CHƯƠNG 1

Câu hỏi trắc nghiệm

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

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

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

A. 3

B. 4

C. 5

D. 6

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

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

A. 46%

B. 56%

C. 76%

D. 86%

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

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

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

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

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?

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 ngừoi dùng cuối như Microsoft Office, trình duyệt Web.
- Phần mềm nhúng: Phần mềm điều khiển các thiết bị phần cứng như máy giặt, máy điều hoà.

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

Công nghệ phần mềm giúp tạo ra các ứng dụng phục vụ nhu cầu con người, tự động hóa quy trình, tối ưu hóa công việc và cải thiện chất lượng cuộc sống. Phần mềm hỗ trợ trong nhiều lĩnh vực như y tế, giáo dục, tài chính, giao thông và giải trí. Nếu không có công nghệ phần mềm, hầu hết các hệ thống hiện đại sẽ không thể vận hành hiệu quả.

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 phụ lỗi và nâng cấp phần mềm sau khi triển khai.

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

Công nghệ phần mềm không chỉ là một lĩnh vực kỹ thuật mà còn có tác động lớn đến nền kinh tế, ảnh hưởng trực tiếp đến chi phí, lợi ích và hiệu quả tài chính của doanh nghiệp. Ngày nay, phần mềm là yếu tố cốt lõi trong nhiều ngành công nghiệp như tài chính, y tế,

giáo dục, đóng vai trò quan trọng trong việc tự động hóa quy trình, nâng cao năng suất và tối ưu hóa chi phí vận hành. Dưới đây là những khía cạnh kinh tế quan trọng của công nghệ phần mềm:

1. Chi phí phát triển phần mềm

- Việc phát triển phần mềm đòi hỏi nguồn lực lớn, bao gồm nhân sự (lập trình viên, kiểm thử viên, quản lý dự án), công cụ phát triển, cơ sở hạ tầng và phần mềm hỗ trợ.
- 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 thông qua các phương pháp phát triển linh hoạt như Agile, DevOps.

2. Chi phí bảo trì và vận hành

- Chi phí bảo trì phần mềm thường chiếm 50-70% tổng chi phí vòng đời sản phẩm, bao gồm sửa lỗi, cập nhật, mở rộng tính năng và tối ưu hóa hiệu suất.
- Nếu phần mềm được thiết kế tốt ngay từ đầu, doanh nghiệp có thể giảm thiểu chi phí bảo trì và tăng tính ổn định của hệ thống.

3. Giá trị kinh tế và lợi ích kinh doanh

- Sự phát triển của phần mềm giúp tăng năng suất lao động bằng cách tự động hóa quy trình, giảm sự phụ thuộc vào nhân công và cải thiện tốc độ xử lý công việc.
- Doanh nghiệp có thể thương mại hóa phần mềm theo nhiều mô hình khác nhau như bản quyền thương mại (COTS), phần mềm dịch vụ (SaaS), mã nguồn mở (Open-source), giúp tạo ra doanh thu ổn định.

4. Kinh tế quy mô trong phần mềm

- Không giống như sản phẩm vật lý, phần mềm có thể được sao chép và phân phối với chi phí gần như bằng không. Một phần mềm sau khi phát triển có thể được bán cho hàng triệu khách hàng mà không tốn thêm chi phí sản xuất, giúp tối ưu hóa lợi nhuận.
- Các tập đoàn lớn như Microsoft, Google, Amazon đã tận dụng yếu tố này để phát triển hệ sinh thái phần mềm mạnh mẽ, mang lại lợi nhuận khổng lồ.

5. Rủi ro tài chính trong phát triển phần mềm

- Các dự án phần mềm có thể gặp rủi ro như vượt ngân sách, chậm tiến độ, hoặc không đáp ứng được nhu cầu thị trường. Điều này có thể gây tổn thất tài chính lớn, đặc biệt đối với các doanh nghiệp khởi nghiệp.
- Việc áp dụng các phương pháp quản lý dự án như Scrum, Kanban giúp doanh nghiệp giảm thiểu rủi ro và tối ưu hóa nguồn lực.

6. Chi phí cơ hội

- Nếu một phần mềm không được phát triển đúng tiến độ hoặc chất lượng không đảm bảo, doanh nghiệp có thể mất cơ hội kinh doanh, đánh mất thị phần vào tay đối thủ.
- Trong bối cảnh cạnh tranh khốc liệt, việc đầu tư vào công nghệ phần mềm không chỉ là chiến lược tiết kiệm chi phí mà còn là yếu tố quyết định sự thành công của doanh nghiệp.

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

Công nghệ phần mềm là một lĩnh vực không ngừng phát triển, đòi hỏi các hệ thống phần mềm phải liên tục thích ứng với các xu hướng công nghệ mới, sự gia tăng khối lượng dữ liệu và nhu cầu kết nối hệ thống phức tạp. Dưới đây là những khía cạnh công nghệ quan trọng của công nghệ phần mềm:

1. Sự phát triển nhanh chóng của công nghệ

- Công nghệ phần mềm luôn phải cải tiến liên tục để đáp ứng các yêu cầu mới về hiệu suất, bảo mật và khả năng mở rộng.
- Các xu hướng như trí tuệ nhân tạo (AI), điện toán đám mây (Cloud Computing), chuỗi khối (Blockchain) và Internet vạn vật (IoT) đã làm thay đổi cách phần mềm được thiết kế và triển khai.
- Việc tự động hóa quy trình phát triển phần mềm (CI/CD) giúp giảm thời gian phát hành sản phẩm và nâng cao chất lượng phần mềm.

2. Kiến trúc phần mềm hiện đại

- Sự phát triển của kiến trúc microservices giúp phần mềm dễ bảo trì, mở rộng và cập nhật mà không ảnh hưởng đến toàn bộ hệ thống.
- Các ứng dụng serverless trên nền tảng AWS Lambda, Azure Functions giúp giảm chi phí vận hành và tăng hiệu quả xử lý.

- Containerization (Docker, Kubernetes) cho phép triển khai phần mềm linh hoạt hơn trên nhiều môi trường khác nhau.

3. Kết nối các hệ thống phức tạp

- Phần mềm ngày nay không hoạt động độc lập mà cần kết nối với nhiều hệ thống khác nhau, từ hệ thống doanh nghiệp (ERP, CRM), dịch vụ đám mây, đến thiết bị IoT.
- API (Application Programming Interface) đóng vai trò quan trọng trong việc tích hợp các hệ thống, cho phép dữ liệu luân chuyển một cách nhanh chóng và bảo mật.
- Các giao thức truyền thông như RESTful API, GraphQL, WebSocket giúp tối ưu hóa hiệu suất và khả năng mở rộng của hệ thống phần mềm.

