INOVATECH
RESIDÊNCIA EM INOVAÇÃO TECNOLÓGICA

PROVA OBJETIVA

10 de abril de 2026.
Campus Teresina Central do Instituto Federal do Piauí - IFPI

INSTRUÇÕES GERAIS

1. **VERIFICAÇÃO**: Verifique se este caderno contém 30 (trinta) questões de múltipla escolha, cada uma com alternativas de A a E. Caso o caderno esteja incompleto ou apresente defeitos, solicite a substituição imediata ao fiscal.
2. **DURAÇÃO**: Você dispõe de **2 (duas) horas** para realizar a prova, incluindo o preenchimento do Cartão-Resposta.
3. **MATERIAL**: Utilize apenas caneta esferográfica de tinta azul ou preta. Não é permitido o uso de corretivos, lápis ou lapiseiras no Cartão-Resposta.

INSTRUÇÕES PARA O PREENCHIMENTO DO CARTÃO-RESPOSTA

O seu Cartão-Resposta é o único documento válido para a correção. Ele será processado por sistemas automáticos de OCR (Reconhecimento Óptico de Caracteres) e OMR (Reconhecimento Óptico de Marcas). Para garantir a sua pontuação, **siga rigorosamente as orientações abaixo**:

- **PREENCHIMENTO DA BOLHA**: Preencha totalmente o espaço circular correspondente à alternativa escolhida. Não marque apenas um "X" ou um traço; a bolha deve estar completamente preenchida e escura.
- **RESPOSTA ÚNICA:** Cada questão possui apenas uma única resposta correta.
- **ERROS E RASURAS:** Não serão computadas questões com emendas, rasuras ou preenchimento incompleto.
- **MARCAÇÕES DUPLAS (ATENÇÃO):** Caso o candidato preencha mais de uma alternativa para a mesma questão, esta ou será considerada nula ou poderá ser atribuída a ela qualquer uma das alternativas marcadas, independentemente de qualquer observação, seta ou indicação manual feita pelo candidato sobre qual opção ele prefere que seja considerada.
- **PROCESSAMENTO AUTOMÁTICO:** A leitura e correção são realizadas de maneira totalmente automatizada, sem intervenção humana. O sistema lerá apenas o que estiver assinalado no campo de marcação.
- **CUIDADO COM O CARTÃO:** Não dobre, não amasse, não molhe e não suje o seu Cartão-Resposta. Ele não será substituído por erro de preenchimento ou manuseio indevido por parte do candidato.

BOA PROVA!

piauígovtech
INSTITUTO FEDERAL Piauí
FAIFPI FUNDAÇÃO DE APOIO

QUESTÃO 01 Imagine o seguinte cenário hipotético:

Uma equipe está desenvolvendo uma plataforma para Secretaria de Saúde com objetivo coletar dados de pesquisas de campo sobre a saúde de trabalhadores em áreas rurais do estado. Agentes de saúde usam tablets para preencher formulários offline. Ao final do dia, com conexão à internet, o tablet compila as respostas e gera um documento PDF único e criptografado para cada trabalhador pesquisado.

Considerando que haverá milhares de agentes realizando o *upload* de milhares de PDFs simultaneamente, a empresa precisa desenhar uma integração com o serviço web de *Storage* (Armazenamento de Objetos, ex: AWS S3 ou Azure Blob Storage) que seja escalável (evitando gargalos no servidor de *back-end* com tráfego de arquivos) e segura (evitando expor credenciais permanentes de acesso ao *Storage* no aplicativo móvel).

Para resolver este desafio operacional de forma eficiente e segura, qual padrão de integração com o serviço web de *Storage* deve ser aplicado?

A) O tablet envia o PDF via requisição HTTP POST para uma API REST no *back-end* da PiauiGovTech. O *back-end* recebe o stream do arquivo e usa suas credenciais IAM (Access Key e Secret Key) permanentes para realizar o *upload* para o *bucket* de *Storage*.

B) O tablet envia apenas os metadados do arquivo (ex: tamanho, nome) para o *back-end* da PiauiGovTech. O *back-end* gera uma *Pre-signed URL* (URL pré-assinada) de curta duração via SDK do *Storage* e a retorna para o tablet. O tablet então realiza o *upload* do PDF diretamente para o serviço de *Storage* usando essa URL.

C) Uma Chave de Acesso IAM (Access Key e Secret Key) com permissões de escrita no *bucket* é embutida no código-fonte do aplicativo móvel, permitindo que o tablet autentique diretamente com o SDK do *Storage* para realizar o *upload* do PDF.

D) O *back-end* da PiauiGovTech recebe o PDF em formato Base64 via JSON em uma requisição HTTP POST e o armazena temporariamente em uma tabela do banco de dados relacional principal usando o tipo de dados BLOB (Binary Large Object), movendo-o para o *Storage* posteriormente via *cron job*.

E) O tablet realiza um *upload* via SFTP (Secure File Transfer Protocol) para uma instância de computação (Virtual Machine) dedicada na nuvem que atua como um *gateway* de arquivos e, em seguida, sincroniza os dados com o serviço de *Storage* via Rsync.

QUESTÃO 02 Imagine o seguinte cenário hipotético:

No contexto do projeto hipotético "Sistema Integrado de Resposta a Incidentes do Piauí" (SIRI-PI) você está desenhando um subsistema de 'Trânsito e Segurança Pública'. O objetivo é que, ao criar um "Boletim de Ocorrência de Trânsito" (B.O.) no subsistema de Segurança, o sistema realize as seguintes ações:

1. Salvar o B.O. em um banco de dados PostgreSQL interno.
2. Enviar um evento assíncrono para o subsistema de 'Atendimento de Saúde' (ex: via RabbitMQ).
3. Consultar uma API REST de câmeras de tráfego de um município parceiro para anexar dados contextuais ao B.O.

Adotando Clean Architecture, SOLID (especialmente o DIP - Princípio da Inversão de Dependência) e DDD, você precisa organizar o software em camadas com separação clara de responsabilidades, garantindo que o núcleo da aplicação (Domínio e Use Cases) permaneça agnóstico a detalhes técnicos de banco de dados, mensageria ou serviços web externos.

Ao organizar os componentes do software, como as abstrações (Portas) e as implementações (Adaptadores) para esses três serviços externos (PostgreSQL, RabbitMQ e API Municipal REST) devem ser distribuídas entre as camadas, considerando o fluxo de controle de dentro para fora?

A) Definir as interfaces (Portas) na camada de Domínio, e as classes de implementação (Adaptadores) concretas na camada de Frameworks/Drivers, fazendo com que o Use Case invoque as implementações diretamente para maior eficiência.

B) Definir as interfaces (Portas) na camada de Frameworks/Drivers, e as classes de implementação (Adaptadores) concretas na camada de Interface Adapters, fazendo com que a Aplicação dependa dos detalhes de conexão.

C) Definir as interfaces (Portas) na camada de Domínio ou Aplicação, e as classes de implementação (Adaptadores) concretas na camada de Frameworks/Drivers, garantindo que a Aplicação dependa apenas das abstrações e que o fluxo de dependência flua para dentro.

