SUGESTÃO DE RESPOSTAS PARA O 2º EXERCÍCIO ESCOLAR IF977 – 2025.1

Cenário: A **PetFood** é um aplicativo de entrega de comida para animais de estimação que será uma plataforma inovadora que conecta tutores de pets a fornecedores de alimentação natural de qualidade, facilitando o processo de compra e entrega de ração e petiscos baseados em alimentação natural diretamente na porta dos clientes. O aplicativo busca atender a uma crescente demanda por conveniência e cuidado com a alimentação de animais, garantindo praticidade e segurança tanto para os tutores quanto para os pets.

O time de desenvolvimento do PetFood está trabalhando na entrega de um novo recurso: **integração com dispositivos de smart home** (como Alexa, Google Nest e sensores de despensa inteligentes), permitindo que tutores recebam lembretes automáticos sobre os horários de alimentação de seus pets e alertas sobre baixo estoque de ração.

Como engenheiro de software responsável por essa funcionalidade, você foi incumbido de garantir que a **qualidade**, a **segurança** e os **testes automatizados** estejam em conformidade com os padrões do projeto. Além disso, espera-se que você contribua com a integração desse novo recurso ao pipeline de CI/CD já existente.

Questão 01 [2,0] – Fundamentos do Teste de Software

a) Tipos de teste esperados (dois exemplos adequados entre):

- **Teste de Unidade:** Para validar individualmente os métodos responsáveis por acionar lembretes, integrar com APIs de smart home ou atualizar status de alimentação.
- **Teste de Integração:** Para garantir que a comunicação entre o sistema PetFood e os dispositivos de smart home ocorra de forma correta, simulando trocas reais de dados.
- **Teste de Aceitação:** Para verificar se os lembretes funcionam como esperado do ponto de vista do usuário final, contemplando horários corretos e notificações apropriadas.
- **Teste de Regressão:** Para assegurar que novas funcionalidades não comprometam o funcionamento anterior do aplicativo.

b) Justificativas esperadas:

- Demonstrar compreensão do **contexto funcional** (ex.: confiabilidade da integração, atendimento às expectativas do usuário).
- Relacionar o tipo de teste à **validação de requisitos**, **segurança do funcionamento** ou **manutenção evolutiva**.
- Mostrar noções de como os testes contribuem para **rastreabilidade**, **documentação viva** ou **diálogo entre requisitos e arquitetura**.

Questão 02 [3,0] – Qualidade de Software + Garantia de Qualidade Contínua

Critérios de Qualidade (mínimo dois, bem justificados)

- **Confiabilidade (Reliability)**: O sistema deve garantir que o lembrete funcione de forma consistente, mesmo em cenários com diferentes dispositivos e redes.
- **Usabilidade**: O lembrete precisa ser claro, acionado corretamente e compreendido pelo tutor, independentemente do dispositivo.
- **Portabilidade ou Interoperabilidade**: É essencial que a funcionalidade opere com diversos dispositivos de smart home e plataformas.

- **Eficiência de Desempenho**: Os lembretes devem ocorrer no tempo certo, sem atrasos perceptíveis.
- **Segurança**: O dado que aciona o lembrete (e.g., horário, nome do pet) precisa ser protegido.

O aluno pode citar outros critérios desde que coerentes e bem contextualizados ao cenário.

Técnica de Medição ou Monitoramento Contínuo

- **Logging e análise de eventos (monitoramento ativo)**: Coleta de eventos para entender taxas de sucesso/falha no disparo de lembretes.
- **Testes automatizados periódicos com variações de ambiente**: Simulam uso com diferentes dispositivos, medem estabilidade da integração.
- **Dashboards com métricas de falha ou latência de resposta**: Apoiam decisões sobre manutenção.
- **Feedback in-app monitorado**: Canal direto de retorno dos usuários.