4. Xử lý và phân tích khối lượng dữ liệu lớn

- Với sự bùng nổ của dữ liệu, phần mềm hiện đại cần khả năng thu thập, xử lý và phân tích dữ liệu theo thời gian thực.
- Các công nghệ như Big Data, Data Lake, Machine Learning giúp phần mềm khai thác dữ liệu hiệu quả, hỗ trợ doanh nghiệp ra quyết định dựa trên dữ liệu.
- Hệ quản trị cơ sở dữ liệu hiện đại như NoSQL (MongoDB, Cassandra), SQL phân tán (CockroachDB, Amazon Aurora) giúp xử lý dữ liệu lớn với hiệu suất cao.

5. Bảo mật trong công nghệ phần mềm

- Khi phần mềm ngày càng kết nối với nhiều hệ thống, nguy cơ bảo mật cũng gia tăng, đòi hỏi phải tích hợp các giải pháp bảo mật ứng dụng (AppSec), mã hóa dữ liệu, xác thực đa yếu tố (MFA) và Zero Trust Security.
- DevSecOps được áp dụng để tích hợp bảo mật vào toàn bộ vòng đời phát triển phần mềm, giúp phát hiện và khắc phục lỗ hổng sớm hơn.

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

Sau khi phần mềm được triển khai, việc bảo trì là một hoạt động quan trọng và tốn kém, nhằm đảm bảo phần mềm hoạt động ổn định, bảo mật và đáp ứng các yêu cầu thay đổi của người dùng. Theo thống kê, bảo trì phần mềm chiếm khoảng 60% tổng chi phí vòng đời của một hệ thống phần mềm, cho thấy đây là một khía cạnh quan trọng cần được quản lý hiệu quả.

a. Bảo trì sửa lỗi (Corrective Maintenance)

- Mục tiêu: Khắc phục lỗi xuất hiện trong quá trình sử dụng phần mềm.
- Lỗi có thể đến từ code, dữ liệu, giao diện hoặc hiệu suất hệ thống.
- Ví dụ: Fix lỗi crash ứng dụng, sửa lỗi hiển thị trên giao diện người dùng.

b. Bảo trì thích nghi (Adaptive Maintenance)

- Mục tiêu: Điều chỉnh phần mềm để thích nghi với sự thay đổi của môi trường bên ngoài.
- Thường liên quan đến cập nhật hệ điều hành, cơ sở dữ liệu, phần cứng hoặc tích hợp với hệ thống mới.
- Ví dụ: Cập nhật phần mềm để tương thích với phiên bản mới của Windows hoặc Android.

c. Bảo trì hoàn thiện (Perfective Maintenance)

- Mục tiêu: Cải thiện hiệu suất, tối ưu hóa hoặc bổ sung tính năng mới mà người dùng yêu cầu.
- Giúp tăng trải nghiệm người dùng và nâng cao hiệu suất của phần mềm.
- Ví dụ: Nâng cấp giao diện UI/UX, tối ưu thuật toán để chạy nhanh hơn.

d. Bảo trì phòng ngừa (Preventive Maintenance)

- Mục tiêu: Dự đoán và ngăn chặn lỗi xảy ra trong tương lai, giúp phần mềm hoạt động ổn định lâu dài.
- Thường liên quan đến tối ưu hóa mã nguồn, cải thiện bảo mật, kiểm tra hệ thống định kỳ.
- Ví dụ: Refactoring (tái cấu trúc) code để dễ bảo trì hơn, cập nhật vá lỗi bảo mật.

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

Trong quá trình phát triển phần mềm, việc hoàn thành dự án đúng tiến độ là một thách thức lớn. Thực tế, nhiều dự án bị trễ tiến độ so với kế hoạch ban đầu do ước lượng thời gian không chính xác và nhiều yếu tố khách quan lẫn chủ quan tác động. Dưới đây là các nguyên nhân phổ biến dẫn đến tình trạng này:

1. Yêu cầu thay đổi liên tục từ phía khách hàng

- Khách hàng có thể thay đổi yêu cầu giữa chừng, đòi hỏi nhóm phát triển phải điều chỉnh thiết kế, tính năng hoặc kiến trúc hệ thống.

- Việc thay đổi này tác động đến kế hoạch ban đầu, khiến nhóm phát triển phải làm lại một số phần hoặc thay đổi toàn bộ chiến lược.
- Nếu không có quy trình quản lý yêu cầu chặt chẽ, dự án dễ rơi vào tình trạng "scope creep" – tức là phạm vi dự án mở rộng liên tục mà không có kế hoạch rõ ràng.

2. Ước lượng thời gian không chính xác

- Ước lượng không chính xác là nguyên nhân phổ biến nhất dẫn đến trễ hạn.
- Một số yếu tố gây sai lệch trong ước lượng thời gian:
    - Đánh giá thấp độ phức tạp của phần mềm.
    - Không tính toán đúng thời gian phát sinh lỗi và sửa lỗi.
    - Không dự trù thời gian cho việc kiểm thử và triển khai.
- Nếu ước lượng sai ngay từ đầu, các giai đoạn sau dễ bị chồng chéo, dẫn đến chậm tiến độ.

3. Thiếu nhân lực hoặc sự phối hợp kém trong nhóm phát triển

- Thiếu nhân sự hoặc chất lượng nhân sự không đáp ứng yêu cầu khiến công việc bị đình trệ.
- Phân công công việc không hợp lý có thể dẫn đến một số thành viên quá tải trong khi người khác không có việc làm.
- Sự phối hợp kém giữa lập trình viên, kiểm thử viên, quản lý dự án và khách hàng có thể dẫn đến hiểu sai yêu cầu, phát triển sai hướng, gây mất thời gian chỉnh sửa.

4. Thiếu tài liệu và quy trình phát triển không rõ ràng

- Nếu không có tài liệu thiết kế phần mềm đầy đủ, lập trình viên có thể hiểu sai yêu cầu, dẫn đến việc phát triển sai tính năng.
- Quy trình phát triển không rõ ràng sẽ khiến nhóm làm việc thiếu tổ chức, khó kiểm soát tiến độ.
- Thiếu kiểm thử liên tục có thể dẫn đến phát hiện lỗi muộn, kéo dài thời gian sửa lỗi.

5. Công nghệ mới hoặc không phù hợp

- Việc áp dụng công nghệ mới mà nhóm phát triển chưa có kinh nghiệm có thể làm chậm tiến độ do thời gian học tập và thử nghiệm.

- Nếu lựa chọn công nghệ không phù hợp, có thể dẫn đến tình trạng tắc nghẽn hiệu suất hoặc khó mở rộng, khiến nhóm phải quay lại thay đổi công nghệ giữa chừng.

6. Vấn đề tài chính và thay đổi trong tổ chức

- Nếu dự án gặp vấn đề về ngân sách, có thể dẫn đến cắt giảm nhân sự hoặc tài nguyên, ảnh hưởng đến tiến độ.
- Thay đổi lãnh đạo hoặc tái cơ cấu công ty cũng có thể làm gián đoạn dự án, khiến quá trình phát triển bị trì hoãn.

