Justificativa Técnica das Decisões Arquiteturais

Projeto: AgroSolutions IoT Platform

1. Visão Geral da Arquitetura

A solução foi desenhada seguindo o padrão de **Microsserviços**, visando atender aos requisitos de alta escalabilidade e desacoplamento exigidos por um ecossistema de Internet das Coisas (IoT). A divisão da aplicação em contextos delimitados (*Bounded Contexts*) permite que a ingestão de dados massivos dos sensores escale independentemente das rotinas administrativas de cadastro .

2. Stack Tecnológica e Decisões de Design

2.1. Linguagem e Framework: .NET 8

Optou-se pelo uso do **.NET 8 (LTS)** devido à sua robustez, alta performance em cenários de alta concorrência e gerenciamento eficiente de memória, características cruciais para o microsserviço de Ingestão de Dados que deve lidar com múltiplas requisições simultâneas.

2.2. Banco de Dados: MongoDB (NoSQL)

A escolha do **MongoDB** como persistência central atende ao requisito bônus do projeto e se justifica por dois motivos principais:

1. **Flexibilidade de Schema ("Schemaless"):** Dados de telemetria de sensores variam e podem evoluir. O modelo documental permite armazenar payloads JSON heterogêneos sem a rigidez de migrações em tabelas relacionais.
2. **Alta Performance de Escrita:** O MongoDB oferece excelente vazão (*throughput*) para a gravação intensiva de séries temporais geradas pelos sensores. *Nota:* Para simplificar a infraestrutura e reduzir custos operacionais, utilizou-se o MongoDB também para os dados cadastrais (Produtores e Talhões), consolidando a tecnologia de banco de dados.

2.3. Mensageria e Assincronismo: AWS SNS + SQS

Substituímos brokers autogerenciados (como RabbitMQ em container), pela utilização de serviços gerenciados de nuvem (**AWS SNS** e **SQS**) para atender ao requisito de mensageria.

- **Justificativa:** O padrão *Publish/Subscribe* via SNS permite que um evento de sensor seja consumido por múltiplos serviços futuros sem alterar a origem. O uso do SQS garante a resiliência e o desacoplamento: se o *Worker* de processamento falhar, as mensagens persistem na fila, garantindo que nenhum dado do sensor seja perdido (Durabilidade). Além disso, elimina-se os custos da utilização desses brokers

2.4. Orquestração: Kubernetes

A aplicação é containerizada e orquestrada via **Kubernetes** . Isso permite o *Auto-scaling*

horizontal, especificamente para o pod da API de Ingestão (Agro.Ingestion.API), garantindo que o sistema suporta picos de carga sem derrubar os serviços administrativos.

2.5. Observabilidade e Dashboard: Prometheus + Grafana

Para atender aos requisitos de monitoramento, APM e visualização de dados históricos e alertas , implementou-se a stack New Relic, Prometheus e Grafana.

- **Decisão Estratégica:** Em vez de desenvolver um frontend dedicado, utilizou-se o Grafana para visualizar **Métricas de Negócio** (ex: "Nível de Umidade por Talhão"). O *Worker* de análise expõe métricas customizadas que são coletadas pelo Prometheus. Isso centraliza a saúde da infraestrutura e o estado do negócio em um único painel ("Single Pane of Glass").
- Segregamos o monitoramento da infraestrutura no agente New Relic instalado nos containers, cumprindo o requisito da utilização de um APM

2.6. Segurança: Autenticação JWT com RBAC

Implementou-se autenticação via tokens **JWT** (JSON Web Tokens) . Utilizou-se o controle de acesso baseado em roles (RBAC) para segregar a segurança:

- **Role Farmer:** Acesso humano às configurações e dashboards.
- **Role Device:** Acesso sistêmico restrito exclusivamente ao *endpoint* de envio de dados, minimizando a superfície de ataque caso um dispositivo seja comprometido.

Esse JWT é um token gerado por uma chave simétrica HMAC-sha256 pela praticidade, porém o ideal seria substituir por uma lógica que utiliza chaves assimétricas (RS256) para criptografar e descriptografar o token.

Atendimento aos Requisitos Não Funcionais:

- **Escalabilidade e Alta Carga:** A solução foi desenhada utilizando **Kubernetes** para orquestração de containers, permitindo o escalonamento horizontal (HPA) dos microsserviços sob demanda. Para a ingestão massiva de dados IoT, utilizamos o padrão **Fan-Out com AWS SNS e SQS**, garantindo que picos de tráfego sejam absorvidos pelas filas sem degradar a performance da aplicação.

- **Resiliência:** O desacoplamento via mensageria assegura que, em caso de indisponibilidade momentânea do serviço de análise (agtc-srv-analysis), os dados permaneçam seguros na fila (SQS) para processamento posterior, garantindo zero perda de dados (Data Durability).

- **Segurança:** A integridade do acesso é garantida via autenticação **JWT (JSON Web Tokens)** centralizada no serviço agtc-srv-auth.

- **Observabilidade:** A saúde do ecossistema é monitorada via **Prometheus e Grafana**, permitindo a detecção proativa de incidentes e visualização em tempo real das métricas vitais dos talhões.