Justificativa da abordagem como suporte à melhoria contínua

- Promove **detecção precoce de falhas**, especialmente com base em padrões de uso real.
- Permite adaptação rápida do sistema conforme o comportamento da base de usuários e tipos de dispositivos.
- Melhora a **resiliência da funcionalidade**, evitando regressões e sustentando a escalabilidade da solução.
- Favorece **decisões baseadas em dados**, alinhando técnica e negócio.

Questão 03 [5,0] – DevSecOps + Testes Avançados + Integração Contínua

a) Ameaças de segurança relevantes (espera-se de 2 a 3 exemplos bem fundamentados):

- **Vazamento de dados sensíveis**: Exposição de dados como horários de alimentação, nomes dos pets ou localização dos dispositivos.
- **Ataques de injeção** (e.g., comandos maliciosos via integração com APIs ou assistentes virtuais).
- **Autenticação fraca / falhas de autorização**: Acesso indevido ao controle de dispositivos ou dados do usuário.
- **Dependências inseguras**: Uso de bibliotecas de terceiros com vulnerabilidades conhecidas.
- **Comunicação não criptografada** entre o app e os dispositivos integrados.
- **Exposição de APIs mal protegidas**.

A resposta ideal aponta ameaças relevantes **para o contexto da integração com smart home**, com compreensão dos riscos sistêmicos e do impacto para os usuários.

b) Práticas, ferramentas e automações para mitigar as ameaças:

- **SAST (Static Application Security Testing)** integrado ao pipeline para detectar vulnerabilidades no código.
- **DAST (Dynamic Application Security Testing)** para testes automatizados em ambiente de execução.
- **Dependabot / OWASP Dependency-Check / Snyk** para escanear bibliotecas de terceiros.

- **Controle de acesso baseado em papéis (RBAC)** e autenticação multifator para APIs internas.
- **Análise de arquitetura com foco em segurança (ex: modelo C4 + STRIDE)**.
- **Uso de métricas de cobertura de testes de segurança (e.g., > 80%)**.
- **Monitoramento contínuo de produção com alertas para comportamento anômalo**.
- **Testes de segurança como parte do ciclo de build (shift-left)**.
- **Revisão de código com foco em segurança e ownership coletivo**.

Espera-se que o aluno conecte boas práticas com ferramentas modernas, mostrando visão de automação e integração ao CI/CD.

c) Estratégias para preservar a agilidade do pipeline:

- **Automatizar ao máximo os testes de segurança**, minimizando a intervenção manual.
- **Aplicar o princípio do** *shift-left*: segurança tratada desde as fases iniciais do desenvolvimento.
- **Executar testes de segurança em paralelo ao build/deploy**.
- **Usar análise incremental em SAST**, para verificar só o código alterado e acelerar ciclos.
- **Empregar estratégias de "security gates" parametrizáveis**, que só bloqueiam releases com falhas críticas.
- **Promover cultura de segurança distribuída**: desenvolvedores conscientes da importância da segurança e capazes de antecipar problemas.
- **Documentação viva e integração entre times (Dev, QA, Sec)** para comunicação eficaz e sem burocracia.

Respostas ideais reconhecem que segurança e agilidade são **complementares**, não conflitantes — e propõem soluções técnicas e culturais para manter o equilíbrio.

RUBRICA DE CORREÇÃO — 2º EXERCÍCIO ESCOLAR DE ENGENHARIA DE SOFTWARE: PETFOOD

Questão 1 — Fácil (Peso: 2,0)

Tema: Fundamentos do Teste de Software

Critério	Nota Máxima	Descrição
Seleção adequada dos tipos de teste (2 tipos coerentes)	0,8	0,4 por tipo de teste tecnicamente apropriado ao contexto descrito. Tipos genéricos ou incompatíveis somam no máximo 0,2 cada.
Justificativa técnica clara e conectada ao contexto do PetFood/smart home	0,8	0,4 por justificativa coerente, alinhada à proposta da questão e com entendimento dos objetivos do teste (não basta citar "garante que funciona").
Articulação com princípios discutidos em aula (testes como estratégia, rastreabilidade etc.)	0,4	Pontuação atribuída se o aluno mencionar ou aplicar um ou mais princípios debatidos na aula (mesmo que implicitamente), como rastreabilidade, validação de requisito, etc.

