Universidade Federal de Pernambuco :: Centro de Informática
Sistemas de Informação :: Engenharia de Software
Prof. Vinicius Cardoso Garcia

INSTRUÇÕES: Leia o enunciado com atenção e cuidado, e responda a questão do exercício.

- Esta avaliação tem 20 questões objetivas para um total de 10 pontos com **sua resolução sendo individual e sem consulta**.
- Adicionalmente, esta avliação também tem 1 questão EXTRA **não obrigatória**.
- Organize o tempo, a prova tem duração de **até 2 horas**.
- Por favor, mantenha-se focado na prova. Não é permitido **abrir qualquer aba** diferente do formulário do 3º EE durante a realização da avaliação a não ser, em caráter **EXTRAORDINÁRIO**, o uso do Bloco de Notas, vim, gedit ou similares/equivalentes.
- Não é permitido **nada em cima da mesa ou no colo**. Guardem os celulares no bolso ou na mochila (ou equivalente) e a mesma deve estar no chão.
- Responda todas as questões no **formulário de respostas**.
- Dúvidas podem ser expostas, **publicamente**, durante **os primeiros 30 minutos**.
- Não é permitido sair da sala (i.e. ir ao sanitário) **durante a realização da prova**. Certifique-se de atender a essa necessidade antes do início da avaliação.
- **Entender o enunciado faz parte da avaliação**.

Bem-vindo à seção de questões objetivas da prova. Por favor, leia atentamente cada enunciado e escolha a opção que melhor responde à pergunta proposta. Lembre-se de que esta seção contém 20 questões, cada uma valendo 0,5 pontos, totalizando 10 pontos.

É fundamental que você mantenha o foco e a concentração durante toda a resolução das questões. Não hesite em reler os enunciados e as alternativas antes de selecionar sua resposta.

1. Qual das seguintes opções melhor descreve uma preocupação chave relacionada aos impactos sociais da tecnologia no contexto de desenvolvimento sustentável?

(a) Aumento da dependência de dispositivos móveis e redes sociais.
(b) Dificuldade em manter a compatibilidade entre diferentes tecnologias.
(c) Contribuição para a desigualdade social devido ao acesso desigual à tecnologia.
(d) Necessidade de atualizações constantes de hardware e software.
(e) Redução do número de empregos devido à automação e inteligência artificial.

2. Qual das seguintes opções é uma aplicação comum do aprendizado de máquina na engenharia de software?

(a) Automatizar o processo de geração de código a partir de especificações de design.
(b) Substituir a necessidade de testes de software manuais em todos os cenários.
(c) Prever o tempo necessário para a conclusão de tarefas de desenvolvimento baseado em dados históricos.
(d) Criar interfaces gráficas de usuário sem a intervenção de desenvolvedores.
(e) Eliminar completamente a ocorrência de bugs em softwares desenvolvidos.

3. O que é Big Data e por que é importante no contexto da Engenharia de Software?

(a) Big Data refere-se ao volume de dados que excede a capacidade de processamento dos sistemas tradicionais de banco de dados. É importante na Engenharia de Software porque permite às organizações analisar grandes quantidades de dados para obter insights valiosos, como padrões de comportamento do usuário e tendências de mercado, para informar o desenvolvimento de software e a tomada de decisões estratégicas.
(b) Big Data refere-se à complexidade dos dados, incluindo sua variedade, velocidade e volume. É importante na Engenharia de Software porque permite às organizações lidar com a diversidade de dados gerados por diferentes fontes, como mídias sociais,

dispositivos móveis e sensores, e processá-los de maneira eficiente para obter informações úteis.

(c) Big Data refere-se à capacidade de armazenar grandes volumes de dados em sistemas distribuídos e acessá-los rapidamente para análise. É importante na Engenharia de Software porque permite às organizações lidar com a escalabilidade de dados gerados por aplicativos e dispositivos, garantindo que o software possa escalar conforme necessário para atender às demandas de processamento de dados em tempo real.