7. Không có kế hoạch dự phòng rủi ro

- Nếu không có kế hoạch đối phó với rủi ro, khi gặp sự cố như nhân viên nghỉ việc, lỗi phần mềm nghiêm trọng, hoặc yêu cầu thay đổi lớn, dự án sẽ bị đình trệ.
- Một số nhóm phát triển không dự trù thời gian buffer để xử lý vấn đề phát sinh, dẫn đến việc chậm tiến độ.

Cách giảm thiểu tình trạng trễ hạn

- Áp dụng mô hình phát triển phù hợp: Agile, Scrum giúp linh hoạt xử lý yêu cầu thay đổi.
- Sử dụng phương pháp ước lượng chính xác: Kết hợp Planning Poker, PERT, Function Points để dự báo thời gian tốt hơn.
- Quản lý yêu cầu chặt chẽ: Xây dựng tài liệu yêu cầu (SRS), quản lý bằng công cụ như JIRA, Trello để theo dõi tiến độ.
- Cải thiện kỹ năng quản lý dự án: Lập kế hoạch chi tiết, phân bổ công việc hợp lý và giám sát tiến độ thường xuyên.
- Kiểm thử sớm và liên tục: Áp dụng CI/CD, Test Automation để phát hiện lỗi sớm và tránh sửa lỗi muộn.

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

1. Bảo trì sửa lỗi (Corrective Maintenance)

- Nhằm khắc phục các lỗi phát sinh trong quá trình sử dụng phần mềm.
- Các lỗi có thể bao gồm:
    - Lỗi chức năng: Phần mềm không hoạt động như mong đợi.
    - Lỗi giao diện: Sai sót về hiển thị hoặc trải nghiệm người dùng.

    - Lỗi bảo mật: Lỗ hổng bảo mật cần được vá để tránh rủi ro tấn công.
    - Lỗi hiệu suất: Hệ thống chậm, quá tải hoặc tiêu tốn tài nguyên bất thường.
- Ví dụ: Một ứng dụng ngân hàng gặp lỗi khi xử lý giao dịch trực tuyến, nhóm bảo trì sẽ cập nhật mã nguồn để khắc phục.

2. Bảo trì thích ứng (Adaptive Maintenance)

- Nhằm điều chỉnh phần mềm để thích ứng với những thay đổi của môi trường hệ thống.
- Các thay đổi có thể đến từ:
    - Hệ điều hành mới: Phần mềm cần cập nhật để hoạt động trên Windows 11 hoặc phiên bản iOS mới nhất.
    - Phần cứng mới: Hỗ trợ cho vi xử lý mới hoặc thiết bị di động mới.
    - Công nghệ mới: Tích hợp trí tuệ nhân tạo (AI), điện toán đám mây (Cloud) hoặc công nghệ bảo mật tiên tiến.
- Ví dụ: Một phần mềm kế toán cần cập nhật để tuân theo quy định tài chính mới của chính phủ.

3. Bảo trì hoàn thiện (Perfective Maintenance)

- Nhằm nâng cao hiệu suất, tính ổn định và trải nghiệm người dùng.
- Hoạt động này không sửa lỗi mà tập trung vào cải thiện hệ thống dựa trên phản hồi của người dùng và yêu cầu của doanh nghiệp.
- Các hoạt động chính:
    - Tối ưu hóa mã nguồn để tăng tốc độ xử lý.
    - Cải tiến giao diện người dùng (UI/UX) để thân thiện hơn.
    - Bổ sung tính năng mới để đáp ứng nhu cầu khách hàng.
- Ví dụ: Cập nhật thuật toán tìm kiếm trên một ứng dụng thương mại điện tử để giúp khách hàng tìm sản phẩm nhanh hơn.

4. Bảo trì phòng ngừa (Preventive Maintenance)

- Nhằm dự đoán và ngăn chặn các sự cố có thể xảy ra trong tương lai.
- Các hoạt động chính:

- Phân tích mã nguồn để phát hiện lỗi tiềm ẩn.
- Sao lưu dữ liệu định kỳ để đề phòng mất dữ liệu.
- Cập nhật hệ thống bảo mật để ngăn chặn nguy cơ tấn công mạng.

- Ví dụ: Một công ty triển khai các bản vá bảo mật hàng tháng để ngăn chặn lỗ hổng trước khi hacker khai thác.

Câu hỏi thảo luận nhóm

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

- **Phần mềm hệ thống:** Quản lý và điều khiển phần cứng máy tính, tạo nền tảng cho các ứng dụng hoạt động. Ví dụ: Hệ điều hành (Windows, Linux), trình điều khiển thiết bị.
- **Phần mềm ứng dụng**: Được phát triển để phục vụ nhu cầu cụ thể của người dùng, như soạn thảo văn bản, duyệt web, chơi game. Ví dụ: Microsoft Word, Google Chrome.

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

- Hỗ trợ giao dịch trực tuyến, ngân hàng điện tử.
- Cung cấp hệ thống quản lý tài chính, kế toán.
- Đảm bảo bảo mật dữ liệu và giao dịch an toàn.
- Phân tích dữ liệu tài chính, hỗ trợ ra quyết định.
- Tự động hóa quy trình như quản lý rủi ro, kiểm soát gian lận.

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

- **Tính tương thích**: Phần mềm cần được cập nhật để tương thích với hệ thống mới.
- **Bảo mật**: Liên tục vá lỗi bảo mật để chống lại các cuộc tấn công mạng.
- **Hiệu suất**: Phải tối ưu hóa để đảm bảo chạy mượt mà khi dữ liệu và số lượng người dùng tăng.
- **Yêu cầu thay đổi**: Người dùng yêu cầu thêm tính năng mới hoặc sửa đổi giao diện.
- **Tài liệu không đầy đủ**: Việc bảo trì gặp khó khăn nếu tài liệu thiết kế ban đầu không rõ ràng.

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

- **Cập nhật bảo mật**: Ngăn chặn hacker đánh cắp dữ liệu khách hàng.
- **Cải thiện hiệu suất**: Đảm bảo tốc độ tải trang nhanh, đáp ứng nhu cầu mua sắm.
- **Tương thích với công nghệ mới**: Hỗ trợ nhiều nền tảng thanh toán và thiết bị.
- **Cập nhật tính năng mới**: Cải thiện trải nghiệm khách hàng, tăng doanh thu.
- **Khắc phục lỗi hệ thống**: Đảm bảo quá trình mua hàng không bị gián đoạn.

