DOCUMENTAÇÃO AEP – Aplicativo de Segurança Digital

**SOBRE AS MANUTENÇÕES:**  
  
**CORRETIVA**A manutenção corretiva se refere à correção de bugs identificados após o lançamento do aplicativo. Exemplos incluem:

* Correção de falhas no cálculo de pontuação dos quizzes.
* Resolução de problemas de renderização em dispositivos específicos.
* Correção de bugs de integração entre o backend (API de dados de segurança) e a interface do usuário.

**Exemplos:** Bug: O conteúdo do quiz não era carregado em dispositivos com uma versão anterior de Flutter. Correção: Ajuste na função de carregamento para compatibilidade com múltiplas versões.

**ADAPTATIVA**Envolve ajustar o aplicativo para novos requisitos ou mudanças no ambiente, como atualizações de sistemas operacionais e mudanças nas APIs de segurança.

* Atualização para suporte a uma nova versão do Android ou iOS.
* Ajustes na integração com APIs de segurança que sofreram modificações.

**Exemplo:** Atualização para compatibilidade com o iOS 18 e novos recursos de segurança, como criptografia mais forte.

**PERFECTIVA**Essa manutenção está focada em melhorias contínuas para a experiência do usuário (UX/UI) e novas funcionalidades, como a introdução de mais quizzes e melhorias de performance.

* Melhoria no layout da página de quizzes para torná-la mais acessível.
* Adição de novos tipos de quizzes baseados em inteligência artificial para desafios mais complexos.

**Exemplo:** Introdução de quizzes interativos com feedback em tempo real para melhorar a retenção de conhecimento.

**PREDITIVA**Monitoramento de logs e análise do desempenho do aplicativo para prever e prevenir falhas futuras, principalmente relacionadas ao tráfego de dados e sobrecarga de APIs.

* Implementação de alertas para detectar quando o aplicativo estiver sobrecarregado ou com problemas de resposta devido ao grande volume de dados carregados.
* Análise de logs para prever problemas de performance antes que os usuários sejam afetados.

**Exemplo:** Utilização de logs para detectar latências nas respostas das APIs e melhorar o tempo de carregamento.  
  
**SOBRE O PLANO DE GERENCIAMENTO DE CONFIGURAÇÃO  
  
VERSIONAMENTO**O projeto deve ser gerenciado utilizando Git, com o seguinte fluxo:

* **Branches principais**: main (produção) e develop (desenvolvimento).
* **Branches secundárias**: Feature branches para novas funcionalidades, fix branches para correções e release branches para preparação de versões de produção.
* **Commits**: Todos os commits devem ser claros e descrever as mudanças feitas no código.
* **Tagging**: Cada versão do aplicativo deve ser marcada com tags no Git, como v1.0.0, seguindo a convenção SemVer (Versionamento Semântico).

**CONTROLE DE MUDANÇAS**Antes de qualquer mudança ser implementada, deve haver um processo formal de aprovação:

1. **Solicitação de mudança**: Descrição do problema ou nova funcionalidade.
2. **Avaliação de impacto**: Análise do impacto da mudança no código existente e nos usuários.
3. **Aprovação**: Mudanças devem ser aprovadas por um time responsável (gerentes de projeto e desenvolvedores principais).
4. **Implementação**: Uma vez aprovada, a mudança é implementada em um branch separado e submetida a testes rigorosos antes de ser mesclada.

**ITENS DE CONFIGURAÇÃO**Documentação dos principais itens que fazem parte da configuração do aplicativo, incluindo:

* **Dependências**: Bibliotecas externas usadas, como http, provider, e quaisquer pacotes de segurança (ex. Firebase para autenticação).
* **Configurações de segurança**: Detalhamento das bibliotecas e estratégias usadas para garantir a segurança do aplicativo (ex. criptografia de dados).
* **Recursos de interface de usuário**: Ferramentas e pacotes que gerenciam a UI, como flutter\_hooks ou rxdart para manipulação de estados.