(d) Big Data refere-se à capacidade de analisar grandes conjuntos de dados usando algoritmos avançados de processamento e aprendizado de máquina. É importante na Engenharia de Software porque permite às organizações descobrir padrões ocultos e insights valiosos em dados complexos, como dados não estruturados e semiestruturados, para melhorar a tomada de decisões e a personalização de serviços.

(e) Big Data refere-se à confiabilidade e à qualidade dos dados, garantindo que sejam precisos, completos e consistentes. É importante na Engenharia de Software porque permite às organizações garantir a integridade dos dados em todos os estágios do ciclo de vida do software, desde a coleta até a análise e o uso em aplicativos e sistemas.

4. Qual das seguintes alternativas melhor descreve as características principais de uma API RESTful?

(a) Utiliza o protocolo SOAP para comunicação entre serviços.
(b) Requer estado de sessão para manter a integridade das transações.
(c) Usa métodos HTTP para realizar operações em recursos.
(d) Depende fortemente de estruturas de dados complexas para comunicação.
(e) É exclusivamente voltada para a integração de sistemas legados.

5. Qual é o objetivo principal da integração de sistemas em Engenharia de Software?

(a) O objetivo principal é garantir que todos os sistemas e aplicações tenham interfaces de usuário consistentes para melhorar a experiência do usuário final.
(b) O objetivo principal é conectar diferentes sistemas e aplicações para que possam compartilhar dados e funcionalidades, facilitando a criação de soluções mais completas e eficientes.
(c) O objetivo principal é garantir que todos os sistemas e aplicações estejam protegidos contra ameaças de segurança, como ataques de hackers e vazamento de dados.
(d) O objetivo principal é documentar os requisitos de diferentes sistemas e aplicações para garantir que eles possam funcionar juntos de forma harmoniosa, facilitando a comunicação entre equipes de desenvolvimento.
(e) O objetivo principal é testar sistemas individualmente antes de integrá-los em um ambiente de produção, garantindo a qualidade e confiabilidade das soluções desenvolvidas.

6. O que é Mapeamento Objeto-Relacional (ORM) e qual é o objetivo principal dessa técnica no contexto de desenvolvimento de software?

(a) O Mapeamento Objeto-Relacional (ORM) é uma técnica de desenvolvimento de software que mapeia os objetos do modelo de domínio de uma aplicação para as tabelas do banco de dados relacional. O objetivo principal do ORM é facilitar a integração entre o paradigma orientado a objetos utilizado na programação e o modelo relacional utilizado nos sistemas de gerenciamento de banco de dados, permitindo que os desenvolvedores trabalhem com objetos de maneira mais natural e eficiente, sem se preocupar com detalhes de baixo nível da persistência de dados.
(b) O Mapeamento Objeto-Relacional (ORM) é uma técnica de desenvolvimento de software que mapeia os objetos do modelo de domínio de uma aplicação para os documentos do banco de dados NoSQL. O objetivo principal do ORM é facilitar a integração entre o paradigma orientado a objetos utilizado na programação e o modelo de dados semi-estruturado utilizado nos bancos de dados NoSQL, permitindo que os

desenvolvedores trabalhem com objetos de maneira mais natural e eficiente, sem se preocupar com detalhes de baixo nível da persistência de dados.

(c) O Mapeamento Objeto-Relacional (ORM) é uma técnica de desenvolvimento de software que mapeia os objetos do modelo de domínio de uma aplicação para as APIs de acesso a dados. O objetivo principal do ORM é facilitar a integração entre o paradigma orientado a objetos utilizado na programação e as interfaces de acesso a dados fornecidas pelos sistemas de gerenciamento de banco de dados, permitindo que os desenvolvedores trabalhem com objetos de maneira mais natural e eficiente, sem se preocupar com detalhes de baixo nível da persistência de dados.

