Descrição do Projeto e Arquitetura da Solução

Integração de Sistemas de Informação

Aluno: 23010 – Hugo Cruz

Aluno: 23016 – Dani Cruz

Professor: Óscar Ribeiro

Licenciatura em Engenharia de Sistemas Informáticos

Barcelos | dezembro, 2025

Índice

Índice de Figuras

1 Introdução

A preservação de vinhos exige condições ambientais muito controladas, uma vez que fatores como a luz, a temperatura e a humidade influenciam diretamente a longevidade do vinho. Tendo isto em conta, o tema que decidimos escolher foi a implementação de um sistema inteligente de monitorização de adegas capaz de recolher, centralizar e apresentar todos os valores obtidos pelos sensores instalados nessas mesmas adegas. Desta forma, torna-se possível saber em que condições se encontra o vinho pretendemos comprar.

Este projeto foi desenvolvido no âmbito da unidade curricular Integração de Sistemas de Informação, lecionada por Óscar Ribeiro e tem como objetivo criar um sistema integrado que combina diferentes tecnologias e camadas de software que fomos aprendendo ao longo do semestre, sendo estas capazes de responder ao tema do nosso trabalho.

A nossa solução inclui um front-end que funciona como ponto de acesso ao sistema permitindo aos utilizadores consultar de forma simples as condições em que os vinhos foram armazenados e também proceder à compra dos mesmos. Todas as interações realizadas no front-end são encaminhadas para uma API, responsável por gerir a comunicação interna da aplicação. Esta API recorre a um serviço SOAP, que atua como intermediário no acesso à nossa base de dados, executando operações de armazenamento e consulta de forma estruturada da mesma.

