng cho từng giai đoạn.

- Áp dụng phương pháp phát triển phần mềm phù hợp như Agile để quản lý linh hoạt các thay đổi.
- Sử dụng công cụ quản lý dự án như Jira, Trello hoặc Asana để giám sát tiến độ và phân công nhiệm vụ.

+ Tăng cường đào tạo và hỗ trợ đội ngũ
- Tổ chức các khóa học hoặc workshop nội bộ để nâng cao kỹ năng cho các thành viên.
- Khuyến khích chia sẻ kiến thức trong nhóm để cải thiện hiệu suất làm việc.
- Nếu cần, hãy tìm kiếm chuyên gia hoặc đối tác bên ngoài để hỗ trợ kỹ thuật.

+Quản lý chi phí chặt chẽ
- Xác định ngân sách cụ thể cho từng giai đoạn phát triển.
- Theo dõi sát sao chi phí phát sinh để tránh vượt quá ngân sách ban đầu.
- Thực hiện các cuộc họp định kỳ để đánh giá tiến độ và chi phí.

+Giao tiếp liên tục với khách hàng
- Cập nhật thường xuyên tình hình dự án cho khách hàng để họ hiểu được lý do trì hoãn (nếu có).
- Thống nhất lại các thay đổi cần thiết trong phạm vi dự án, tránh thay đổi yêu cầu giữa chừng khi không thực sự cần thiết.

Tình huống 9:
Khi phát hiện một lỗi bảo mật nghiêm trọng sau khi đã bàn giao phần mềm cho khách hàng, đây là tình huống rất nhạy cảm và cần được xử lý nhanh chóng, chuyên nghiệp.
+ Đánh giá mức độ nghiêm trọng của lỗi ngay lập tức
- Xác định mức độ ảnh hưởng của lỗi:
    - Có gây rò rỉ dữ liệu nhạy cảm không?
    - Có thể bị hacker khai thác từ xa không?
    - Ảnh hưởng đến toàn bộ hệ thống hay chỉ một phần nhỏ?
- Xác minh xem lỗi đã bị khai thác hay chưa (kiểm tra log hệ thống, dấu vết tấn công).

+ Thông báo ngay lập tức cho khách hàng *(nhưng phải cẩn trọng)*

- Thông báo cho khách hàng về lỗi một cách minh bạch và chuyên nghiệp, tránh gây hoang mang.
- Giải thích rõ:
    - Lỗi phát sinh từ đâu (nếu có thể xác định nguyên nhân)
    - Mức độ ảnh hưởng của lỗi
    - Các bước khắc phục đang được tiến hành
- Cam kết với khách hàng về thời gian khắc phục và các biện pháp tạm thời bảo vệ hệ thống.

+Triển khai giải pháp khẩn cấp (Hotfix)

- Phát triển và thử nghiệm một bản vá lỗi bảo mật (security patch) càng sớm càng tốt.
- Nếu chưa thể sửa lỗi ngay lập tức, hãy thực hiện các biện pháp tạm thời như:
    - Chặn các cổng hoặc chức năng dễ bị tấn công.
    - Giới hạn quyền truy cập hoặc tạm thời vô hiệu hóa tính năng dễ bị khai thác.

+Tăng cường giám sát hệ thống

- Kích hoạt các công cụ giám sát an ninh (IDS/IPS) để phát hiện các hành động bất thường.
- Theo dõi nhật ký hệ thống (logs) để xác định các dấu hiệu tấn công hoặc truy cập trái phép.

+ Kiểm tra và kiểm thử lại toàn bộ hệ thống

- Sau khi khắc phục, thực hiện kiểm thử toàn diện để đảm bảo không còn lỗ hổng nào bị bỏ sót.
- Áp dụng kiểm thử xâm nhập (penetration testing) nếu cần thiết.

+ Cập nhật chính sách bảo mật và quy trình phát triển

- Rà soát lại quy trình phát triển phần mềm để tránh lỗi tương tự trong tương lai, bao gồm:
    - Thực hiện kiểm tra bảo mật tự động trong quá trình phát triển (Security Code Review).
    - Đào tạo đội ngũ phát triển về các nguyên tắc bảo mật.