(d) O Mapeamento Objeto-Relacional (ORM) é uma técnica de desenvolvimento de software que mapeia os objetos do modelo de domínio de uma aplicação para os registros do sistema operacional. O objetivo principal do ORM é facilitar a integração entre o paradigma orientado a objetos utilizado na programação e os sistemas operacionais utilizados nos servidores de aplicação, permitindo que os desenvolvedores trabalhem com objetos de maneira mais natural e eficiente, sem se preocupar com detalhes de baixo nível da persistência de dados.

(e) O Mapeamento Objeto-Relacional (ORM) é uma técnica de desenvolvimento de software que mapeia os objetos do modelo de domínio de uma aplicação para as interfaces gráficas de usuário. O objetivo principal do ORM é facilitar a integração entre o paradigma orientado a objetos utilizado na programação e as interfaces de usuário fornecidas pelos sistemas de interface gráfica, permitindo que os desenvolvedores trabalhem com objetos de maneira mais natural e eficiente, sem se preocupar com detalhes de baixo nível da persistência de dados.

7. Qual das seguintes opções melhor representa um princípio ético fundamental na engenharia de software?

(a) Priorizar o desenvolvimento de software de código aberto em detrimento de software proprietário.
(b) Assegurar que o software seja eficiente em termos de uso de recursos computacionais.
(c) Garantir que o software não cause dano a indivíduos, organizações ou à sociedade como um todo.
(d) Focar no desenvolvimento rápido de software para atender às demandas do mercado.
(e) Utilizar apenas linguagens de programação modernas e atualizadas no desenvolvimento de software.

8. Qual das seguintes situações melhor justifica a escolha de uma API RESTful em vez de um Web Service baseado em SOAP para integração entre sistemas?

(a) Quando é necessário suportar transações complexas e atômicas entre sistemas distribuídos.
(b) Em cenários onde a interoperabilidade com sistemas legados é uma prioridade.
(c) Quando é crucial minimizar a sobrecarga de dados nas comunicações entre cliente e servidor.
(d) Em ambientes onde a segurança de nível militar é obrigatória para todas as transações.
(e) Quando há a necessidade de suportar comunicação síncrona entre sistemas em tempo real.

9. A partir do conceito de escalabilidade horizontal e vertical na computação em nuvem, selecione a alternativa que melhor caracteriza as situações em que cada abordagem seria mais adequada.

(a) A escalabilidade horizontal envolve adicionar mais recursos computacionais, como servidores virtuais, para lidar com um aumento na demanda, enquanto a escalabilidade vertical envolve aumentar a capacidade dos recursos existentes, como aumentar a quantidade de RAM ou CPU em um servidor. A escalabilidade horizontal é mais adequada para cargas de trabalho que podem ser facilmente distribuídas em vários

servidores, enquanto a escalabilidade vertical é mais adequada para cargas de trabalho que exigem recursos adicionais em um único servidor, como bancos de dados.

(b) A escalabilidade horizontal envolve aumentar a capacidade de processamento de um único servidor, enquanto a escalabilidade vertical envolve adicionar mais servidores para lidar com um aumento na demanda. A escalabilidade horizontal é mais adequada para cargas de trabalho que podem ser facilmente paralelizadas e distribuídas entre vários servidores, enquanto a escalabilidade vertical é mais adequada para cargas de trabalho que exigem recursos adicionais em um único servidor, como processamento intensivo.

(c) A escalabilidade horizontal envolve adicionar mais servidores para lidar com um aumento na demanda, enquanto a escalabilidade vertical envolve aumentar a capacidade dos servidores existentes para lidar com uma carga de trabalho maior. A escalabilidade horizontal é mais adequada para cargas de trabalho que podem ser facilmente distribuídas entre vários servidores, enquanto a escalabilidade vertical é mais adequada para cargas de trabalho que exigem recursos adicionais em um único servidor, como armazenamento de dados.