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

- **Tăng chi phí và thời gian**: Thay đổi yêu cầu dẫn đến việc viết lại mã nguồn, kiểm thử lại, kéo dài tiến độ dự án.
- **Khó khăn trong quản lý dự án**: Dễ làm nhóm phát triển mất phương hướng, khó kiểm soát phạm vi dự án.
- **Xung đột giữa các bên liên quan**: Nhà phát triển, khách hàng, và nhà đầu tư có thể có ý kiến khác nhau.
- **Ảnh hưởng đến chất lượng phần mềm**: Các thay đổi liên tục có thể gây lỗi hoặc làm phần mềm trở nên phức tạp.
- **Giải pháp**: Sử dụng phương pháp phát triển linh hoạt (Agile), quản lý yêu cầu chặt chẽ ngay từ đầu.

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

Chi phí phát triển và chi phí bảo trì phần mềm đều đóng vai trò quan trọng trong vòng đời của một sản phẩm phần mềm. Tuy nhiên, chi phí bảo trì thường cao hơn nhiều so với chi phí phát triển ban đầu, chiếm từ 60% đến 80% tổng chi phí vòng đời của phần mềm.

Tiêu chí	Chi phí phát triển phần mềm	Chi phí bảo trì phần mềm
Khái niệm	Chi phí để thiết kế, lập trình, kiểm thử và triển khai phần mềm ban đầu.	Chi phí để sửa lỗi, nâng cấp, tối ưu hóa và đảm bảo phần mềm hoạt động lâu dài.
Tỷ lệ trong vòng đời phần mềm	Chiếm khoảng 20% - 40% tổng chi phí vòng đời phần mềm.	Chiếm khoảng 60% - 80% tổng chi phí vòng đời phần mềm.
Mục tiêu chính	Xây dựng phần mềm từ đầu với đầy đủ tính năng đáp ứng yêu cầu.	Duy trì, cập nhật và cải tiến phần mềm theo thời gian.
Hoạt động chính	- Phân tích yêu cầu, thiết kế hệ thống. - Viết mã nguồn, kiểm thử, triển khai.	- Sửa lỗi (bảo trì sửa lỗi). - Cập nhật để phù hợp với công nghệ mới (bảo trì thích ứng). - Cải thiện hiệu suất, thêm tính năng mới (bảo trì hoàn thiện). - Ngăn ngừa sự cố tiềm ẩn (bảo trì phòng ngừa).

Độ phức tạp	Dễ kiểm soát hơn vì các yêu cầu thường được xác định trước.	Khó kiểm soát vì yêu cầu có thể thay đổi liên tục và xuất hiện các vấn đề không lường trước.
Tác động đến doanh nghiệp	Cần đầu tư ban đầu lớn nhưng có thể lên kế hoạch trước.	Nếu không bảo trì tốt, phần mềm có thể lỗi thời, gây mất dữ liệu, giảm hiệu suất hoặc mất khách hàng.

Ví dụ : Phần mềm quản lý khách sạn

- Chi phí phát triển ban đầu: Công ty A thuê một nhóm lập trình viên để xây dựng hệ thống quản lý khách sạn với chi phí khoảng 100.000 USD.
- Chi phí bảo trì hàng năm:
    - Sửa lỗi phần mềm: 10.000 USD/năm
    - Cập nhật để tương thích với phần cứng và hệ điều hành mới: 20.000 USD/năm
    - Thêm tính năng mới (như tích hợp AI để dự báo đặt phòng): 30.000 USD/năm
    - Bảo mật và sao lưu dữ liệu: 15.000 USD/năm
      => Tổng chi phí bảo trì hàng năm: 75.000 USD
      => Sau 5 năm, chi phí bảo trì đã lên đến 375.000 USD, gấp 3,75 lần chi phí phát triển ban đầu.

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

Tiêu chí	Yêu cầu chức năng (Functional Requirements)	Yêu cầu phi chức năng (Non-functional Requirements)
Khái niệm	Mô tả hệ thống phải làm gì, các chức năng và tính năng mà phần mềm cần có để đáp ứng yêu cầu của người dùng.	Mô tả hệ thống hoạt động như thế nào, tập trung vào hiệu suất, bảo mật, khả năng mở rộng và trải nghiệm người dùng.

Mục tiêu	Định nghĩa các thao tác, dịch vụ và chức năng mà hệ thống phải cung cấp.	Định nghĩa các tiêu chí chất lượng, hiệu suất, bảo mật và các ràng buộc hệ thống.
Đối tượng chính	Người dùng cuối, khách hàng, nhà phát triển.	Kiến trúc sư hệ thống, quản lý dự án, đội bảo mật.
Ví dụ	- Đăng ký tài khoản người dùng. - Đăng nhập và xác thực tài khoản. - Tìm kiếm sản phẩm theo tên, danh mục. - Thanh toán trực tuyến bằng thẻ tín dụng.	- Hệ thống phải có thời gian phản hồi dưới 2 giây cho mỗi yêu cầu. - Dữ liệu người dùng phải được mã hóa để đảm bảo bảo mật. - Ứng dụng phải hỗ trợ 100.000 người dùng truy cập đồng thời. - Giao diện phải thân thiện, dễ sử dụng trên di động và máy tính.
Tác động đến phần mềm	Nếu không đáp ứng yêu cầu chức năng, hệ thống không thể hoạt động đúng.	Nếu không đáp ứng yêu cầu phi chức năng, hệ thống vẫn có thể chạy nhưng kém hiệu quả, chậm, không bảo mật.

Ví dụ : Website thương mại điện tử

- Yêu cầu chức năng:
    - Người dùng có thể tìm kiếm sản phẩm theo danh mục.
    - Hệ thống hỗ trợ thanh toán bằng nhiều phương thức (thẻ, ví điện tử, chuyển khoản).
    - Người dùng có thể đánh giá và bình luận về sản phẩm.
- Yêu cầu phi chức năng:
    - Website phải tải trang dưới 3 giây ngay cả khi có 10.000 lượt truy cập đồng thời.
    - Hệ thống phải có khả năng tự động mở rộng tài nguyên khi số lượng truy cập tăng đột biến.
    - Giao diện phải tương thích với cả máy tính, điện thoại và máy tính bảng.

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

1. Mô hình thác nước (Waterfall Model)
Mô hình này chia quá trình phát triển phần mềm thành các giai đoạn tuần tự, từ phân tích yêu cầu đến thiết kế, triển khai, kiểm thử và bảo trì. Mỗi giai đoạn phải hoàn thành trước khi chuyển sang giai đoạn tiếp theo.

Các giai đoạn:

1. Phân tích yêu cầu
2. Thiết kế hệ thống
3. Triển khai (lập trình)
4. Kiểm thử
5. Bảo trì

Ưu điểm:
✓ Dễ hiểu và dễ quản lý do có quy trình rõ ràng.
✓ Phù hợp với các dự án có yêu cầu ổn định, ít thay đổi.
✓ Dễ dàng lên kế hoạch và ước lượng chi phí.

