3. Seleção da tecnologia: após avaliação das tecnologias, apresentar a escolhida e a justificativa para tal escolha (considerar aspectos como escalabilidade, facilidade de manutenção e atualização, bem como a disponibilidade de suporte e treinamento para a equipe responsável pela implementação e manutenção da tecnologia escolhida).

1. Flutter (Desenvolvimento Mobile)

Escalabilidade: Suporte a Android, iOS e Web com um único código.
Facilidade de manutenção: Hot reload, código unificado e boas práticas de modularização.
Suporte e treinamento: Documentação oficial rica, comunidade ativa e muitos cursos disponíveis.

2. http (Dart) (Integração com APIs)

Escalabilidade: Leve, ideal para chamadas REST em larga escala.
Facilidade de manutenção: Biblioteca nativa, sem dependências externas.
Suporte e treinamento: Forte documentação oficial e ampla base de exemplos na comunidade Flutter.

3. Firebase Firestore (Banco de Dados)

Escalabilidade: Banco NoSQL gerenciado e em tempo real, com suporte a milhões de conexões.
Facilidade de manutenção: Zero manutenção de infraestrutura e atualizações automáticas.
Suporte e treinamento: Forte documentação oficial (Google) e tutoriais mantidos pela Google e comunidade.

4. Node.js com Firebase Functions (Backend/API)

Escalabilidade: Arquitetura serverless que cresce automaticamente com a demanda.
Facilidade de manutenção: Funções isoladas, atualizações simples e rápidas.
Suporte e treinamento: Forte documentação oficial Node.js, Forte documentação oficial para Firebase (Google)

5. Firebase Authentication (Autenticação)

Escalabilidade: Gerencia autenticação de forma segura e pronta para grande volume de usuários.
Facilidade de manutenção: Integração pronta com Flutter e Firestore, sem necessidade de servidor próprio.

Suporte e treinamento: Forte documentação oficial (Google) e integração com provedores externos.

6. Firebase Cloud Messaging (Push Notifications)

Escalabilidade: Capaz de enviar notificações para milhões de dispositivos simultaneamente.
Facilidade de manutenção: Integração nativa com Flutter e sem necessidade de servidor próprio.
Suporte e treinamento: Suporte oficial e forte documentação (Google)

7. Firebase (Hosting, Firestore, Functions) (Cloud & Hospedagem)

Escalabilidade: Solução unificada e totalmente gerenciada, ideal para MVPs e apps de grande porte.
Facilidade de manutenção: Deploys simples, integração CI/CD, regras de segurança e monitoramento nativo.
Suporte e treinamento: Forte documentação oficial (Google)

8. Google Maps API (Geolocalização)

Escalabilidade: API robusta, confiável e preparada para alto volume de uso.
Facilidade de manutenção: Contrato estável de uso e pacotes prontos para Flutter.
Suporte e treinamento: Forte documentação oficial (Google)

9. Flutter Test + Postman (Testes)

Escalabilidade: Permite criação de suítes de testes automatizados para apps em crescimento.
Facilidade de manutenção: Testes integrados ao ciclo de desenvolvimento, fácil atualização.
Suporte e treinamento: Documentação oficial

10. Firebase Analytics + Sentry (Monitoramento & Analytics)

Escalabilidade: Coleta de dados e logs em larga escala, ideal para apps com muitos usuários.
Facilidade de manutenção: Integração automática com o app e painéis em tempo real.
Suporte e treinamento: Forte documentação oficial (Google)

Abaixo segue explicação de alguns dos termos utilizados que, ou me geraram duvidas, ou podem gerar duvidas para algum membro do grupo. Não há necessidade de incluir em nosso trabalho, serve apenas como aprendizado.

Hot Reload:

É um recurso disponível em frameworks como Flutter que permite atualizar a interface do aplicativo imediatamente após uma alteração no código, sem precisar reiniciar o app inteiro.

Práticas de modularização:

São técnicas e princípios usados para organizar o código em módulos ou partes menores, independentes e reutilizáveis. Isso facilita a leitura, manutenção, atualização e teste do software.

Arquitetura serverless:

Arquitetura serverless é um modelo de desenvolvimento onde o provedor de nuvem gerencia automaticamente a infraestrutura necessária para executar o código, eliminando a necessidade do desenvolvedor se preocupar com servidores, escalabilidade ou manutenção da infraestrutura.

MVPs:

Significa Minimum Viable Product, ou Produto Mínimo Viável em português.

É a versão mais simples e básica de um produto ou aplicativo que contém apenas as funcionalidades essenciais para atender a um problema ou necessidade principal dos usuários. O objetivo do MVP é lançar rapidamente no mercado para testar a aceitação, coletar feedback real dos usuários e validar ideias com o menor investimento possível.

Deploys simples:

Significa que o processo de colocar seu app no ambiente de produção (ou seja, disponível para os usuários) é rápido, automatizado e sem complicações. Você consegue enviar atualizações de código sem muitos passos manuais, com menos risco de erros e downtime.

Integração CI/CD:

CI/CD é a sigla para Continuous Integration (Integração Contínua) e Continuous Delivery ou Continuous Deployment (Entrega ou Implantação Contínua).

Logs:

Logs são registros automáticos que um sistema, aplicativo ou servidor gera para documentar eventos, ações, erros ou qualquer atividade que aconteça durante sua execução.

Eles funcionam como uma espécie de “diário” técnico que ajuda desenvolvedores e administradores a entender o que está acontecendo dentro do software.