(d) A escalabilidade horizontal envolve aumentar a capacidade de armazenamento de um servidor, enquanto a escalabilidade vertical envolve adicionar mais servidores para lidar com um aumento na demanda. A escalabilidade horizontal é mais adequada para cargas de trabalho que exigem armazenamento adicional, como aplicativos de análise de big data, enquanto a escalabilidade vertical é mais adequada para cargas de trabalho que exigem mais poder de processamento, como aplicativos de computação intensiva.

(e) A escalabilidade horizontal envolve aumentar a capacidade de armazenamento de um único servidor, enquanto a escalabilidade vertical envolve adicionar mais servidores para lidar com um aumento na demanda. A escalabilidade horizontal é mais adequada para cargas de trabalho que podem ser facilmente distribuídas entre vários servidores, enquanto a escalabilidade vertical é mais adequada para cargas de trabalho que exigem recursos adicionais em um único servidor, como aplicativos de banco de dados.

10. Quando se compara aplicações nativas com aplicações híbridas no desenvolvimento móvel, qual das seguintes afirmações é verdadeira?

(a) Aplicações nativas são desenvolvidas com linguagens específicas de plataforma, como Swift para iOS e Kotlin para Android, enquanto aplicações híbridas são desenvolvidas com linguagens web como HTML, CSS e JavaScript.

(b) Aplicações híbridas têm acesso total a todos os recursos do dispositivo móvel, enquanto aplicações nativas têm acesso limitado.

(c) Aplicações nativas são executadas no navegador do dispositivo, enquanto aplicações híbridas são executadas diretamente no sistema operacional do dispositivo.

(d) O custo de desenvolvimento e manutenção é significativamente maior para aplicações híbridas do que para aplicações nativas.

(e) Aplicações híbridas são mais adequadas para jogos de alta performance e aplicações que exigem intensivo processamento gráfico.

11. Selecione a alternativa quue melhor explica a diferença entre autenticação e autorização no contexto da segurança de sistemas de informação. E que, além disso, discutee a importância de implementar ambos os mecanismos em um sistema para garantir a proteção adequada dos dados.

(a) Autenticação refere-se ao processo de verificar a identidade de um usuário, geralmente por meio de credenciais como nome de usuário e senha, enquanto autorização diz respeito às permissões concedidas a um usuário para acessar recursos específicos após ter sua identidade verificada. Implementar ambos os mecanismos é importante para garantir que apenas usuários legítimos tenham acesso aos dados e recursos do sistema, e que esses usuários tenham apenas as permissões necessárias para realizar suas tarefas, reduzindo assim o risco de acessos não autorizados e violações de segurança.

(b) Autenticação refere-se ao processo de verificar a integridade e autenticidade dos dados transmitidos entre sistemas, enquanto autorização diz respeito à atribuição de níveis de acesso a usuários com base em suas credenciais de autenticação. Implementar ambos

os mecanismos é importante para garantir que os dados transmitidos sejam protegidos contra adulteração ou interceptação não autorizada, e que apenas usuários autorizados tenham acesso aos recursos do sistema, prevenindo assim violações de segurança e vazamento de informações confidenciais.

(c) Autenticação refere-se ao processo de atribuir permissões específicas a usuários com base em suas credenciais de login, enquanto autorização diz respeito à verificação da identidade de um usuário durante o acesso a recursos do sistema. Implementar ambos os mecanismos é importante para garantir que apenas usuários legítimos tenham acesso aos dados e recursos do sistema, e que esses usuários tenham apenas as permissões necessárias para realizar suas tarefas, reduzindo assim o risco de acessos não autorizados e violações de segurança.

(d) Autenticação refere-se ao processo de garantir a confidencialidade dos dados durante sua transmissão entre sistemas, enquanto autorização diz respeito à verificação da identidade de um usuário durante o acesso a recursos do sistema. Implementar ambos os mecanismos é importante para garantir que apenas usuários legítimos tenham acesso aos dados e recursos do sistema, e que esses usuários tenham apenas as permissões necessárias para realizar suas tarefas, reduzindo assim o risco de acessos não autorizados e violações de segurança.