Nhược điểm:
✗ Khó thích ứng với các thay đổi yêu cầu sau khi đã hoàn thành một giai đoạn.
✗ Nếu xảy ra lỗi trong giai đoạn đầu, chi phí sửa chữa sẽ rất cao.
✗ Thời gian phát triển dài, người dùng phải chờ đến khi hoàn thành mới có sản phẩm để sử dụng.

Ví dụ: Dùng cho phát triển phần mềm hệ thống nhúng, phần mềm quân sự, nơi yêu cầu phải rõ ràng ngay từ đầu.

2. Mô hình xoắn ốc (Spiral Model)
Mô hình này kết hợp lặp lại (iterative) và quản lý rủi ro, với mỗi vòng xoắn bao gồm các bước: lập kế hoạch, phân tích rủi ro, thực hiện và đánh giá.

Ưu điểm:
✓ Quản lý rủi ro tốt, thích hợp cho các dự án phức tạp.
✓ Linh hoạt trong việc thay đổi yêu cầu trong từng vòng lặp.
✓ Phù hợp với phần mềm lớn, có nhiều rủi ro chưa xác định.

Nhược điểm:

✗ Chi phí cao do cần nhiều tài nguyên để phân tích rủi ro.

✗ Quá trình phát triển phức tạp, khó quản lý.

Ví dụ: Áp dụng cho dự án phần mềm ngân hàng, phần mềm hàng không, nơi yêu cầu an toàn cao và có nhiều rủi ro.

3. Mô hình phát triển linh hoạt (Agile Model)

Mô hình này tập trung vào phát triển phần mềm theo từng vòng lặp nhỏ (iteration), với sự tham gia liên tục của khách hàng để điều chỉnh sản phẩm.

Ưu điểm:

✓ Thích ứng nhanh với sự thay đổi của yêu cầu.

✓ Tạo ra sản phẩm sớm và có thể phát hành từng phiên bản nhỏ.

✓ Giúp cải thiện sự hợp tác giữa các thành viên trong nhóm.

Nhược điểm:

✗ Đòi hỏi đội ngũ phát triển có kỹ năng cao.

✗ Khó dự đoán chi phí và thời gian hoàn thành chính xác.

✗ Không phù hợp với dự án có yêu cầu rõ ràng ngay từ đầu.

Ví dụ: Áp dụng cho các ứng dụng web, mobile app như Facebook, Zalo, Grab, nơi yêu cầu thay đổi thường xuyên.

4. Mô hình phát triển lặp và gia tăng (Incremental & Iterative Model)

- Mô hình gia tăng (Incremental): Chia phần mềm thành các module nhỏ, mỗi phần được phát triển và phát hành theo từng giai đoạn.
- Mô hình lặp (Iterative): Xây dựng phần mềm theo từng phiên bản, mỗi phiên bản cải tiến so với phiên bản trước.

Ưu điểm:

✓ Cho phép phát hành sớm các tính năng quan trọng.

✓ Giảm thiểu rủi ro nhờ cải thiện dần qua từng vòng lặp.

✓ Khách hàng có thể trải nghiệm và phản hồi ngay từ các giai đoạn đầu.

Nhược điểm:

✗ Quá trình phát triển có thể kéo dài do liên tục phải cập nhật.

✗ Cần có kế hoạch rõ ràng để tránh việc các phiên bản sau chồng chéo nhau.

Ví dụ: Phát triển phần mềm game, website thương mại điện tử như Shopee, Tiki, Lazada.

5. Mô hình DevOps
DevOps là mô hình kết hợp giữa phát triển phần mềm (Development) và vận hành hệ thống (Operations), giúp tự động hóa và tối ưu quy trình triển khai.

Ưu điểm:
✓ Rút ngắn thời gian phát hành phần mềm.
✓ Cải thiện độ ổn định và hiệu suất của hệ thống.
✓ Giúp phát hiện lỗi sớm và giảm thiểu rủi ro khi triển khai.

Nhược điểm:
✗ Cần công cụ và đội ngũ có chuyên môn cao.
✗ Yêu cầu tích hợp liên tục (CI/CD), không phù hợp với dự án nhỏ.

Ví dụ: Dùng trong các công ty công nghệ lớn như Google, Amazon, Netflix, nơi yêu cầu triển khai nhanh và ổn định

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

Sau khi bàn giao, phần mềm vẫn có thể gặp lỗi do nhiều nguyên nhân như yêu cầu chưa rõ ràng, lỗi lập trình, hoặc thay đổi trong môi trường triển khai. Dưới đây là một số giải pháp để giảm thiểu lỗi phần mềm sau khi bàn giao:

1. Kiểm thử kỹ lưỡng trước khi bàn giao:

- Thực hiện kiểm thử đơn vị (Unit Test), kiểm thử tích hợp (Integration Test), kiểm thử hệ thống (System Test) trước khi phát hành.
- Sử dụng kiểm thử tự động (Automated Testing) để phát hiện lỗi sớm.
- Áp dụng kiểm thử hồi quy (Regression Testing) khi có thay đổi.

Ví dụ: Ứng dụng ngân hàng điện tử (Mobile Banking App): Trước khi phát hành, ngân hàng thực hiện kiểm thử bảo mật (Security Testing) để ngăn chặn lỗi có thể gây rò rỉ thông tin tài khoản khách hàng.

2. Đào tạo người dùng và cung cấp tài liệu hướng dẫn:

- Hướng dẫn người dùng cách sử dụng phần mềm đúng cách.
- Cung cấp tài liệu hướng dẫn, video tutorial giúp người dùng hiểu rõ chức năng.

Ví dụ: Hệ thống ERP của doanh nghiệp: Sau khi triển khai, công ty tổ chức các buổi đào tạo cho nhân viên để họ sử dụng hệ thống hiệu quả, giảm thiểu lỗi thao tác.

3. Xây dựng quy trình bảo trì và hỗ trợ sau bàn giao:

- Thiết lập kênh hỗ trợ khách hàng để tiếp nhận và xử lý lỗi kịp thời.
- Áp dụng bảo trì phòng ngừa (Preventive Maintenance) để phát hiện và sửa lỗi trước khi chúng ảnh hưởng đến hệ thống.

Ví dụ: Google Chrome: Khi người dùng báo lỗi trình duyệt, Google có nhóm hỗ trợ phản hồi nhanh và phát hành bản vá lỗi trong vòng vài ngày.

4. Sử dụng công cụ giám sát và log lỗi:

- Cài đặt hệ thống giám sát (Monitoring System) để theo dõi hiệu suất và lỗi trong thời gian thực.
- Ghi lại nhật ký lỗi (Logging) để dễ dàng phân tích và xử lý sự cố.

Ví dụ: Facebook: Sử dụng AI và hệ thống log lỗi tự động để phát hiện các bài đăng vi phạm chính sách, giúp giảm lỗi hệ thống và trải nghiệm người dùng mượt mà hơn.