2 Diagrama ER

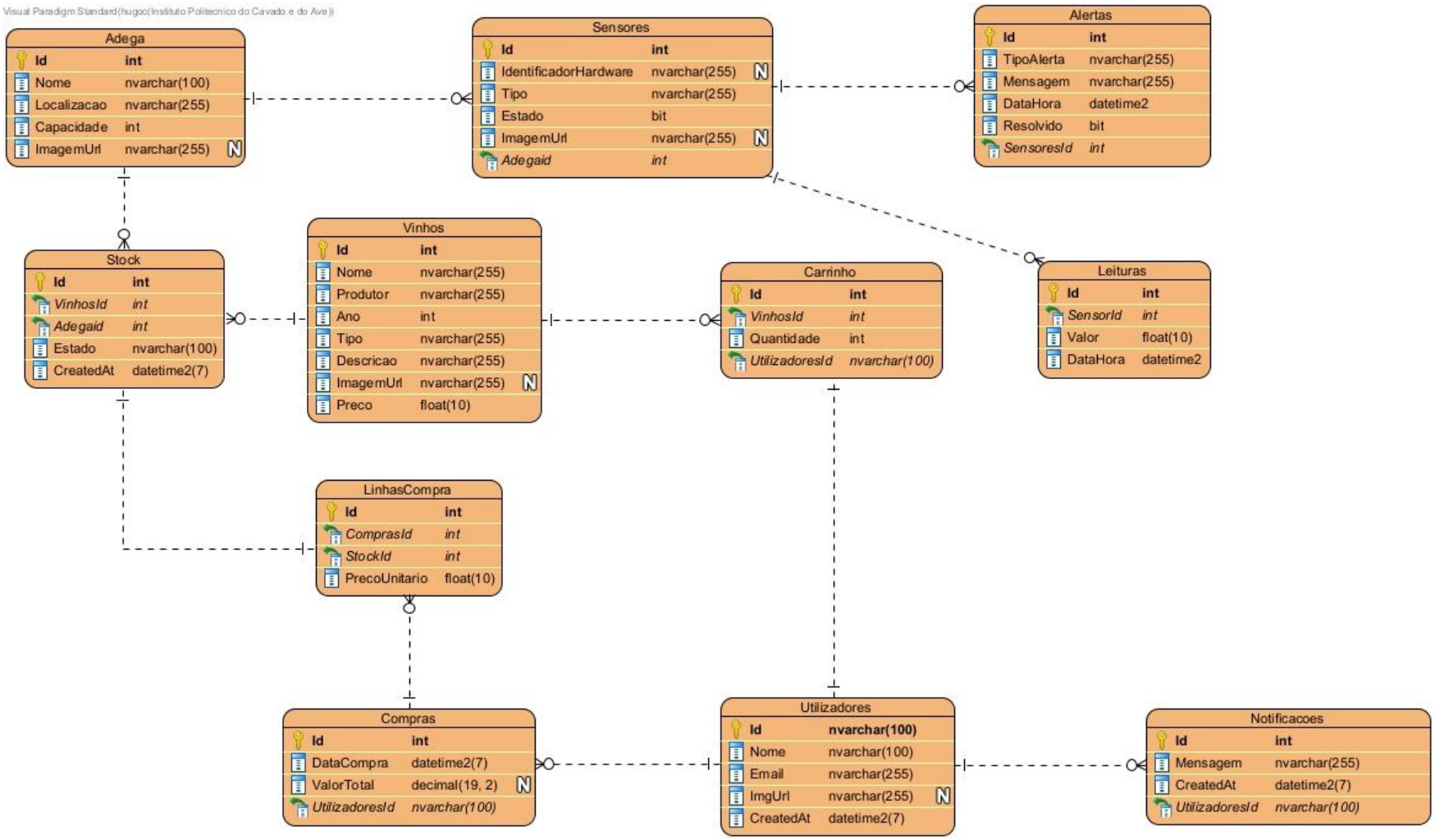

Figura 1 – Diagrama ER

3 Arquiteturado sistema

3.1 Visão Geral

A solução adota uma Arquitetura Orientada a Serviços (SOA) híbrida, desenhada para maximizar a segurança e escalabilidade. O sistema separa estritamente as interfaces públicas (Frontend e API) do acesso aos dados sensíveis, que é gerido exclusivamente por um serviço privado (SOAP).

3.2 Componentes e Responsabilidades

Frontend (Aplicação Web)

A interface visual com a qual o utilizador final interage.

- **Tecnologia:** React.js.
- **Responsabilidade:**
    - Apresentar os dados de forma intuitiva aos utilizadores.
    - Consumir a **API RESTful** para obter e enviar dados (via pedidos HTTP/JSON).
    - Não possui qualquer lógica de negócio crítica nem acesso a bases de dados.

Interface Pública: API RESTful

Este é o ponto central de comunicação e orquestração do sistema.

- **Tecnologia:** ASP.NET Core Web API.
- **Protocolo:** HTTP/HTTPS (JSON).
- **Responsabilidades:**
    - Gateway para Clientes: Serve como ponto de entrada único tanto para a APP IoT como para o Frontend.
    - Orquestração de Lógica (BLL): Contém a camada de Lógica de Negócio que valida regras, processa dados dos sensores IoT e gere o fluxo da aplicação.
    - Consumo de Serviços Externos: Responsável pela integração direta com APIs de terceiros (ex: validador de cartões).
    - Abstração de Dados: A API não comunica diretamente com a Base de Dados. Qualquer necessidade de persistência ou leitura é delegada ao serviço SOAP interno.

Serviço de Dados Privado: SOAP XML (WCF)

Atua como uma "camada de dados remota" e segura.

- **Tecnologia:** ASMX.
- **Protocolo:** SOAP (XML).
- **Visibilidade:** Privada (Deploy em rede interna Azure). Apenas a API REST tem permissão para invocar este serviço.
- **Responsabilidade Única:** Acesso à Base de Dados. Encapsula todas as operações CRUD e é o único componente com credenciais para aceder ao SQL Server.

Sistema de Gestão de Base de Dados (SGBD)

- **Tecnologia:** Microsoft SQL Server (MSSQL).
- **Segurança:** A base de dados aceita conexões apenas do serviço SOAP Privado. Está totalmente isolada da API pública e da internet.

Serviços Externos

Validação de Pagamentos (CreditCardValidator):

- **Endpoint:** https://secure.ftipgw.com/ArgoFire/validate.asmx?WSDL
- **Integração:** A API REST consome este serviço externo para validar compras iniciadas no Frontend React.