(e) Autenticação refere-se ao processo de verificar a integridade e autenticidade dos dados transmitidos entre sistemas, enquanto autorização diz respeito à verificação da identidade de um usuário durante o acesso a recursos do sistema. Implementar ambos os mecanismos é importante para garantir que os dados transmitidos sejam protegidos contra adulteração ou interceptação não autorizada, e que apenas usuários autorizados tenham acesso aos recursos do sistema, prevenindo assim violações de segurança e vazamento de informações confidenciais.

12. Qual característica NÃO é típica de uma API RESTful?

(a) Uso de métodos HTTP como GET, POST, PUT e DELETE.
(b) Manutenção de estado entre requisições subsequentes.
(c) Formato de troca de dados baseado em JSON ou XML.
(d) Uso de URLs para identificar recursos.
(e) Orientação a recursos, onde cada recurso é identificado por um URI.

13. Na integração de sistemas, qual das seguintes técnicas é frequentemente utilizada para permitir que diferentes sistemas de software comuniquem entre si de forma eficaz?

(a) Uso exclusivo de linguagens de programação de alto nível.
(b) Implementação de interfaces gráficas avançadas para os usuários finais.
(c) Emprego de APIs (Application Programming Interfaces) para estabelecer conexões entre sistemas.
(d) Centralização de todos os dados em um único banco de dados físico.
(e) Utilização de algoritmos de aprendizado de máquina para prever interações entre sistemas.

14. No contexto do uso do Git como sistema de controle de versão, o que acontece quando um desenvolvedor executa o comando git merge?

(a) O comando cria uma nova branch baseada na branch atual.
(b) O comando compara e combina as mudanças em diferentes branches.
(c) O comando exclui a branch especificada e todas as suas alterações.
(d) O comando reverte o repositório para o estado de um commit anterior.
(e) O comando clona um repositório remoto para o sistema local.

15. Com relação aos os desafios relacionados à segurança da informação na computação em nuvem e as respectivas estratégias eficazes para mitigar esses desafios, selecione a alternativa que apresenta os melhores argmentos.

Boa sorte!

(a) Os desafios incluem a perda de controle sobre os dados e recursos de segurança ao migrar para ambientes de nuvem, preocupações com a conformidade regulatória e a necessidade de proteger os dados contra acessos não autorizados. Estratégias eficazes incluem a criptografia de dados em repouso e em trânsito, o uso de autenticação de vários fatores, a implementação de controles de acesso granulares e a realização de auditorias regulares de segurança.
(b) Os desafios incluem a perda de visibilidade sobre a infraestrutura de segurança ao migrar para a nuvem, a dificuldade de identificar e responder a ameaças em ambientes dinâmicos de nuvem e a falta de controle sobre as políticas de segurança implementadas pelo provedor de nuvem. Estratégias eficazes incluem a implementação de ferramentas de monitoramento de segurança em tempo real, a adoção de políticas de segurança baseadas em princípios de "defesa em profundidade" e a realização de testes regulares de penetração e vulnerabilidade.
(c) Os desafios incluem a dependência de provedores de nuvem terceirizados para proteger os dados e recursos de segurança, a dificuldade de garantir a integridade e a confidencialidade dos dados em ambientes compartilhados e a necessidade de garantir a conformidade com regulamentações de segurança e privacidade. Estratégias eficazes incluem a implementação de controles de acesso rigorosos, a segregação de dados sensíveis em ambientes dedicados e a realização de avaliações de risco regulares para identificar e mitigar vulnerabilidades.
(d) Os desafios incluem a complexidade de gerenciar e manter a segurança em ambientes de nuvem distribuídos e heterogêneos, a falta de visibilidade sobre as práticas de segurança do provedor de nuvem e a necessidade de garantir a segurança dos dados durante todo o ciclo de vida, desde a criação até a exclusão. Estratégias eficazes incluem a implementação de políticas de governança de segurança abrangentes, o uso de ferramentas de automação de segurança para garantir conformidade contínua e a realização de auditorias independentes de segurança.
(e) Os desafios incluem a falta de controle sobre a infraestrutura de segurança em ambientes de nuvem pública, a dificuldade de proteger os dados contra ameaças internas e externas e a necessidade de garantir a conformidade com regulamentações de segurança e privacidade. Estratégias eficazes incluem a implementação de firewalls avançados e soluções de detecção de intrusões, a criptografia de dados sensíveis e a implementação de políticas de segurança que enfatizam a educação e a conscientização dos usuários.