D) Criar uma única classe de serviço utilitária (IntegrationManager) na camada de Aplicação que centralize as instâncias concretas do PostgreSQL, RabbitMQ e da API REST para realizar todas as integrações em um único local, respeitando o SRP (SOLID).

E) Definir as interfaces (Portas) na camada de Interface Adapters, e as classes de implementação (Adaptadores) concretas na camada de Frameworks/Drivers, garantindo que o Domínio invoque as portas via Controllers para orquestração.

. QUESTÃO 03 Imagine o seguinte cenário hipotético:

No âmbito do projeto "Pro-Técnico", um sistema de seleção em larga escala gerenciado para avaliar alunos de cursos técnicos em todo o estado, você está realizando uma análise de segurança web com base no OWASP Top 10. O sistema permite que candidatos autenticados façam o *download* de seus relatórios de avaliação individuais em formato PDF.

Ao testar a funcionalidade de *download*, você descobre que a URL utilizada é estruturada da seguinte forma: https://pro-tecnico.pi.gov.br/relatorios/download?aluno_id=12345. Você, logado com o perfil de candidato autenticado, constata que ao alterar o parâmetro aluno_id=12345 para aluno_id=12346 (o ID do próximo aluno), o sistema exibe o relatório de avaliação do outro candidato, sem verificar se o usuário logado tem permissão para acessar aquele recurso específico.

Considerando o OWASP Top 10, qual abordagem deve ser aplicada para mitigar permanentemente esta vulnerabilidade de Controle de Acesso Quebrado (*Broken Access Control*)?

A) Implementar criptografia simétrica forte (como AES-256) para proteger todos os dados pessoais sensíveis dos candidatos armazenados no banco de dados e criptografar o parâmetro aluno_id na URL utilizando uma chave secreta exclusiva por sessão do usuário.

B) Configurar um *middleware* de segurança no back-end que valide a integridade do parâmetro aluno_id utilizando uma soma de verificação (hash) gerada no *client-side* (navegador do aluno) e comparada no servidor antes de disponibilizar o arquivo para *download*.

C) Adotar o padrão de IDs não sequenciais e criptograficamente fortes (como UUID v4) para todos os recursos e desabilitar o método HTTP POST na rota de *download* para evitar ataques de força bruta (*brute force*).

D) Implementar o controle de acesso baseado em atributos (ABAC) ou baseado em funções (RBAC) no servidor, onde o back-end verifica explicitamente se o aluno_id extraído do *token* de autenticação na sessão do usuário corresponde ao aluno_id solicitado na URL antes de servir o recurso protegido.

E) Substituir o método HTTP GET pelo método HTTP POST na rota de *download*, passando o parâmetro aluno_id no corpo da requisição (*body payload*), o que ocultará o ID da URL e impedirá a exploração direta através da barra de endereços do navegador.

. QUESTÃO 04 Imagine o seguinte cenário hipotético:

A equipe de desenvolvimento da PiauiGovTech está aprimorando seu fluxo de DevOps para entregar serviços públicos de forma ágil e contínua para o "Portal de Serviços do Estado". Eles decidiram centralizar seus códigos no GitHub, utilizando o GitHub Actions para automação de CI/CD, e Docker/Docker Compose para conteinerização e definição de orquestração local/homologação.

Um desenvolvedor finalizou a implementação de uma nova funcionalidade e abriu um Pull Request. O objetivo é garantir que, antes de realizar o *merge* para a branch principal (main), o código passe por testes e, *imediatamente após o merge*, uma imagem Docker imutável seja gerada, enviada para um *registry* e implantada automaticamente no ambiente de "Homologação" para validação final.

Considerando as boas práticas, padrão ouro, de GitHub Actions e modelos de workflow do Git, qual sequência de ações e configurações de workflow deve ser utilizada para atingir esse objetivo de entrega contínua (CD) de forma robusta, eficiente e segura?

A) Utilizar o modelo GitHub Flow; desenvolver na branch feature/nova-funcionalidade e abrir Pull Request para a branch main. Criar um workflow no .github/workflows/ci-cd.yml configurado para disparar on: push em branches de feature. As etapas devem ser: Stage 1 (Build da imagem Docker) e Stage 2 (Deploy remoto no ambiente de Homologação via SSH usando Docker Compose), pulando testes automatizados para acelerar o *time-to-market*.

B) Utilizar o modelo Gitflow workflow; desenvolver na branch feature/nova-funcionalidade, realizar o merge para a branch develop. Após testes manuais em develop, abrir Pull Request para a branch master. Criar um workflow configurado para disparar on: push apenas na branch master. As etapas devem ser: Stage 1 (Build da imagem Docker), Stage 2 (Testes automatizados conteinerizados), Stage 3 (Push para o GitHub Container Registry - ghcr.io) e Stage 4 (Deploy remoto via SSH usando Docker Compose).

C) Utilizar o modelo GitHub Flow workflow; desenvolver na branch feature/nova-funcionalidade, abrir Pull Request para a branch

main. Criar dois workflows distintos no GitHub Actions: 1. Workflow A (on: pull_request para main): realiza o Stage 1 (Build da imagem Docker) e Stage 2 (Execução de testes automatizados conteinerizados). 2. Workflow B (on: push para main - acionado após o merge): realiza o Stage 1 (Build da imagem final), Stage 2 (Geração de tag imutável e Push para o GitHub Container Registry - ghcr.io) e Stage 3 (Deploy remoto via SSH atualizando o ambiente de Homologação com Docker Compose).

D) Utilizar o modelo Trunk-Based Development ou GitHub Flow; desenvolver na branch feature/nova-funcionalidade, realizar commits frequentes. Criar um único workflow configurado para disparar on: push em qualquer branch (feature ou main). As etapas devem ser: Stage 1 (Build da imagem Docker), Stage 2 (Geração de tag imutável e Push para o GitHub Container Registry - ghcr.io), Stage 3 (Testes automatizados conteinerizados) e Stage 4 (Deploy remoto automático em Homologação imediatamente após o push na branch de feature), sem necessidade de Pull Request.

E) Utilizar o modelo Gitflow; desenvolver na branch develop e realizar commits diretamente na branch main para evitar conflitos de merge. Criar um workflow configurado com on: schedule para rodar a cada 1 hora. As etapas devem ser: Stage 1 (Execução de testes unitários no ambiente local do runner), Stage 2 (Deploy via SSH no servidor de Homologação copiando o código fonte diretamente) e Stage 3 (Build da imagem Docker no servidor de destino para economizar largura de banda de rede), utilizando variáveis de ambiente de produção expostas diretamente no arquivo YAML para facilitar o debug.

. QUESTÃO 05 Imagine o seguinte cenário hipotético:

Você é o desenvolvedor líder responsável por criar o aplicativo móvel da "Loteria do Estadual". O aplicativo deve permitir que usuários registrados visualizem o saldo de sua carteira, realizem apostas e consultem o histórico de transações. O sistema opera sob uma premissa rigorosa de segurança e deve fornecer uma experiência de usuário fluida, mesmo em áreas rurais do estado com conectividade de rede intermitente (padrão offline-first).

