

OpSafe – Decisão Arquitetural de Banco de Dados

Versão 1.0

Documento Técnico Empresarial

Autor: Caio César Ponte

Data: 15 de novembro de 2025

Este documento formaliza a decisão técnica sobre a tecnologia de banco de dados a ser utilizada no projeto OpSafe, apresentando o contexto, alternativas avaliadas, análise comparativa, riscos, justificativas e decisão final.

Sumário

1	Contexto Geral	2
2	Requisitos Técnicos e de Negócio	2
2.1	2.1 Requisitos Funcionais do Banco	2
2.2	2.2 Requisitos Não Funcionais	3
3	Alternativas Avaliadas	3
4	Análise Comparativa	3
4.1	4.1 Critérios Usados	3
4.2	4.2 Tabela Comparativa	4
5	Discussão	4
5.1	5.1 MongoDB	4
5.2	5.2 Firestore	5
5.3	5.3 DynamoDB	5
6	Decisão Final	5
6.1	6.1 Justificativas	5
7	Riscos Identificados	5
7.1	7.1 Mitigações	6
8	Resultado	6
8.1	8.1 Próximos passos	6

1 Contexto Geral

O projeto OpSafe é uma plataforma SaaS para gestão operacional de empresas de segurança privada e patrimonial. Seu escopo envolve:

- Controle de inventário de equipamentos (radios, EPIs, coletes, viaturas);
- Rastreamento de movimentações (checkout, checkin, transferência);
- Emissão e auditoria de termos digitais eletrônicos;
- Gestão de operadores e postos de serviço;
- Histórico completo de manutenção preventiva e corretiva;
- Alertas automáticos e trilhas de auditoria;
- Multi-tenant com isolamento por organização.

O banco de dados precisa atender:

- Alta flexibilidade nos documentos e evolução das regras de negócio;
- Modelo rico de consultas transacionais;
- Capacidade de suportar relatórios operacionais (agregações);
- Multi-tenant baseado em `organizationId`;
- Baixo overhead de manutenção;
- Suporte a auditoria histórica extensa;
- Escalabilidade horizontal simples.

O objetivo deste documento é registrar formalmente a escolha tecnológica do banco de dados primário da plataforma.

2 Requisitos Técnicos e de Negócio

2.1 2.1 Requisitos Funcionais do Banco

- Suporte a documentos complexos com subestruturas;
- Facilidade na criação e evolução de coleções;
- Consultas filtradas por múltiplos campos;

- Agregações para relatórios (sum, group, project);
- Capacidade de trabalhar com auditoria (`before/after`);
- Operações de soft delete com filtros globais;
- Consultas multi-tenant eficientes.

2.2 2.2 Requisitos Não Funcionais

- Confiabilidade e redundância;
- Escalabilidade em nuvem;
- Baixo acoplamento ao backend;
- Ferramentas maduras de administração;
- Drivers estáveis em TypeScript;
- Baixo custo operacional inicial.

3 Alternativas Avaliadas

Foram consideradas as seguintes tecnologias NoSQL viáveis:

- MongoDB Atlas
- Google Firestore
- AWS DynamoDB
- PostgreSQL com JSONB (alternativa híbrida)

4 Análise Comparativa

4.1 4.1 Critérios Usados

- Flexibilidade de modelo
- Complexidade de consultas
- Manutenção e custo operacional
- Curva de aprendizado

- Suporte a multi-tenant
- Adequação ao fluxo de auditoria
- Recursos para agregações e relatórios

4.2 Tabela Comparativa

Critério	MongoDB	Firestore	DynamoDB
Modelo de documentos	Excelente	Bom	Bom
Consultas complexas	Excelente	Limitado	Limitado
Agregações / Relatórios	Forte	Fraco	Ausente
Multi-tenant por ID	Simples	Simples	Requer planejamento rígido
Auditoria (before/after)	Natural	Difícil	Difícil
Curva de aprendizado	Baixa	Baixa	Alta
Escalabilidade	Excelente	Muito alta	Muito alta
Custo mínimo	Moderado	Baixo	Baixo
Flexibilidade para evoluir regras	Excelente	Média	Baixa

5 Discussão

5.1 MongoDB

Atende de forma superior aos requisitos principais:

- Flexibilidade alta para evolução incremental das coleções;
- Aggregations poderosas para relatórios operacionais;
- Suporte natural a objetos aninhados;
- Adaptabilidade a documentos grandes, como termos digitais;
- Modelo fortíssimo para auditoria (pode armazenar snapshots grandes sem impacto conceitual);
- Baixa fricção para multi-tenant;
- Mongoose e driver nativo maduros em Node/TypeScript.

5.2 5.2 Firestore

Excelente para aplicações em tempo real, porém possui:

- Limitações severas de query combinada;
- Relatórios complexos exigem reestruturações;
- Nível baixo de suporte para documentos grandes de auditoria.

5.3 5.3 DynamoDB

Extremamente escalável, porém:

- Requer modelagem antecipada dos padrões de acesso;
- Não se adequa ao estágio de exploração e evolução contínua do projeto;
- Auditoria complexa devido ao modelo de chave simples.

6 Decisão Final

Após a análise técnica, o banco de dados escolhido para o OpSafe é:

MongoDB Atlas (Nuvem Gerenciada)

6.1 6.1 Justificativas

- É a solução que melhor atende ao modelo multi-tenant do projeto;
- Permite evolução rápida das regras de negócio;
- Suporta consultas e agregações avançadas necessárias para relatórios;
- Lida bem com documentos grandes e históricos de auditoria;
- Possui ecossistema maduro para TypeScript;
- Reduz complexidade operacional por ser totalmente gerenciado.

7 Riscos Identificados

- **Custo pode crescer** com volume de dados;
- **Indexação incorreta** pode afetar performance;
- **Auditoria extensa** pode gerar coleções muito grandes;

7.1 Mitigações

- Monitoramento e revisão trimestral de índices;
- Particionamento lógico por `organizationId` em consultas intensivas;
- Armazenamento de auditorias frias em buckets externos quando necessário;
- Uso de backups automáticos e política de retenção.

8 Resultado

A decisão técnica está formalizada: ****MongoDB Atlas será o banco oficial do OpSafe****. Esta definição orientará:

- O desenvolvimento do backend;
- A criação dos repositórios de domínio;
- A arquitetura de índices;
- Os padrões de auditoria do sistema.

8.1 Próximos passos

1. Criar cluster MongoDB Atlas (dev, stage e prod);
2. Definir índices iniciais por coleção;
3. Implementar o primeiro repositório (organizations);
4. Criar estrutura base da API com filtros globais por `organizationId` e `isDeleted`.