16. No contexto da Lei Geral de Proteção de Dados (LGPD), qual das seguintes afirmações melhor descreve uma responsabilidade ética e legal dos engenheiros de software ao desenvolver sistemas que processam dados pessoais?

(a) A LGPD exige que todos os sistemas de informação armazenem os dados pessoais dos usuários por pelo menos 10 anos para fins de auditoria.
(b) Os engenheiros de software devem assegurar que o sistema permita aos usuários acessar e corrigir seus próprios dados pessoais, em conformidade com os princípios de transparência e acesso da LGPD.
(c) Sob a LGPD, é permitido compartilhar dados pessoais com terceiros sem consentimento explícito, desde que os dados sejam anonimizados.
(d) A LGPD se aplica exclusivamente a dados coletados digitalmente, não sendo necessária a preocupação com dados coletados por meios não digitais.
(e) Segundo a LGPD, é obrigatório criptografar todos os dados pessoais, independentemente de seu nível de sensibilidade.

17. Qual das seguintes afirmações melhor descreve a análise estática de código?

(a) É o processo de testar o código em um ambiente de produção para encontrar erros de runtime.
(b) Envolve a revisão manual do código fonte para identificar padrões de codificação inadequados.

(c) Consiste em executar o programa em um ambiente controlado para detectar falhas de segurança.
(d) Refere-se à verificação automática do código fonte para identificar erros e vulnerabilidades sem executá-lo.
(e) É a prática de usar ferramentas de automação para melhorar a eficiência do processo de compilação do código.

18. Considerando os princípios de uma API RESTful, qual das seguintes opções representa uma violação desses princípios?

(a) Utilizar URIs para representar recursos, como /clientes/123.
(b) Empregar métodos HTTP como GET para leitura e POST para criação de recursos.
(c) Enviar informações de estado do cliente na URL para manter uma sessão entre requisições.
(d) Retornar dados em diferentes formatos (como JSON ou XML) com base no cabeçalho de aceitação da requisição.
(e) Usar códigos de status HTTP apropriados para representar o resultado das operações (por exemplo, 200 para sucesso, 404 para não encontrado).

19. Com relação aos principais desafios enfrentados no desenvolvimento de aplicativos móveis híbridos em comparação com aplicativos móveis nativos e as estratégias ou técnicas que podem ser adotadas para mitigar esses desafios e melhorar a qualidade e o desempenho dos aplicativos híbridos, selecione a alternativa que apresenta os melhores argumentos.