Um apostador autenticado tenta consultar seu histórico de transações financeiras enquanto está sem conexão com a internet. O aplicativo deve exibir os dados armazenados localmente no dispositivo de forma segura. Ao recuperar a conectividade, o aplicativo deve, de forma transparente, sincronizar quaisquer novas apostas realizadas localmente com o servidor central, garantindo a integridade dos dados e a consistência do saldo.

Considerando o uso de React Native (com Redux e Async Storage) ou Flutter (com BLoC e Hive/Flutter Secure Storage) para o desenvolvimento, qual conjunto de implementações técnicas deve ser aplicado para garantir o funcionamento offline-first e a segurança dos dados financeiros?

A) Utilizar o componente nativo de rede (como NetInfo) para verificar a conectividade antes de cada requisição RESTful. Se a rede estiver indisponível, exibir uma mensagem de erro genérica instruindo o usuário a tentar novamente quando tiver conexão, impedindo a visualização do histórico e a realização de apostas offline.

B) No React Native, armazenar o histórico de transações em JSON bruto no Async Storage, confiando na criptografia padrão do sistema operacional. Utilizar o Redux para gerenciar o estado global e, para apostas offline, enfileirar as ações em memória para despacho imediato assim que a conexão retornar.

C) Implementar banco de dados local criptografado (como SQLite/SQLCipher no React Native ou Hive criptografado no Flutter) gerenciado por BLoC/Redux. Criar uma fila de sincronização persistente para apostas offline, utilizando chaves nativas (Keystore/Keychain) para garantir a segurança dos dados antes do envio ao servidor.

D) No Flutter, utilizar o Hive como banco de dados NoSQL local não criptografado para armazenar o histórico de transações, aproveitando a tipagem estrita. Gerenciar o estado com Provider e confiar na validação de balanceamento feita exclusivamente pelo servidor central durante a sincronização pós-conexão para resolver conflitos.

E) No Flutter, utilizar o pacote shared_preferences para armazenar o histórico de transações e a fila de apostas offline em formato JSON, criptografando os valores individualmente com uma chave fixa embutida no código da aplicação. Utilizar setState nos widgets principais para gerenciar o estado offline e garantir a atualização da UI.

. QUESTÃO 06 Imagine o seguinte cenário hipotético:

A SEFAZ está projetando um painel de monitoramento fiscal com React e TypeScript que exibe uma tabela com até 5.000 linhas de notas fiscais, atualizada a cada 30 segundos via WebSocket. Cada linha possui um botão de ação que abre um modal de detalhamento. Os requisitos do projeto estabelecem:

1. A tabela deve ser acessível para leitores de tela (WCAG 2.1 AA).
2. A interface não pode travar durante as atualizações em tempo real.
3. O código deve ser sustentável para uma equipe de seis desenvolvedores.

A equipe propõe a seguinte arquitetura inicial do componente principal:

```
function FiscalDashboard() {
  const [invoices, setInvoices] =
useState<Invoice[]>([]);

  useEffect(() => {
    const ws = new
WebSocket("wss://api.sefaz.pi.gov.br/invoic
es");
    ws.onmessage = e =>
setInvoices(JSON.parse(e.data));
    return () => ws.close();
  }, []);

  return (
    <div>
      {invoices.map(invoice => (
        <TableRow key={invoice.id}
invoice={invoice} />
      ))}
    </div>
  );
}
```

O arquiteto de front-end aponta que essa implementação, se levada à produção, viola simultaneamente os três requisitos do projeto.

Considerando os três requisitos estabelecidos e as limitações identificadas na arquitetura proposta, o conjunto de decisões técnicas que resolve de forma completa os problemas apontados é:

A) Substituir o por <table> com <thead> e <tbody> semânticos, aplicar React.memo no componente LinhaTabela para evitar re-renderizações em massa, e mover a lógica do WebSocket para um custom hook useNotasWebSocket reutilizável pela equipe.

B) Implementar virtualização de lista com uma biblioteca como react-window, aplicar useCallback no handler do WebSocket para estabilizar a referência da função e adicionar atributos aria-label no contêiner da tabela para atender à leitura por tecnologias assistivas.

C) Substituir o useState por useReducer para controlar as atualizações do WebSocket, envolver o mapa de linhas em React.memo e usar elementos <section> com aria-labelledby para estruturar semanticamente o painel para leitores de tela.

D) Adotar virtualização de lista para renderizar apenas as linhas visíveis, substituir o por elementos <table> semânticos com atributos ARIA adequados e extrair o WebSocket e o estado derivado para um custom hook isolado, separando responsabilidades para a equipe.

E) Aplicar useDeferredValue no estado notas para adiar atualizações de baixa prioridade durante a renderização, usar <ol> com role="row" nas linhas para estrutura semântica, e encapsular o WebSocket em um contexto global compartilhado entre os componentes do painel.

. QUESTÃO 07 Imagine o seguinte cenário hipotético:

O DETRAN está construindo um sistema de consulta de habilitações com React e TypeScript. O sistema possui três módulos principais: ConsultaCNH, HistoricoInfrações e AgendamentoVistoria. Os três módulos precisam acessar e modificar os dados do condutor autenticado. A equipe avalia as seguintes estratégias para compartilhar esse estado entre os módulos sem criar acoplamento excessivo:

- Estratégia A: Elevar o estado para o componente raiz e passá-lo via props por todos os níveis da árvore até os módulos consumidores.
- Estratégia B: Criar um contexto React com useContext e um reducer com useReducer, expondo estado e dispatch por um Provider único no nível da aplicação.
- Estratégia C: Adotar uma biblioteca de estado global como Zustand ou Redux Toolkit, definindo uma store centralizada com seletores por módulo.
- Estratégia D: Utilizar localStorage com eventos customizados para sincronizar o estado entre os módulos de forma reativa, sem acoplamento via árvore de componentes.

A equipe precisa escolher a estratégia que equilibra a rastreabilidade das mutações de estado, escalabilidade para novos módulos e manutenibilidade do código a longo prazo.

Considerando os critérios técnicos estabelecidos pela equipe, a avaliação precisa das estratégias propostas é:

A) A Estratégia B é suficiente para o escopo descrito, pois o useReducer centraliza as mutações e o useContext elimina o prop drilling, sendo as Estratégias C e D excessivas para uma aplicação com três módulos.

B) A Estratégia C oferece a melhor relação entre rastreabilidade e escalabilidade, pois seletores por módulo evitam re-renderizações desnecessárias que o useContext provoca ao atualizar todos os consumidores do Provider.

C) A Estratégia A é a mais rastreável porque o fluxo unidirecional de props torna explícitas todas as dependências de dados, sendo as demais estratégias indicadas apenas quando há mais de dez módulos na aplicação.

D) A Estratégia D desacopla completamente os módulos da árvore React e é a mais escalável, pois eventos customizados permitem adicionar novos módulos consumidores sem alterar o Provider ou a store existente.