5. Kiểm tra và cập nhật phần mềm định kỳ:

- Phát hành các bản cập nhật định kỳ để sửa lỗi và cải thiện hiệu suất.
- Áp dụng bảo trì điều chỉnh (Adaptive Maintenance) nếu có thay đổi về môi trường sử dụng.

Ví dụ: Hệ điều hành Windows: Microsoft phát hành bản cập nhật bảo mật hàng tháng (Patch Tuesday) để vá lỗi và cải thiện hiệu suất hệ thống.

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

Đội kiểm thử phần mềm (QA - Quality Assurance) đóng vai trò quan trọng trong việc đảm bảo chất lượng sản phẩm trước khi phát hành. Họ giúp phát hiện lỗi, đảm bảo phần mềm đáp ứng yêu cầu và mang lại trải nghiệm tốt nhất cho người dùng.

1. Phát hiện và ngăn chặn lỗi trước khi triển khai:

- Kiểm thử giúp phát hiện lỗi logic, lỗi chức năng, lỗi giao diện, lỗi bảo mật trước khi phần mềm đến tay người dùng.
- Giảm chi phí sửa lỗi bằng cách phát hiện lỗi sớm trong vòng đời phát triển phần mềm.

Ví dụ: Một lỗi nhỏ trong ứng dụng ngân hàng có thể dẫn đến mất tiền của khách hàng. Đội kiểm thử giúp đảm bảo tính chính xác trước khi phát hành.

2. Đảm bảo phần mềm đáp ứng đúng yêu cầu:

- Đội kiểm thử so sánh phần mềm với tài liệu đặc tả yêu cầu (SRS - Software Requirement Specification) để đảm bảo phần mềm hoạt động đúng như mong muốn.
- Kiểm tra cả yêu cầu chức năng (các tính năng chính) và yêu cầu phi chức năng (hiệu suất, bảo mật, giao diện).

Ví dụ: Trong một website thương mại điện tử, đội kiểm thử sẽ kiểm tra xem quy trình đặt hàng, thanh toán, gửi email xác nhận có hoạt động chính xác không.

3. Cải thiện trải nghiệm người dùng (UX/UI Testing):

- Đảm bảo giao diện dễ sử dụng, trực quan, không có lỗi hiển thị trên các thiết bị khác nhau.
- Kiểm tra tính khả dụng (Usability Testing) để phát hiện các vấn đề gây khó chịu cho người dùng.

Ví dụ: Một ứng dụng di động bị lỗi font chữ hoặc nút bấm quá nhỏ sẽ gây khó khăn cho người dùng. Đội kiểm thử giúp phát hiện và đề xuất cải thiện.

4. Đảm bảo hiệu suất và khả năng mở rộng:

- Thực hiện kiểm thử tải (Load Testing), kiểm thử áp lực (Stress Testing), kiểm thử hiệu suất (Performance Testing) để đảm bảo hệ thống hoạt động mượt mà.
- Đánh giá khả năng mở rộng khi có số lượng người dùng lớn.

Ví dụ: Shopee, Lazada cần đảm bảo hệ thống không bị sập vào ngày sale lớn (11.11, 12.12) khi hàng triệu người truy cập cùng lúc.

5. Đảm bảo bảo mật phần mềm:

- Kiểm tra các lỗ hổng bảo mật (Penetration Testing) để ngăn chặn tấn công từ hacker.
- Đảm bảo dữ liệu người dùng được mã hóa an toàn.

Ví dụ: Ứng dụng thanh toán điện tử cần kiểm thử kỹ lưỡng để tránh lỗi SQL Injection, XSS, CSRF có thể khiến hacker đánh cắp thông tin thẻ tín dụng.

6. Hỗ trợ đội phát triển khắc phục lỗi nhanh chóng:

- Đội kiểm thử ghi lại lỗi trong hệ thống quản lý lỗi (JIRA, Bugzilla) và phối hợp với lập trình viên để sửa lỗi.
- Thực hiện kiểm thử hồi quy (Regression Testing) để đảm bảo lỗi đã sửa không ảnh hưởng đến các chức năng khác.

Ví dụ: Khi phát triển một tính năng mới trong Facebook, đội kiểm thử kiểm tra xem tính năng này có làm ảnh hưởng đến news feed, tin nhắn, video call hay không.

Câu hỏi tình huống

Câu hỏi 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.

Khi khách hàng phản ánh lỗi trong phần mềm quản lý tài chính, công ty cần nhanh chóng tiếp nhận thông tin, xác định nguyên nhân và mức độ ảnh hưởng. Trước hết, đội ngũ kỹ thuật phải kiểm tra lại thuật toán xử lý giao dịch, đặc biệt với các giao dịch có giá trị lớn, để tìm ra lỗi và điều chỉnh. Sau đó, phần mềm cần được kiểm thử kỹ lưỡng trước khi cập nhật cho khách hàng, đồng thời hướng dẫn họ thực hiện nâng cấp. Để tránh tình trạng này tái diễn, công ty nên cải thiện quy trình kiểm thử, bổ sung các tình huống kiểm tra thực tế, nâng cao năng lực đội ngũ phát triển và thiết lập chính sách hỗ trợ sau bàn giao. Quan trọng hơn, công ty cần chủ động xin lỗi khách hàng, hỗ trợ khắc phục nhanh chóng và cam kết đảm bảo chất lượng dịch vụ để duy trì uy tín và sự tin tưởng.

Câu hỏi 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?

Khi khách hàng yêu cầu bổ sung tính năng quản lý kho thuốc trong giai đoạn kiểm thử, với vai trò trưởng nhóm phát triển, em sẽ xử lý tình huống này theo các bước sau: Đầu tiên, ta cần phải tiếp nhận yêu cầu từ khách hàng và phải làm rõ phạm vi tính năng mới mà khách hàng yêu cầu, bao gồm các chức nắng cần có, mức độ ưu tiên của chức năng, thời gian mong muốn và tác động của nó đối với hệ thống hiện tại. Tiếp theo, ta phải phối hợp với đội ngũ phân tích mức độ ảnh hưởng của của tính năng này với tiến độ của dự án, kiến trúc phần mềm và cả tài nguyên chúng ta đang có. Nếu việc bổ sung không gây ra gián đoạn lớn cho dự án thì sẽ tiến hành tích hợp ngay trong giai đoạn kiểm thử. Tuy nhiên, nếu tính năng quá phức tạp, ta cần phải thảo luận với khách hàng đưa ra các phương án khác như đưa tính năng vào phiên bản cập nhật sau khi triển hoặc chia thành nhiều giai đoạn tùy thuộc vào tính năng khách hàng yêu cầu để giảm thiểu rủi ro cho dự án. Sau khi đã thống

nhất phương án thực hiện, tiến hành điều phối nhóm phát triển để thiết kể, triển khai và kiểm thử tính năng mới cunngx như phối hợp với khách hàng để kiểm tra và xác nhận tính năng đã đáp ứng đúng yêu cầu của khách hàng không trước khi tích hợp vào hệ thống. Cuối cùng, cập nhật lại tài liệu dự án và quy trình kiểm thử để đảm bảo tính nhất quán trong toàn bộ hệ thống.