Notas intermediárias

- Aluno cita apenas **um tipo de teste** corretamente, com boa justificativa: **até 1,2 pontos**.
- Aluno cita dois testes corretos mas com **justificativas genéricas ou rasas**: **até 1,2 pontos**.
- Aluno faz **associação ao papel estratégico dos testes** (mesmo sem citar o termo), demonstrando boa compreensão: **merece a bonificação de até 0,4 ponto**.
- Respostas apenas conceituais, sem contextualização com o cenário do PetFood, devem **sofrer decréscimo** proporcional, especialmente na coluna de justificativa.

Questão 2 — Intermediária (Peso: 3,0)

Tema: Qualidade de Software + Garantia de Qualidade Contínua

Critério	Nota Máxima	Descrição
Seleção e justificativa de dois critérios de qualidade coerentes com o contexto	1,2	0,6 por critério corretamente identificado e bem justificado. Critérios genéricos ou mal aplicados rendem no máximo 0,3 cada.
Apresentação de uma técnica de medição ou monitoramento contínuo adequada ao cenário	0,8	Técnica precisa estar relacionada à prática real de avaliação contínua e ser aplicável ao caso do PetFood com smart devices.
Justificativa clara da estratégia como forma de mitigar falhas e apoiar decisões de evolução	1,0	O aluno deve mostrar como a proposta ajuda a sustentar qualidade, prever problemas e alinhar com objetivos de negócio. Sem conexão com impacto, nota reduzida.

Notas intermediárias

- O aluno lista os dois critérios corretamente mas **não justifica bem**: atribui no máximo 0,6.
- Se apresenta **apenas uma métrica fraca ou imprecisa**, reduz proporcionalmente a nota da coluna correspondente.
- Caso o aluno traga uma solução baseada em *observabilidade, monitoramento contínuo, testes A/B, simulações*, etc., com boa argumentação, deve receber **nota máxima ou quase total**.
- **Inovações viáveis** devem ser valorizadas, desde que mantenham coerência com os princípios de engenharia de software discutidos na disciplina.

Questão 3 — Difícil (Peso: 5,0)

Tema: DevSecOps + Testes Avançados + Integração Contínua

Critério	Nota Máxima	Descrição
a) Ameaças de segurança contextualizadas e relevantes	1,5	0,75 por ameaça bem justificada e ligada ao cenário (máximo 1,5). Respostas genéricas recebem no máximo 0,5 por item.
b) Práticas, ferramentas e automações aplicáveis ao pipeline CI/CD com foco em segurança	2,0	Até 2,0 pontos por apresentar no mínimo 3 práticas/ferramentas adequadas, com explicação clara de sua função e relevância. Cada item bem explicado vale ~0,6.
c) Estratégias para integrar segurança sem comprometer agilidade	1,5	Deve mostrar equilíbrio entre segurança e entrega contínua. Respostas que destacam automação, shift-left, cultura e integração merecem nota alta.

Notas intermediárias

- Se o aluno demonstrar **entendimento estratégico de DevSecOps**, mesmo com leve imprecisão técnica, privilegia a profundidade da análise.
- Menção a **ferramentas específicas** (Snyk, SonarQube, etc.) mostra familiaridade com práticas modernas e deve ser valorizada.
- Respostas que apresentam apenas **listas desconectadas**, sem justificativa ou aplicação ao cenário, devem ser penalizadas por superficialidade.
- Se o aluno **aponta conflitos entre segurança e velocidade**, mas **não oferece formas de mitigar esse trade-off**, isso limita a pontuação do item c.