E) As Estratégias B e C são equivalentes em rastreabilidade e escalabilidade, diferindo apenas na curva de aprendizado, de modo que a escolha deve ser baseada exclusivamente na familiaridade da equipe com cada ferramenta.

. QUESTÃO 08 Imagine o seguinte cenário hipotético:

A equipe de desenvolvimento do portal do servidor público do Governo do Estado do Piauí está refatorando um componente legado de classe para um componente funcional com React e TypeScript. O componente original buscava dados de uma API de contracheques ao ser montado e cancelava a requisição ao ser desmontado, evitando atualizações de estado em componentes já removidos da árvore.

Durante a refatoração, a equipe produziu o seguinte código:

```
function Payslip({ employeeId }: {
employeeId: string }) {
  const [data, setData] = useState<Payment
| null>(null);

  useEffect(() => {
    fetch(`/api/payslip/${employeeId}`)
      .then(res => res.json())
      .then(payload => setData(payload));
  }, [employeeId]);

  return <PaymentViewer data={data} />;
}
```

A liderança técnica aponta que o código não reproduz corretamente o comportamento de cancelamento da versão legada, expondo o sistema a um vazamento de estado em navegações rápidas entre matrículas.

Considerando o problema identificado, a correção a ser aplicada para o useEffect desse componente é:

A) Retornar uma função de limpeza no useEffect que utilize um sinalizador booleano local (let ativo = true) para condicionar a chamada ao setDados, impedindo atualizações após a desmontagem do componente.

B) Instanciar um AbortController dentro do useEffect, passar seu signal para o fetch e invocar controller.abort() na função de limpeza retornada, interrompendo a requisição em curso ao desmontar.

C) Mover a lógica de busca para um useCallback que receba matricula como dependência, invocar esse callback dentro do useEffect e retornar sua referência como função de limpeza.

D) Envolver o useEffect em um bloco try/catch assíncrono com uma flag de controle e retornar uma função de limpeza que redefina essa flag, prevenindo a atualização de estado em caso de erro ou desmontagem.

E) Substituir o useEffect por useMemo para memorizar a promessa da requisição, derivando o estado do componente diretamente do valor memorizado e eliminando a necessidade de limpeza explícita.

. QUESTÃO 09 Imagine o seguinte cenário hipotético:

O sistema de monitoramento da Polícia Civil do Estado do Piauí integra denúncias anônimas, registros de ocorrências e vínculos entre suspeitos. Atualmente, os dados são armazenados em uma estrutura linear única, em que cada registro contém informações do suspeito e referências textuais a outros envolvidos.

Com o aumento do volume de dados, surgiram três necessidades críticas:

(i) localizar rapidamente um suspeito por meio de CPF ou identificador único,

(ii) listar todos os suspeitos associados a um determinado tipo de crime e

(iii) modelar e percorrer explicitamente relações entre suspeitos, permitindo análise de conectividade.

A equipe técnica identificou que a estrutura atual não atende aos requisitos de desempenho e propôs uma refatoração utilizando diferentes estruturas de dados.

Diante desse cenário, qual abordagem técnica atende simultaneamente aos três requisitos apresentados, considerando acesso eficiente por identificador, agrupamento por categoria e modelagem explícita de relações?

A) Utilizar uma lista sequencial para armazenar os suspeitos e aplicar filtros iterativos para buscas, agrupamentos e identificação manual de relações.

B) Utilizar uma árvore hierárquica para organizar os suspeitos por categoria de crime, mantendo as buscas e as relações por meio de travessias adicionais.

C) Utilizar uma tabela hash para indexar suspeitos por identificador e listas auxiliares, mantendo relações como referências textuais.

D) Utilizar uma combinação de tabela hash para acesso por identificador, listas por tipo de crime e um grafo para modelar relações.

E) Utilizar uma matriz de adjacência completa para representar suspeitos e suas relações, incluindo atributos diretamente na matriz.

QUESTÃO 10 Imagine o seguinte cenário hipotético:

A plataforma integrada de atendimento ao cidadão do Estado do Piauí consolida solicitações de serviços públicos, como a emissão de documentos, a apresentação de denúncias e o envio de pedidos administrativos.

O sistema atual enfrenta um gargalo crítico: consultas frequentes para verificar rapidamente se um cidadão já realizou determinada solicitação e análises periódicas que identificam padrões de comportamento coletivo.

A equipe técnica identificou que o volume de dados cresceu exponencialmente e que o sistema precisa atender a dois requisitos conflitantes: (i) fornecer respostas com tempo de acesso constante (ou próximo de O(1)) para consultas individuais em tempo real; e (ii) permitir modelagem explícita de relações entre registros por meio de estruturas adequadas a grafos esparsos, mantendo o consumo de memória controlado.

Diante desse cenário, qual decisão técnica atende simultaneamente aos requisitos de acesso em tempo constante e modelagem eficiente de relações com uso controlado de memória?

A) Utilizar uma tabela hash para acesso direto por identificador e um grafo com lista de adjacência para modelar relações entre solicitações de forma esparsa.

B) Utilizar uma lista sequencial para armazenar todas as solicitações e aplicar algoritmos de busca linear para consultas e varreduras completas na análise de relações.

C) Utilizar uma árvore balanceada para organizar as solicitações por identificador, mantendo a análise das relações por meio de travessias sucessivas na estrutura.

D) Utilizar uma tabela hash para consultas em tempo constante e uma matriz de adjacência para representar relações completas entre solicitações correlatas.

E) Utilizar uma matriz de adjacência completa para armazenar dados e relações, priorizando o acesso direto em detrimento do consumo de memória.

QUESTÃO 11 Imagine o seguinte cenário hipotético:

Um instituto de transparência em eleições gerencia o banco de dados da plataforma "Voz do Piauí" para consolidar dados de pesquisas eleitorais municipais. Um administrador de banco de dados (DBA) precisa organizar e executar diversas tarefas críticas de manutenção e operação utilizando a linguagem SQL padrão.

Considere as seguintes tarefas SQL a serem executadas no banco de dados de pesquisas eleitorais:

```
I. GRANT SELECT ON intencao_voto TO
analista_estatistico;

II. UPDATE candidatos SET partido =
'Partido DBA' WHERE nome = 'Candidato A';

III. CREATE TABLE candidatos (id SERIAL
PRIMARY KEY, nome TEXT, partido
VARCHAR(10));

IV. DELETE FROM respostas WHERE
data_pesquisa < '2024-01-01';

V. REVOKE INSERT ON intencao_voto FROM
digitador_junior;

VI. TRUNCATE TABLE logs_auditoria;

VII. ALTER TABLE pesquisas ADD COLUMN
tipo_pesquisa TEXT DEFAULT 'Municipal';
```

Classifique cada tarefa acima como pertencente à Linguagem de Definição de Dados (DDL), Linguagem de Manipulação de Dados (DML) ou Linguagem de Controle de Dados (DCL) e marque a alternativa com a sequência correta.