(a) Os principais desafios no desenvolvimento de aplicativos móveis híbridos incluem desempenho inferior em comparação com aplicativos nativos, acesso limitado aos recursos do dispositivo e complexidade na integração com funcionalidades nativas. Para mitigar esses desafios, é importante otimizar o código JavaScript, utilizar bibliotecas e frameworks de alto desempenho, como React Native ou Ionic, e implementar plugins nativos para acessar recursos específicos do dispositivo. Além disso, testes rigorosos e monitoramento de desempenho são essenciais para identificar e corrigir problemas de forma proativa.
(b) Os principais desafios no desenvolvimento de aplicativos móveis híbridos incluem desempenho inferior em comparação com aplicativos nativos, acesso limitado aos recursos do dispositivo e complexidade na integração com funcionalidades nativas. Para mitigar esses desafios, é importante adotar abordagens de caching eficientes, usar técnicas de pré-processamento de dados e implementar estratégias de otimização de rede. Além disso, é fundamental realizar testes de desempenho em diferentes dispositivos e condições de rede, visando identificar e corrigir gargalos de desempenho.
(c) Os principais desafios no desenvolvimento de aplicativos móveis híbridos incluem desempenho inferior em comparação com aplicativos nativos, acesso limitado aos recursos do dispositivo e complexidade na integração com funcionalidades nativas. Para mitigar esses desafios, é importante investir em capacitação e treinamento da equipe de desenvolvimento, adotar boas práticas de arquitetura de software e design de código limpo, e utilizar ferramentas de análise estática de código para identificar e corrigir problemas de qualidade de código. Além disso, é fundamental realizar testes abrangentes em diferentes dispositivos e cenários de uso para garantir a estabilidade e a confiabilidade do aplicativo.
(d) Os principais desafios no desenvolvimento de aplicativos móveis híbridos incluem desempenho inferior em comparação com aplicativos nativos, acesso limitado aos recursos do dispositivo e complexidade na integração com funcionalidades nativas. Para mitigar esses desafios, é importante investir em estratégias de cache eficientes, adotar técnicas de pré-renderização de interfaces de usuário e implementar armazenamento local de dados sempre que possível. Além disso, a utilização de técnicas de compilação ahead-of-time (AOT) e just-in-time (JIT) pode melhorar o desempenho do aplicativo híbrido, reduzindo o tempo de carregamento e a latência durante a execução.

Boa sorte!

(e) Os principais desafios no desenvolvimento de aplicativos móveis híbridos incluem desempenho inferior em comparação com aplicativos nativos, acesso limitado aos recursos do dispositivo e complexidade na integração com funcionalidades nativas. Para mitigar esses desafios, é importante otimizar o código JavaScript, utilizar bibliotecas e frameworks de alto desempenho, como React Native ou Ionic, e implementar plugins nativos para acessar recursos específicos do dispositivo. Além disso, a utilização de ferramentas de análise estática de código pode ajudar a identificar e corrigir problemas de qualidade de código, enquanto testes automatizados podem garantir a funcionalidade e a estabilidade do aplicativo em diferentes dispositivos e cenários de uso.

20. Considere um cenário onde uma empresa precisa integrar seu sistema de gerenciamento de estoque (baseado em um banco de dados relacional) com uma nova plataforma de e-commerce (que utiliza um banco de dados não relacional). Qual das seguintes abordagens apresenta o maior desafio técnico para a integração desses sistemas heterogêneos?

(a) Utilização de uma API RESTful para comunicação entre os dois sistemas, assegurando a transferência de dados em formato JSON.
(b) Implementação de um serviço de mensageria para garantir a entrega assíncrona de dados entre os sistemas.
(c) Criação de um middleware de conversão de dados que mapeia e transforma dados entre os modelos relacional e não relacional.
(d) Emprego de um padrão de design de microserviços para modularizar cada componente do sistema.
(e) Configuração de um gateway de API para gerenciar o tráfego e as solicitações entre os sistemas.

Bem-vindo à seção da questão extra (opcional) da prova. Esta questão oferece a oportunidade de ganhar até 1 ponto adicional, caso você opte por respondê-la.
Por favor, leia o enunciado com atenção e, se desejar, responda à questão proposta. Lembrese de que esta questão é completamente opcional e não afetará sua pontuação caso decida não respondê-la.

Certifique-se de considerar cuidadosamente se deseja ou não tentar responder à questão extra, levando em conta o tempo disponível e sua confiança na resposta.

Boa sorte!

21. Como a arquitetura de software e o design de software se inter-relacionam e como a mudança em um pode afetar o outro? Quais os desafios e como superá-los, podem dar exemplos práticos?