Câu hỏi 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?

Khi nhóm phát triển phần mềm bị trễ tiến độ do nhiều thành viên không hiểu rõ yêu cầu của khách hàng, để giải quyết vấn đề này và đảm bảo tiến độ dự án, trước tiên, cần tổ chức một cuộc họp nội bộ để xác định cụ thể những điểm chưa rõ trong yêu cầu cũng như mức độ ảnh hưởng của chúng đến tiến độ chung. Sau đó, cần liên hệ trực tiếp với khách hàng để làm rõ các yêu cầu chưa thống nhất, đồng thời ghi nhận mọi thay đổi hoặc điều chỉnh cần thiết. Nếu cần thiết, có thể đề xuất tổ chức các buổi trao đổi giữa nhóm phát triển và khách hàng để đảm bảo tất cả thành viên đều hiểu đúng về yêu cầu dự án. Tiếp theo, cần cập nhật lại tài liệu yêu cầu một cách rõ ràng, dễ hiểu và cung cấp hướng dẫn cụ thể cho từng thành viên trong nhóm. Đồng thời, việc xem xét lại kế hoạch dự án, ưu tiên các nhiệm vụ quan trọng và phân công công việc hợp lý cũng là điều cần thiết để bù đắp phần tiến độ đã bị trễ. Ngoài ra, cần thiết lập cơ chế giao tiếp hiệu quả hơn trong nhóm, chẳng hạn như họp ngắn hằng ngày để cập nhật tiến độ và giải quyết các vướng mắc kịp thời. Nếu cần thiết, có thể tổ chức các buổi đào tạo bổ sung để đảm bảo các thành viên nắm vững yêu cầu và thực hiện đúng nhiệm vụ. Cuối cùng, việc theo dõi sát sao tiến độ, điều chỉnh linh hoạt khi cần thiết và phối hợp chặt chẽ với khách hàng sẽ giúp đảm bảo sản phẩm đáp ứng đúng yêu cầu mà không làm ảnh hưởng đến chất lượng hoặc thời gian bàn giao.

Câu hỏi 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?

Khi nhận được phản hồi về việc giao diện phần mềm quản lý thư viện khó sử dụng và không thân thiện, đội phát triển cần nhanh chóng tìm cách cải thiện trải nghiệm người dùng. Trước tiên, cần thu thập ý kiến chi tiết từ người dùng thông qua khảo sát, phỏng vấn hoặc quan sát trực tiếp quá trình họ sử dụng phần mềm để xác định rõ những điểm gây khó khăn. Sau đó, phân tích các vấn đề này để tìm ra nguyên nhân, có thể liên quan đến bố cục giao diện, màu sắc, phông chữ hoặc thao tác sử dụng phức tạp. Dựa trên những thông tin thu thập được, đội phát triển có thể điều chỉnh lại thiết kế theo hướng trực quan, đơn giản và dễ sử dụng hơn. Việc áp dụng các nguyên tắc UX/UI như tối ưu điều hướng, sử dụng biểu

tượng trực quan, bổ sung hướng dẫn sử dụng hoặc tính năng trợ giúp cũng sẽ giúp nâng cao trải nghiệm người dùng. Trước khi triển khai phiên bản cập nhật, cần tiến hành kiểm thử với một nhóm người dùng thực tế để đảm bảo những thay đổi thực sự mang lại hiệu quả. Sau khi cập nhật phần mềm, đội ngũ hỗ trợ kỹ thuật nên tiếp tục theo dõi phản hồi để có những điều chỉnh kịp thời, giúp phần mềm ngày càng hoàn thiện hơn.

Câu hỏi 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?

Trả lời: Khi một dự án phần mềm vượt quá ngân sách do thời gian hoàn thành kéo dài hơn kế hoạch, cần có những giải pháp cụ thể để hạn chế tình trạng này trong tương lai. Trước hết, việc lập kế hoạch chi tiết và thực tế ngay từ đầu là rất quan trọng. Cần đánh giá kỹ lưỡng phạm vi dự án, ước lượng chính xác nguồn lực, thời gian và chi phí, đồng thời dự trù ngân sách cho các rủi ro có thể phát sinh. Tiếp theo, áp dụng các phương pháp quản lý dự án hiệu quả như Agile hoặc Scrum giúp theo dõi tiến độ liên tục, phản ứng linh hoạt trước các thay đổi và tối ưu hóa quy trình làm việc. Bên cạnh đó, cần giám sát chặt chẽ tiến độ và ngân sách trong suốt quá trình triển khai, sử dụng các công cụ quản lý để phát hiện sớm các dấu hiệu chậm trễ hoặc chi tiêu vượt mức, từ đó đưa ra các điều chỉnh kịp thời. Ngoài ra, việc tối ưu hóa nguồn lực, đảm bảo đội ngũ làm việc hiệu quả và giảm thiểu lãng phí cũng góp phần kiểm soát ngân sách. Cuối cùng, cần thường xuyên trao đổi với khách hàng để kiểm soát phạm vi yêu cầu, tránh phát sinh thêm những thay đổi lớn làm kéo dài thời gian và tăng chi phí. Bằng cách áp dụng những biện pháp này, dự án có thể được thực hiện đúng tiến độ và trong giới hạn ngân sách cho phép.

Câu hỏi 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?

Việc quyết định sửa lỗi hay không cần dựa trên mức độ ảnh hưởng của lỗi đối với hệ thống và chi phí khắc phục. Nếu lỗi không gây tác động lớn đến hoạt động của phần mềm và không ảnh hưởng đến trải nghiệm người dùng, có thể tạm thời ghi nhận và theo dõi thay vì sửa ngay lập tức, đặc biệt khi chi phí khắc phục quá cao. Tuy nhiên, cũng cần xem xét khả năng lỗi này có thể dẫn đến những vấn đề nghiêm trọng hơn trong tương lai hay không. Nếu có nguy cơ ảnh hưởng đến dữ liệu, bảo mật hoặc hiệu suất hệ thống, thì việc sửa lỗi sớm có thể giúp tránh được chi phí lớn hơn sau này. Ngoài ra, nếu phần mềm đang trong giai đoạn bảo trì có kế hoạch, có thể đợi đến lần cập nhật tiếp theo để sửa lỗi nhằm tối ưu chi phí. Vì vậy, quyết định sửa hay không cần được cân nhắc kỹ lưỡng dựa trên mức độ ảnh hưởng, tính cấp thiết và ngân sách hiện có.