A) I (DCL), II (DML), III (DDL), IV (DML), V (DCL), VI (DDL), VII (DDL)

B) I (DDL), II (DML), III (DDL), IV (DML), V (DCL), VI (DML), VII (DDL)

C) I (DCL), II (DDL), III (DML), IV (DML), V (DCL), VI (DML), VII (DML)

D) I (DDL), II (DML), III (DDL), IV (DML), V (DDL), VI (DDL), VII (DDL)

E) I (DCL), II (DML), III (DML), IV (DML), V (DCL), VI (DML), VII (DML)

. QUESTÃO 12 Imagine o seguinte cenário hipotético:

O sistema de arrecadação da Secretaria da Fazenda (SEFAZ) mantém uma tabela com mais de 50 milhões de registros de tributos. Durante o fechamento fiscal, uma consulta SQL que filtra por `status = 'PENDENTE'` e pela `data_vencimento` do mês corrente apresenta alta latência. O plano de execução indica a ocorrência de *Full Table Scan*.

Considerando a necessidade de otimizar o tempo de resposta sem alterar significativamente a arquitetura, qual solução técnica é mais adequada?

A) Criar um índice composto nas colunas de status e data de vencimento para acelerar a filtragem dos registros.

B) Implementar uma visão materializada com atualização síncrona a cada alteração para reduzir o custo de leitura.

C) Alterar o nível de isolamento das transações para serializável visando reduzir inconsistências na leitura.

D) Desnormalizar a tabela principal incorporando dados relacionados para reduzir operações de junção.

E) Migrar os dados recentes para um banco NoSQL em memória visando reduzir a latência de leitura.

. QUESTÃO 13 Imagine o seguinte cenário hipotético:

A Secretaria de Saúde do Piauí solicitou o desenvolvimento urgente de um sistema para centralizar o agendamento de vacinas. O Governador estabeleceu um prazo restrito de quatro semanas para iniciar a geração de valor para a população. A equipe de engenharia debate a melhor forma de estruturar o projeto baseando-se nos conceitos modernos de Produto Mínimo Viável (MVP) e *Lean Startup*. Considerando as metodologias ágeis focadas em maximizar o aprendizado prático, qual ação técnica o residente deve adotar?

A) Construir um formulário integrado a um banco relacional robusto, contemplando todas as vacinas disponíveis atualmente na rede estadual.

B) Desenvolver um protótipo estático navegável utilizando dados fictícios, priorizando a aprovação visual da diretoria antes da codificação.

C) Implementar uma arquitetura escalável de microsserviços focada no módulo central, garantindo alta disponibilidade antes do lançamento.

D) Lançar uma interface funcional restrita ao agendamento de uma vacina piloto, operando com infraestrutura mínima para coletar métricas de uso.

E) Escrever uma documentação técnica de requisitos com diagramas UML, estabelecendo um escopo rígido para evitar retrabalho de código.

. QUESTÃO 14 Imagine o seguinte cenário hipotético:

O sistema de notas da Rede Estadual de Ensino passa por uma profunda modernização. Um desenvolvedor júnior submete um *Pull Request* (PR) contendo 2.500 linhas alteradas, misturando a correção de um erro crítico de médias, uma nova paleta de cores para a interface e mudanças estruturais no banco de dados. Baseado nas boas práticas de *Code Review* e integração contínua, qual diretriz técnica o revisor deve aplicar imediatamente?

A) Aprovar o código integralmente para não bloquear a esteira de integração, inserindo comentários sobre a necessidade de futuras refatorações.

B) Reescrever o código enviado pelo desenvolvedor durante a sessão de revisão, assegurando que os padrões de arquitetura limpa sejam seguidos.

C) Executar testes de carga manuais no ambiente de homologação com o código atual, focando a validação de performance do banco de dados.

D) Rejeitar a submissão devido ao elevado número de linhas, impondo a adoção restrita de programação em par contínua nas próximas tarefas.

E) Solicitar a divisão do artefato técnico em partes menores e de alta coesão, separando isoladamente as correções, alterações visuais e de banco.

. QUESTÃO 15 Imagine o seguinte cenário hipotético:

O sistema principal do DETRAN-PI opera há 15 anos sob uma arquitetura monolítica fortemente

acoplada. A dívida técnica é tão elevada que adicionar um novo campo exige semanas de testes.

A gestão autorizou a modernização da plataforma, mas impôs que a emissão diária de CNHs não seja interrompida sob nenhuma hipótese.

Analisando as estratégias avançadas para lidar com sistemas defasados, qual abordagem arquitetural mitiga os riscos sistêmicos?

A) Aplicar o padrão estrangulador (*Strangler Fig*), extraindo gradativamente as regras de negócio para novos serviços enquanto o legado se mantém operante.

B) Congelar a criação de novas funcionalidades no sistema principal, direcionando a equipe para reescrever o código do zero em microsserviços modernos.

C) Refatorar o código-fonte monolítico adotando padrões de projeto estruturais, preservando integralmente as tecnologias de base e o framework atual.

D) Migrar os esforços operacionais de qualidade para a automação exaustiva de testes de interface gráfica visando contornar a fragilidade estrutural subjacente.

E) Terceirizar a manutenção corretiva do sistema legado para uma consultoria técnica, alocando a equipe interna exclusivamente no levantamento de novos requisitos.

QUESTÃO 16 Imagine o seguinte cenário hipotético:

O sistema de despacho da Polícia Militar adota a metodologia *Scrum* com iterações de duas semanas.

Devido à natureza sensível da operação ostensiva, surgem incidentes críticos diariamente, forçando o *Product Owner* a inserir demandas emergenciais no meio da *sprint* ativa. Esse padrão destrói o fluxo estipulado e impede a previsibilidade.

Analisando a incompatibilidade do processo atual com o domínio de alta urgência, qual adaptação resolve o problema considerando a necessidade de lidar com demandas emergenciais contínuas?

A) Aumentar a duração das *sprints* de desenvolvimento para quatro semanas, criando uma margem temporal maior para solucionar falhas não previstas.

B) Proibir categoricamente a inclusão de falhas detectadas durante a *sprint* ativa, forçando a corporação a aguardar o próximo ciclo de planejamento.

C) Transitar para um fluxo contínuo orientado a classes de serviço (*Kanban*), instituindo limites de trabalho em progresso para acomodar incidentes.

D) Contratar um gestor de projetos tradicional focado no controle restrito de escopo, rejeitando solicitações não planejadas no início da execução.

E) Adicionar profissionais especializados em suporte à equipe ágil intermitentemente, garantindo que o acréscimo de pessoal elimine rapidamente os gargalos.

QUESTÃO 17 Imagine o seguinte cenário hipotético:

A Secretaria da Fazenda precisa implantar um módulo de cálculo de alíquotas fiscais devido a uma nova lei que entra em vigor em 15 dias, sob risco de paralisação na arrecadação do estado.

A equipe avalia que o tempo é suficiente para codificar a funcionalidade, mas não para desenvolver a suíte completa de testes. Ao avaliar o conflito direto entre prazo legal e qualidade sistêmica, qual decisão atende ao prazo legal e mantém a consistência das regras de negócio?

A) Postergar o lançamento do sistema governamental por mais um mês, assegurando a cobertura técnica integral do código por meio de testes unitários.