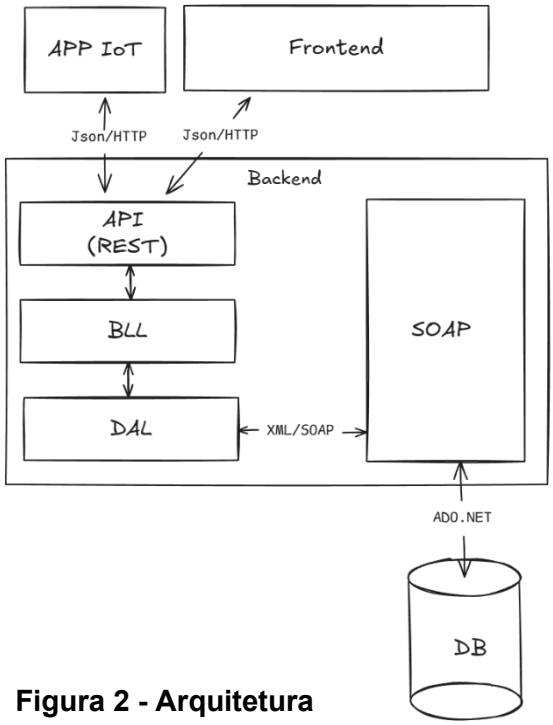

Figura 2 - Arquitetura

4 Identificação das principais rotas

4.1 Gestão de Vinhos (/api/vinhos)

Estas rotas permitem gerir o catálogo de vinhos disponíveis:

- **`GET /vinhos:`** Listar todos os vinhos.
- **`GET /vinhos/{id}:`** Obter detalhes de um vinho específico.
- **`POST /vinhos:`** Criar um novo registo de vinho.
- **`PUT /vinhos/{id}:`** Atualizar informações de um vinho.
- **`DELETE /vinhos/{id}:`** Remover um vinho do catálogo.

4.2 Gestão de Adegas e Stock (/api/adegas)

Focado na gestão física das adegas e no inventário de vinhos dentro delas.

- **`GET /adegas:`** Listar todas as adegas.
- **`GET /adegas/{id}:`** Obter detalhes de uma adega específica.
- **`POST /adegas:`** Registar uma nova adega.
- **`DELETE /adegas/{id}:`** Remover uma adega.
- **`PUT /adegas/{id}:`** Atualizar uma adega.
- **`GET /adegas/{id}/stock:`** Ver o inventário (stock) de uma adega específica (quais vinhos e quantidades).
- **`POST /adegas/{id}/stock:`** Adicionar stock de um vinho a uma adega (Tabela Stock).
- **`PUT /adegas/{id}/stock/{vinhoId}:`** Ajustar a quantidade de stock de um vinho.

4.3 Sensores e Monitorização IoT (/api/sensores)

Para gerir o hardware instalado nas adegas e as leituras ambientais (temperatura, humidade, luz).

- **`GET /sensores:`** Listar sensores (pode filtrar por adega).
- **`POST /sensores:`** Registar um novo sensor e associá-lo a uma adega (AdegaId).
- **`GET /sensores/{id}/leituras:`** Obter o histórico de leituras de um sensor específico (Tabela Leituras).
- **`POST /leituras:`** Rota para o hardware enviar dados (recebe SensorId, Valor, DataHora).
- **`GET /sensores/{id}/alertas:`** Ver alertas gerados por um sensor específico.

4.4 Gestão de Compras e Carrinho (/api/compras & /api/carrinho)

Fluxo de e-commerce para os utilizadores comprarem vinhos.

Carrinho de Compras:

- **`GET /carrinho:`** Obter o carrinho do utilizador atual.
- **`POST /carrinho:`** Adicionar um item (VinhosId, Quantidade) ao carrinho.
- **`PUT /carrinho/{itemId}:`** Atualizar a quantidade de um item.
- **`DELETE /carrinho/{itemId}:`** Remover um item do carrinho.

Processamento de Compras:

- **`POST /compras:`** Finalizar a compra (transforma o Carrinho em Compras e LinhasCompra, e decrementa o Stock).
- **`GET /compras:`** Listar histórico de compras do utilizador logado.
- **`GET /compras/{id}:`** Ver detalhes de uma compra específica (itens comprados).

4.5 Utilizadores e Notificações (/api/utilizadores)

Gestão de perfil e comunicação com o utilizador.

- **`GET /utilizadores/perfil:`** Obter dados do utilizador logado (baseado no Auth0UserId).
- **`PUT /utilizadores/perfil:`** Atualizar dados do perfil.
- **`GET /notificacoes:`** Listar notificações não lidas do utilizador.
- **`PUT /notificacoes/{id}/lida:`** Marcar uma notificação como lida/resolvida.