    - Tích hợp các công cụ kiểm tra lỗ hổng như SonarQube hoặc OWASP Dependency-Check.

+ Báo cáo đầy đủ và minh bạch

- Soạn thảo một bản báo cáo chi tiết gửi cho khách hàng, bao gồm:
    - Mô tả lỗi và nguyên nhân gốc rễ (Root Cause Analysis)
    - Các biện pháp đã thực hiện để khắc phục
    - Các bước tiếp theo để tăng cường bảo mật hệ thống

Tình huống 10:

Để sửa đổi phần mềm phù hợp với quy trình sản xuất mới mà không ảnh hưởng đến hoạt động sản xuất hiện tại của khách hàng, đội phát triển nên thực hiện các bước sau:

+ Phân tích yêu cầu và tác động của thay đổi

- Thu thập thông tin chi tiết từ khách hàng về quy trình sản xuất mới.
- Xác định phạm vi thay đổi và các tính năng liên quan trong hệ thống hiện tại.
- Đánh giá ảnh hưởng của các thay đổi đối với:
    - Hiệu suất hệ thống.
    - Tính năng hiện tại.
    - Tính liên tục của quy trình sản xuất.

+ Lập kế hoạch sửa đổi phần mềm

- Xây dựng một kế hoạch chi tiết với các nội dung:
    - Các tính năng cần thay đổi hoặc thêm mới.
    - Mốc thời gian thực hiện và triển khai.
    - Tài nguyên cần thiết (nhân lực, công cụ, hạ tầng).
- Lên phương án giảm thiểu gián đoạn sản xuất, như cập nhật vào giờ thấp điểm hoặc cuối tuần.

+ Phát triển và kiểm thử trên môi trường độc lập

- Thiết lập môi trường kiểm thử (staging) giống với hệ thống sản xuất thật.
- Triển khai các thay đổi và tiến hành các bài kiểm thử như:
    - Unit Testing: Kiểm thử chức năng từng phần.
    - Integration Testing: Đảm bảo các module tương tác đúng.

    - User Acceptance Testing (UAT): Mời người dùng thực nghiệm để đánh giá hiệu quả.

+ Triển khai theo từng giai đoạn (Phased Rollout)

- Cập nhật phần mềm theo từng giai đoạn nhỏ, giảm thiểu rủi ro:
    - Triển khai thử nghiệm trên một phân xưởng nhỏ.
    - Theo dõi hiệu suất, phản hồi và điều chỉnh nếu cần.
    - Triển khai toàn bộ sau khi thử nghiệm thành công.

+ Đảm bảo sao lưu và bảo mật dữ liệu

- Tiến hành sao lưu toàn bộ dữ liệu hệ thống trước khi thực hiện bất kỳ thay đổi nào.
- Xác thực tính toàn vẹn của dữ liệu sau khi hoàn thành cập nhật.

+ Đào tạo nhân viên và hỗ trợ kỹ thuật

- Tổ chức các buổi hướng dẫn sử dụng các tính năng mới.
- Cung cấp tài liệu hướng dẫn chi tiết cho người dùng.
- Thiết lập kênh hỗ trợ kỹ thuật 24/7 trong giai đoạn đầu triển khai.

+ Giám sát và đánh giá sau cập nhật

- Theo dõi hoạt động của hệ thống trong thời gian đầu sau khi triển khai để phát hiện và khắc phục lỗi nhanh chóng.
- Đánh giá mức độ phù hợp của phần mềm với quy trình sản xuất mới và lấy phản hồi từ khách hàng để điều chỉnh thêm nếu cần.

Kết luận:
Việc sửa đổi phần mềm mà không ảnh hưởng đến sản xuất đòi hỏi một quy trình nghiêm ngặt, bao gồm phát triển trong môi trường độc lập, triển khai theo từng giai đoạn và kiểm thử kỹ lưỡng. Đảm bảo sự phối hợp chặt chẽ giữa đội phát triển và khách hàng sẽ giúp quá trình chuyển đổi diễn ra suôn sẻ và hiệu quả.