B) Concentrar a automação de testes estritamente nas regras matemáticas do cálculo tributário, assumindo a dívida técnica de testes temporariamente na interface visual.

C) Entregar o produto no prazo legal sem verificações automatizadas, mitigando os riscos por meio da forte alocação da equipe no suporte telefônico pós-lançamento.

D) Automatizar os fluxos visuais de interação do contribuinte via testes de ponta a ponta, negligenciando as validações lógicas sistêmicas do banco de dados.

E) Reduzir de forma autônoma o escopo funcional estipulado pela legislação, removendo rotinas de validação fiscal para conquistar tempo de compilar testes automatizados.

QUESTÃO 18 Imagine o seguinte cenário hipotético:

O "Portal do Cidadão" consolidará dados de saúde e educação atualmente espalhados em 10 APIs governamentais defasadas e lentas.

Um requisito não funcional impõe que o carregamento do painel principal (*Dashboard*)

ocorra em menos de 1 segundo, suportando até 100 mil acessos simultâneos previstos.

Avaliando *trade-offs* de integração entre sistemas sob severa pressão de leitura, qual arquitetura garante a performance exigida?

A) Realizar chamadas síncronas bloqueantes para as APIs legadas a cada acesso, garantindo que os relatórios na tela correspondam aos dados fidedignos em tempo real.

B) Migrar as antigas bases de dados relacionais para um super servidor centralizado na nuvem, explorando consultas indexadas de junção para compor a visão global do portal.

C) Hospedar o banco de dados integralmente na memória volátil (RAM) de cada réplica da aplicação *backend*, duplicando fisicamente os dados populacionais para zerar a latência.

D) Implementar uma camada de projeção consolidada (cache/NoSQL), atualizando o perfil dos cidadãos de forma progressiva e assíncrona a partir da emissão de novos eventos.

E) Utilizar procedimentos armazenados (*Stored Procedures*) no banco de dados central do portal, encarregando-o de orquestrar nativamente as chamadas remotas de autenticação.

QUESTÃO 19 Imagine o seguinte cenário hipotético:

Durante o refinamento do *Backlog* de um aplicativo de benefícios sociais, surgiram demandas concorrentes. Os desenvolvedores propõem refatorar a estrutura do banco de dados visando reduzir o esforço em futuras funcionalidades; a diretoria política exige melhorias visuais na tela inicial; e uma auditoria de segurança reportou vulnerabilidades de *SQL Injection* que expõem os dados bancários dos cidadãos. Avaliando os critérios ágeis de valor público e mitigação de riscos, qual priorização reduz o risco de segurança e maximiza o valor ao cidadão?

A) Ordenar as tarefas colocando a correção das vulnerabilidades de segurança como prioridade máxima, seguida imediatamente pelas entregas que gerem maior valor funcional aos beneficiários.

B) Priorizar as alterações estéticas solicitadas pela alta gestão governamental para assegurar o patrocínio orçamentário, mitigando temporariamente as falhas de segurança na rede interna.

C) Selecionar as tarefas de refatoração de infraestrutura visando elevar artificialmente as métricas de velocidade da equipe, acelerando as entregas programadas nas próximas *sprints*.

D) Organizar o fluxo metodológico estritamente pela data original de protocolo, assegurando que demandas mais antigas de melhoria visual sejam solucionadas antes de falhas recentes.

E) Delegar a decisão da ordem de execução inteiramente aos engenheiros desenvolvedores, possibilitando que demandas exclusivas de arquitetura e tecnologia ditem o rumo inicial da *sprint*.

QUESTÃO 20 Imagine o seguinte cenário hipotético:

O projeto "Smart City" instalará milhares de sensores em semáforos urbanos, emitindo telemetria continuamente em alta frequência.

A arquitetura de software deve cumprir dois requisitos simultâneos: (i) processar essas leituras para alertar cidadãos via celular sobre alagamentos quase em tempo real; e (ii) armazenar todos os registros de forma persistente para futuras análises pesadas de *Big Data* governamental.

Sintetizando conceitos de sistemas de alta vazão e tolerância a falhas, qual topologia atende a ambas as restrições?

A) Desenhar microsserviços limitados a processamento síncrono que inserem leituras diretamente no banco relacional, utilizando os *triggers* nativos da base para disparar alertas móveis.

B) Construir um sistema monolítico fortemente acoplado a rotinas de agendamento (*cron jobs*) que consolidam os registros a cada intervalo fechado, emitindo os alertas governamentais em lote.

C) Projetar uma arquitetura orientada a eventos utilizando um barramento de mensageria escalável, possuindo consumidores independentes para disparo responsivo dos alertas e para gravação persistente dos dados.

D) Idealizar o fluxo transacional valendo-se isoladamente de funções executáveis sob demanda (*Serverless*), gravando dados efêmeros em memória temporária para reduzir drásticos custos operacionais.

E) Estabelecer comunicação dinâmica ponto-a-ponto descentralizada estritamente entre os sensores semafóricos e os celulares locais da população, abdicando de qualquer persistência centralizada.

QUESTÃO 21 Imagine o seguinte cenário hipotético:

O sistema de ouvidoria digital do Estado recebe diariamente milhares de manifestações de

cidadãos, incluindo reclamações, denúncias, solicitações de informações e elogios. Atualmente, a triagem inicial é realizada manualmente por servidores, o que aumenta o tempo de resposta e gera filas de atendimento, especialmente em períodos de maior demanda.

A equipe técnica da PiauiGovTech recebeu a missão de automatizar a classificação inicial dessas manifestações, utilizando o histórico de registros já existentes, previamente categorizados pelos atendentes. O objetivo do projeto é reduzir o tempo de encaminhamento sem alterar, neste primeiro momento, as categorias institucionais já adotadas pelo órgão.

Considerando os paradigmas clássicos de aprendizado de máquina, qual abordagem o residente deve adotar para atender corretamente ao problema apresentado?

A) Aplicar aprendizado não supervisionado para descobrir agrupamentos latentes nas manifestações, substituindo as categorias administrativas previamente definidas.

B) Utilizar aprendizado supervisionado para treinar um classificador com o histórico rotulado e prever a categoria de novas manifestações recebidas.

C) Implementar o aprendizado por reforço para que o modelo ajuste a categorização com base nas recompensas atribuídas após cada atendimento concluído.

D) Empregar um modelo generativo de linguagem sem treinamento específico, delegando ao LLM a criação autônoma de novas categorias de triagem.

E) Adotar exclusivamente regras heurísticas baseadas em palavras-chave, eliminando a necessidade de dados históricos e de treinamento estatístico.

QUESTÃO 22 Imagine o seguinte cenário hipotético:

Uma plataforma estadual de acompanhamento de processos administrativos passou a utilizar um LLM para responder a dúvidas frequentes dos cidadãos sobre prazos, documentos exigidos e etapas processuais. Após as primeiras semanas de uso, a equipe percebeu que as respostas geradas frequentemente são genéricas, pouco objetivas e, em alguns casos, deixam de considerar detalhes relevantes do processo informado pelo usuário.

