POLITÉCNICO DE SANTARÉM
ESCOLA SUPERIOR DE GESTÃO E TECNOLOGIA

# Trabalho prático Individual

Desenvolvimento de Serviços Web Multitecnologia

Disciplina de Integração de Sistemas

## Resumo das funcionalidades implementadas por serviço

| Serviço | Funcionalidade | Observações |
|---|---|---|
| **SOAP** | Listar produtos | Utiliza getProdutos() |
| | Adicionar produto | Validação com **XSD** antes de guardar |
| | Editar produto | Validação com **XSD** após edição |
| | Remover produto | Atualiza o XML e mantém formatação com minidom |
| | Validação XSD | Via lxml.etree.XMLSchema |
| **REST** | Listar produtos | GET /produtos |
| | Obter produto por ID | GET /produtos/<id> |
| | Adicionar produto | POST /produtos com validação JSON Schema |
| | Editar produto | PUT /produtos/<id> com validação JSON Schema |
| | Remover produto | DELETE /produtos/<id> |
| | Consulta com JSONPath | GET /consulta?q=<jsonpath> |
| | Exportação para JSON e XML | Guardado em /cliente/exportados |
| | Importação de JSON e XML | A partir de /cliente/importar |
| **GraphQL** | Listar produtos | Query: produtos |
| | Adicionar produto (mutation) | Mutation: adicionarProduto(...) com validação JSON Schema |
| | Editar produto (mutation) | Mutation: editarProduto(...) |
| | Remover produto (mutation) | Mutation: removerProduto(id) |
| **gRPC** | Listar produtos (streaming) | ListarProdutosStream |
| | Listar produtos (lista única) | ListarProdutos |
| | Adicionar produto | AdicionarProduto (unário) |
| | Editar produto | EditarProduto (unário) |
| | Remover produto | RemoverProduto (unário) |

## Tema escolhido

Tema – Catálogo de produtos eletrónicos

## Formato de dados escolhido

Os serviços REST, GraphQL e gRPC utilizam como fonte de dados comum um ficheiro produtos.json, localizado na pasta /servidor/dados, permitindo uma gestão unificada dos dados nesses serviços.

O serviço SOAP, por sua vez, utiliza um ficheiro separado (produtos.xml), localizado em /servidor/soap, estruturado em XML e validado com um ficheiro schema.xsd.

Esta separação garante a independência tecnológica entre os serviços e permite aplicar diferentes formatos de dados e mecanismos de validação (JSON Schema para REST e GraphQL, e XSD para SOAP.).

## Implementação

O projeto é orquestrado com um ficheiro docker-compose.yml, que permite levantar todos os serviços de forma coordenada. Cada serviço possui o seu próprio Dockerfile, responsável por definir a imagem do serviço.

Em cada pasta de serviço, encontra-se um ficheiro requirements.txt, executado a partir do Dockerfile, que garante a instalação das dependências necessárias ao funcionamento do serviço.

A utilização de containers permite um ambiente isolado e reproduzível, facilitando a execução do projeto em qualquer máquina com Docker.

Todos os serviços e o cliente foram desenvolvidos em Python, com recurso às seguintes bibliotecas:

- Flask e requests para o serviço REST;
- Graphene para GraphQL;
- Spyne (servidor) e zeep (cliente) para SOAP;
- gRPC + Protocol Buffers para o serviço gRPC.

## Funcionalidades do cliente multitecnologia

O cliente desenvolvido para este projeto tem como objetivo centralizar a interação com os diferentes serviços disponibilizados (REST, SOAP, GraphQL e gRPC), oferecendo uma interface unificada para a execução de operações sobre o catálogo de produtos.

A implementação recorre à linguagem Python e apresenta um menu interativo que guia o utilizador na seleção do serviço e da operação desejada.

**Funcionalidades Disponíveis:**

- Obter a lista de produtos

O cliente permite visualizar a lista de produtos disponíveis em qualquer um dos serviços. A resposta é adaptada ao formato específico de cada tecnologia, mantendo uma apresentação uniforme ao utilizador.

- Adicionar novos produtos

É possível adicionar novos produtos através dos quatro serviços. Os dados são validados antes da inserção, conforme os mecanismos específicos de cada tecnologia:

REST: validação através de JSON Schema;

SOAP: validação com recurso a ficheiro XSD;

GraphQL: validação via schema definido nas mutations;

gRPC: validação realizada no servidor antes de adicionar o produto.

- Edição Interativa dos produtos

A edição de produtos é realizada de forma interativa, permitindo o utilizador modificar apenas os campos desejados e mantendo os restantes inalterados. Esta funcionalidade encontra-se disponível em todos os serviços.

- Remover produtos

O cliente permite a remoção de produtos com base no respetivo ID. Esta operação está disponível em todos os serviços.

- Exportar a lista de produtos

O cliente suporta a exportação da lista de produtos nos seguintes formatos:

JSON: estruturado, indentado e ordenado para melhor legibilidade;

XML: com estrutura hierárquica clara e indentação adequada.

- Importação de uma lista de produtos

É possível importar produtos a partir de ficheiros:

JSON: validado com JSON Schema;

XML: processado e transformado em objetos produto, respeitando a estrutura exigida.

- Consultas com JSONPath (REST)

Através de um endpoint específico do serviço REST, o cliente permite realizar consultas com expressões JSONPath, proporcionando assim uma maior flexibilidade na filtragem dos dados. Exemplos de consultas possíveis:

$[*].nome – Obtém os nomes de todos os produtos;

$[?(@.preco < 100)] – Filtra os produtos com preço inferior a 100€.

Trabalho Realizado por:

Duarte Canoso, 220001613