Câu hỏi tình huống 7: Khách hàng yêu cầu đội phát triển phải hoàn thành dự án sớm hơn 1 tháng so với kế hoạch ban đầu. Đội phát triển đang gặp khó khăn về nhân lực và tài nguyên. Bạn sẽ xử lý yêu cầu này như thế nào?

Khi khách hàng yêu cầu hoàn thành dự án sớm hơn một tháng trong khi đội phát triển đang gặp khó khăn về nhân lực và tài nguyên, cần có phương án xử lý hợp lý để đảm bảo chất lượng dự án mà vẫn đáp ứng yêu cầu khách hàng. Trước tiên, cần trao đổi trực tiếp với khách hàng để hiểu rõ lý do của yêu cầu rút ngắn thời gian và đánh giá mức độ linh hoạt trong thời gian bàn giao. Nếu có thể, đề xuất một giải pháp trung gian như rút ngắn thời gian nhưng không quá một tháng để giảm áp lực cho đội ngũ. Tiếp theo, phân tích tiến độ hiện tại, xác định các nhiệm vụ quan trọng và xem xét điều chỉnh quy trình làm việc để tối ưu hóa thời gian. Nếu cần thiết, có thể cân nhắc tăng cường nhân lực bằng cách huy động nội bộ hoặc thuê thêm nhân sự tạm thời. Đồng thời, áp dụng các phương pháp làm việc hiệu quả như Agile để tăng tốc độ phát triển và kiểm thử liên tục. Nếu việc hoàn thành sớm có nguy cơ ảnh hưởng đến chất lượng phần mềm, cần thông báo rõ ràng với khách hàng về những rủi ro tiềm ẩn và đưa ra phương án như chia nhỏ dự án để bàn giao từng phần. Việc minh bạch trong trao đổi và đưa ra giải pháp hợp lý sẽ giúp cân bằng giữa yêu cầu khách hàng và khả năng thực hiện của đội phát triển.

Câu hỏi tình huống 8: Một công ty phần mềm nhỏ nhận được dự án phát triển ứng dụng di động. Do hạn chế về nguồn lực và kinh nghiệm, công ty đã liên tục thay đổi công nghệ sử dụng trong dự án. Điều này khiến dự án bị kéo dài và chi phí tăng cao. Bạn sẽ đưa ra giải pháp gì để khắc phục?

Để khắc phục tình trạng kéo dài tiến độ và tăng chi phí do liên tục thay đổi công nghệ trong dự án phát triển ứng dụng di động, công ty cần có chiến lược rõ ràng ngay từ đầu. Trước tiên, cần đánh giá lại yêu cầu dự án và xác định công nghệ phù hợp dựa trên nguồn lực, kinh nghiệm sẵn có và tính ổn định lâu dài. Thay vì liên tục thay đổi, công ty nên chọn một công nghệ phù hợp và cam kết duy trì trong suốt quá trình phát triển để tránh lãng phí thời gian và tài nguyên. Ngoài ra, việc nâng cao năng lực đội ngũ bằng cách tổ chức các buổi đào tạo hoặc thuê chuyên gia tư vấn sẽ giúp giảm bớt khó khăn khi áp dụng công nghệ mới. Đồng thời, công ty cần áp dụng phương pháp quản lý dự án hiệu quả, như Agile hoặc Scrum, để kiểm soát tiến độ và điều chỉnh linh hoạt mà không làm gián đoạn quá trình phát triển. Việc lập kế hoạch rõ ràng, tránh thay đổi không cần thiết và tối ưu hóa nguồn lực sẽ giúp dự án hoàn thành đúng tiến độ và tiết kiệm chi phí.

Câu hỏi tình huống 9: Sau khi bàn giao phần mềm cho khách hàng, đội phát triển phát hiện ra một lỗi bảo mật nghiêm trọng có thể bị hacker khai thác. Là người phụ trách dự án, bạn sẽ giải quyết tình huống này như thế nào?

Khi phát hiện một lỗi bảo mật nghiêm trọng sau khi bàn giao phần mềm cho khách hàng, cần hành động nhanh chóng và có kế hoạch xử lý rõ ràng để giảm thiểu rủi ro. Trước tiên, tôi sẽ đánh giá mức độ nghiêm trọng của lỗi, xác định phạm vi ảnh hưởng và nguy cơ bị khai thác. Nếu lỗi có thể gây thiệt hại nghiêm trọng, tôi sẽ đề xuất tạm thời vô hiệu hóa tính năng liên quan hoặc cung cấp hướng dẫn giảm thiểu rủi ro cho khách hàng trong khi đội phát triển khẩn trương triển khai bản vá. Đồng thời, cần thông báo minh bạch cho khách hàng về sự cố, phương án khắc phục và thời gian dự kiến hoàn thành để đảm bảo sự tin tưởng. Sau khi sửa lỗi, tôi sẽ yêu cầu kiểm thử bảo mật kỹ lưỡng để tránh lặp lại tình trạng tương tự. Cuối cùng, cần xem xét lại quy trình phát triển và kiểm thử bảo mật để cải thiện, đảm bảo rằng các vấn đề bảo mật được phát hiện sớm hơn trong các dự án tiếp theo.

Câu hỏi tình huống 10: Dự án phát triển hệ thống quản lý sản xuất đã được triển khai thành công tại nhà máy. Tuy nhiên, do thay đổi quy trình sản xuất, khách hàng yêu cầu sửa đổi phần mềm để phù hợp với quy trình mới. Đội phát triển cần làm gì để đáp ứng yêu cầu này mà không làm ảnh hưởng đến hoạt động sản xuất của khách hàng?

Để đáp ứng yêu cầu sửa đổi phần mềm theo quy trình sản xuất mới mà không ảnh hưởng đến hoạt động của khách hàng, đội phát triển cần lập kế hoạch triển khai một cách cẩn trọng. Trước tiên, cần làm việc chặt chẽ với khách hàng để hiểu rõ những thay đổi trong quy trình sản xuất và xác định các tính năng cần chỉnh sửa hoặc bổ sung. Sau đó, đội phát triển sẽ phân tích tác động của các thay đổi này đối với hệ thống hiện tại để đảm bảo không gây lỗi hoặc gián đoạn hoạt động. Một môi trường kiểm thử độc lập nên được thiết lập để thử nghiệm các tính năng mới trước khi triển khai chính thức. Ngoài ra, cần thực hiện triển khai theo từng giai đoạn hoặc vào thời điểm ít ảnh hưởng nhất đến hoạt động sản xuất, chẳng hạn ngoài giờ làm việc hoặc cuối tuần. Cuối cùng, đội phát triển nên cung cấp tài liệu hướng dẫn và hỗ trợ đào tạo để đảm bảo nhân viên nhà máy có thể sử dụng hệ thống mới một cách hiệu quả ngay sau khi cập nhật.