A gestão do projeto impôs uma restrição importante: nesta fase, não haverá troca do modelo base nem realização de fine-tuning, pois o contrato vigente prevê apenas ajustes de integração e de uso operacional do serviço já contratado. Assim, a equipe precisa melhorar a qualidade das respostas por meio de intervenções de baixo custo e de rápida implementação.

Diante desse cenário, qual ação técnica deve ser priorizada pelo residente para aumentar a precisão e a utilidade das respostas?

A) Reduzir o tamanho das entradas para economizar tokens, removendo detalhes contextuais e deixando o modelo inferir o restante da resposta.

B) Executar o mesmo prompt repetidas vezes e selecionar uma das saídas geradas de forma probabilística para diversificar a resposta.

C) Reformular os prompts com instruções claras, contexto específico, restrições de formato e exemplos de respostas esperadas para o modelo.

D) Remover orientações detalhadas para ampliar a liberdade de geração do modelo, permitindo respostas mais naturais em linguagem aberta.

E) Substituir temporariamente o LLM por regras fixas em uma árvore de decisão, eliminando o uso de geração textual até nova contratação.

QUESTÃO 23 Imagine o seguinte cenário hipotético:

Uma plataforma de monitoramento de evasão escolar foi desenvolvida para prever estudantes com maior risco de abandono antes do encerramento do período letivo. O modelo inicial apresentou desempenho aparentemente excelente durante o desenvolvimento, atingindo 98% de acurácia nos dados usados no treinamento. Entretanto, ao ser avaliado com dados mais recentes de outras escolas da rede, o desempenho caiu substancialmente, revelando que o comportamento observado em produção ficou muito distante do esperado pela equipe.

Após análise preliminar, os engenheiros levantaram a hipótese de que o modelo tenha aprendido padrões específicos demais do conjunto de treinamento, sem capacidade de generalização para contextos novos. A equipe precisa agora decidir qual intervenção técnica é mais apropriada para melhorar a robustez preditiva sem abandonar a abordagem supervisionada já adotada.

Considerando o cenário descrito, qual ação o residente deve defender como a mais adequada?

A) Aumentar o número de épocas de treinamento para que o modelo memorize com mais precisão o comportamento do conjunto original utilizado.

B) Aplicar técnicas de regularização, validação cruzada e ajuste da complexidade do modelo para reduzir o overfitting e melhorar a generalização.

C) Eliminar o conjunto de validação e avaliar o desempenho exclusivamente com base nos dados de treino, priorizando a consistência estatística interna.

D) Substituir o classificador supervisionado por um método não supervisionado, evitando a dependência de rótulos históricos da rede escolar.

E) Reduzir arbitrariamente a quantidade de atributos e amostras sem análise experimental, visando simplificar o treinamento do algoritmo.

QUESTÃO 24 Imagine o seguinte cenário hipotético:

O sistema estadual de controle e distribuição de medicamentos tem como objetivo apoiar profissionais da rede pública de saúde na consulta rápida a protocolos clínicos, a normas de dispensação e a orientações técnicas atualizadas da Secretaria de Saúde.

Na versão atual, a busca por informações é realizada por meio de um mecanismo baseado em palavras-chave aplicado a uma base documental interna. Entretanto, avaliações de uso revelaram que esse modelo apresenta limitações significativas: consultas feitas em linguagem natural frequentemente retornam documentos pouco relevantes ou deixam de recuperar trechos importantes que são semanticamente relacionados à intenção do usuário, mesmo estando corretos do ponto de vista clínico e administrativo.

Diante desse cenário, a equipe da PiauiGovTech planeja evoluir a solução por meio da incorporação de um modelo de linguagem (LLM) para gerar respostas mais naturais. Contudo, há uma restrição crítica: as respostas devem estar fundamentadas nos documentos oficiais da Secretaria, evitando conteúdos imprecisos, desatualizados ou não alinhados às normas institucionais. Além disso, a solução deve permitir a atualização contínua da base documental e aprimorar a relevância das respostas nas consultas em linguagem natural.

Considerando os requisitos de relevância semântica, aderência ao conteúdo institucional e atualização dinâmica da base de conhecimento, qual arquitetura o residente deve propor para atender adequadamente a esse cenário?

A) Implementar um Sistema Especialista baseado em regras rígidas de decisão (If-Then), onde as consultas em linguagem natural são convertidas em comandos lógicos que acessam protocolos previamente mapeados em um banco de dados relacional, garantindo que o fluxo de resposta seja imutável.

B) Utilizar um modelo de linguagem (LLM) integrado a um mecanismo de busca por correspondência textual exata, que filtra documentos por palavras-chave contidas na pergunta do usuário e anexa o conteúdo integral desses arquivos ao prompt de comando para processamento do modelo.

C) Implementar uma arquitetura de Retrieval-Augmented Generation (RAG), utilizando modelos de embeddings para converter os documentos oficiais em vetores armazenados em uma base de dados vetorial, permitindo que o LLM gere respostas fundamentadas exclusivamente nos trechos recuperados por similaridade semântica.

D) Desenvolver um mecanismo de busca sequencial indexada por metadados administrativos, no qual o sistema percorre cronologicamente os arquivos da Secretaria para encontrar termos técnicos específicos, utilizando o LLM apenas para corrigir a gramática da consulta inicial do profissional.

E) Realizar o Fine-tuning (ajuste fino) de um modelo de linguagem de grande escala utilizando exclusivamente a base de documentos da Secretaria de Saúde, confiando na memória paramétrica do modelo ajustado para responder às consultas de forma direta, sem a necessidade de um componente de recuperação externa.

QUESTÃO 25 Imagine o seguinte cenário hipotético:

Um sistema integrado de transporte público estadual deseja evoluir de um chatbot informativo para um assistente operacional capaz de executar ações em tempo real. A nova versão deverá responder perguntas dos usuários sobre horários previstos, consultar APIs externas de geolocalização e de frota, acionar serviços de cálculo de rotas e, quando necessário, combinar múltiplas chamadas de ferramentas em sequência antes de formular a resposta final.

A equipe considera utilizar um LLM como núcleo de raciocínio, mas precisa de uma arquitetura que permita ao modelo decidir quando invocar ferramentas externas, encadear chamadas e integrar resultados heterogêneos provenientes de serviços distintos. O projeto também prevê evolução futura para múltiplos fluxos operacionais, exigindo uma coordenação mais sofisticada do que prompts isolados.

Considerando esse cenário, qual abordagem técnica o residente deve adotar?

A) Utilizar o LLM apenas como gerador textual, restringindo suas respostas ao conhecimento paramétrico já presente no modelo base, sem acesso a serviços externos.

B) Converter previamente todas as respostas possíveis em registros estáticos de banco de dados, removendo a dependência de consultas em tempo real às APIs operacionais.

C) Implementar um agente com function calling e camada de orquestração, utilizando frameworks apropriados para integrar ferramentas e coordenar múltiplas chamadas.

D) Substituir as integrações online por arquivos locais atualizados periodicamente, priorizando simplicidade operacional em detrimento da responsividade do sistema.

E) Reescrever todo o fluxo como uma árvore fixa de decisões manuais, evitando que o modelo tenha qualquer papel no acionamento de serviços complementares.

. QUESTÃO 26 Imagine o seguinte cenário hipotético:

Um sistema estadual de priorização para distribuição de medicamentos de alto custo passou a utilizar modelos de IA para apoiar a análise de solicitações. Após auditorias internas e questionamentos de órgãos de controle, foram identificados dois riscos principais: (i) possibilidade de viés algorítmico com impacto diferenciado entre grupos de cidadãos e (ii) uso inadequado de dados pessoais sensíveis durante o treinamento e a operação do sistema.

A Secretaria responsável decidiu manter o sistema em operação, condicionando sua continuidade à adoção de medidas técnicas que assegurem conformidade com a Lei Geral de Proteção de Dados (LGPD), especialmente quanto aos princípios de finalidade, necessidade, transparência e responsabilização, bem como à mitigação de vieses algorítmicos, sem interrupção do serviço.

Considerando exclusivamente boas práticas de governança de IA, técnicas de mitigação de viés e os princípios explicitados da LGPD, qual decisão técnica atende integralmente às condições estabelecidas no cenário descrito?

A) Aumentar a complexidade do modelo com o objetivo de melhorar a acurácia global, mantendo inalterados os processos de auditoria, tratamento de dados e monitoramento do sistema.

B) Implementar mecanismos de auditoria contínua, técnicas de explicabilidade, estratégias de minimização ou anonimização de dados pessoais sensíveis e monitoramento sistemático de viés algorítmico, com registro das decisões e rastreabilidade do modelo.

C) Restringir o acesso às informações sobre o funcionamento do sistema e aos critérios de decisão automatizada, limitando a transparência para reduzir riscos institucionais.

D) Ampliar a coleta e utilização de dados pessoais sensíveis no treinamento do modelo, sem revisão dos critérios de necessidade e finalidade, visando aumentar o volume de dados disponíveis.

E) Descontinuar integralmente o uso de IA no processo de priorização, sem adoção de medidas de auditoria, mitigação de viés ou adequação regulatória do sistema existente.

. QUESTÃO 27 Imagine o seguinte cenário hipotético:

O sistema da Secretaria de Saúde precisa filtrar uma lista de cidadãos para identificar aqueles aptos à vacinação. A regra exige selecionar apenas indivíduos com idade maior ou igual a 60 anos e com liberação médica (`pode_vacinar = True`). Considerando boas práticas em Python, qual implementação atende corretamente ao requisito?

A) Utilizar compreensão de listas com filtro direto nas condições de idade e liberação médica.

B) Aplicar função `filter` utilizando operador lógico `or` entre idade e liberação médica.

C) Utilizar laço com condição `while` para construir lista baseada nos critérios definidos.

D) Utilizar método `filter` com operadores `&&` e valores booleanos em sintaxe JavaScript.

E) Utilizar função `map` para retornar valores booleanos das condições avaliadas.

. QUESTÃO 28 Imagine o seguinte cenário hipotético:

Um sistema cruza duas listas com milhões de CPFs para identificar inconsistências. A implementação atual utiliza dois laços aninhados, resultando em alta latência. Considerando otimização de desempenho, qual abordagem reduz a complexidade da operação de busca?

A) Converter listas em conjuntos (set) e utilizar operação de interseção.

B) Ordenar as listas antes de executar busca sequencial.

C) Converter listas em tuplas para melhorar acesso em memória.

D) Processar dados em pequenos lotes utilizando JSON.

E) Utilizar listas encadeadas para reduzir custo de busca.

QUESTÃO 29 Imagine o seguinte cenário hipotético:

Uma organização não governamental (ONG) de defesa do meio ambiente está desenvolvendo um sistema para monitorar a qualidade do ar e da água em diferentes bacias hidrográficas. No desenvolvimento do software, utilizando o paradigma de Orientação a Objetos, a equipe identificou as seguintes necessidades:

1. Todos os equipamentos de monitoramento possuem características comuns, como um identificador único e a localização geopolítica.
2. Existem equipamentos específicos: o "EstacaoMeteorologica" mede a velocidade do vento e pluviosidade; o "BoiaMonitoramentoAgua" mede a turbidez e o pH da água.
3. O sistema central precisa solicitar uma "leitura de dados" para qualquer equipamento, independentemente do seu tipo específico, e cada equipamento deve responder com os dados apropriados à sua natureza.

Considerando os conceitos de Orientação a Objetos, qual abordagem de modelagem deve ser aplicada para atender ao requisito número 3, permitindo que o sistema trate todos os equipamentos de forma genérica para solicitar leituras, mas obtendo o comportamento específico de cada um?

A) Criar classes separadas e independentes para EstacaoMeteorologica e BoiaMonitoramentoAgua, e utilizar uma estrutura de decisão (if/else) no sistema central para verificar o tipo do objeto antes de chamar o método de leitura correspondente.

B) Aplicar o conceito de Herança, fazendo com que as classes específicas herdem de uma classe base Equipamento, e utilizar Encapsulamento para esconder os métodos de leitura do sistema central.

C) Utilizar Polimorfismo, definindo um método realizarLeitura() na classe base Equipamento e sobrescrevendo-o nas subclasses EstacaoMeteorologica e BoiaMonitoramentoAgua com suas implementações específicas.

D) Implementar Sobrecarga de métodos na classe central do sistema, criando múltiplos métodos com o mesmo nome (realizarLeitura), mas que aceitam parâmetros diferentes (um para cada tipo de equipamento específico).

E) Aplicar todas as modificações anteriores.

QUESTÃO 30 Imagine o seguinte cenário hipotético:

O sistema inteligente de distribuição de vagas escolares da SEDUC-PI processa solicitações simultâneas de matrícula e redistribuição de alunos. A implementação atual utiliza orientação a objetos com entidades mutáveis compartilhadas entre serviços concorrentes.

Durante períodos de alta demanda, foram observados: ocorrência de race conditions; inconsistência de dados decorrente de atualizações concorrentes; dificuldade de auditoria das decisões automatizadas devido à presença de efeitos colaterais.

A equipe técnica definiu que a solução deve, simultaneamente: impedir modificações concorrentes sobre o mesmo estado; garantir previsibilidade das operações (ausência de efeitos colaterais); manter a estrutura orientada a objetos na representação do domínio.

Diante desses requisitos, qual decisão técnica atende integralmente às três condições estabelecidas?

A) Manter entidades mutáveis e utilizar mecanismos de sincronização, como locks e controle transacional, para coordenar acessos concorrentes.

B) Migrar completamente o sistema para programação funcional pura, eliminando o uso de classes em todas as camadas.

C) Refatorar os fluxos críticos para utilizar dados imutáveis e funções puras, mantendo a orientação a objetos na modelagem do domínio.

D) Centralizar o controle de estado em um único serviço responsável por coordenar todas as operações concorrentes.

E) Reestruturar o sistema para um modelo imperativo baseado em execução sequencial, reduzindo a concorrência.