 

Relatório

Processamento Estruturado de Informação

2º Ano – 1º Semestre

Afonso Santos 8150025 - LSIRC

Nuno Josefino 8150189 - LEI

Hugo Silva 8130142 - LEI



# Índice

[Índice 2](#_Toc530936363)

[Introdução 3](#_Toc530936364)

[User details 4](#_Toc530936365)

[Currency details 5](#_Toc530936366)

[Product Details 7](#_Toc530936367)

[Sales Details 9](#_Toc530936368)

[Store Details 12](#_Toc530936369)

[BikeOnTrack XML 14](#_Toc530936370)

[Conclusão 15](#_Toc530936371)

# Introdução

Este trabalho surge no âmbito da disciplina de Processamento Estruturado de Informação e tem como objetivo a criação de uma estrutura XML que permita unificar a maneira como os dados são registados em todas as lojas da empresa BikeOnTrack.

De modo a obter esta uniformização dos dados, foi desenvolvido um conjunto de XML schemas que valide a informação inserida e que garanta que as informações são estruturadas da mesma maneira em todas as lojas.

Estes XML schemas foram criados tendo por base os ficheiros .cvs fornecidos pela empresa e como tal devem envolver toda a informação relacionada com as lojas que é necessária.

# User details

Este ficheiro .xsd consiste numa validação de dados do cliente que efetua a compra , tem como elemento “user” composto pelos elementos “userID” e “name”.

O elemento userID é necessário para os dados referentes ás vendas.

**Namespaces:**

Este ficheiro não necessita de importar nenhum outro ficheiro e como tal não recorre há utilização de namespaces nos seus elementos.

**Elementos:**

**userID** – este elemento consiste num simples ID e como tal apenas requer que seja um numero inteiro e positivo.

**name** – este elemento consiste no nome do cliente e apenas requer que seja uma string.

# Currency details

Este ficheiro .xsd consiste na informação referente á tabela CurrencyDetails e serve para organizar e validar a informação nela contida. O elemento principal “currency” é composto por elementos referentes a todos os valores contidos na tabela.

**Namespaces:**

Este ficheiro não necessita de importar nenhum outro ficheiro e como tal não recorre há utilização de namespaces nos seus elementos.

**Elementos:**

**currencyID -** este elemento consiste num simples ID e como tal apenas requer que seja um numero inteiro e positivo.

**currencyRateDate** – este elemento indica a data em que foi efetuada a conversão da moeda e requer que seja uma data do tipo dateTime.

**fromCurrencyCode** – este elemento indica o código da moeda que se vai ter por base para a conversão. Este código é constituído por 3 letras requerendo apenas que seja uma string.

**toCurrencyCode** – este elemento indica o código da moeda para a qual se vai realizar a conversão. Este código é constituído por 3 letras requerendo apenas que seja uma string.

**rateValue –** este elemento indica o valor da moeda do pais referente ao “toCurrencyCode” em relação ao pais referente ao “fromCurrencyCode”.

O valor será um decimal positivo.

# Product Details

Este ficheiro .xsd consiste na informação referente á tabela ProductDetails e serve para organizar e validar a informação nela contida. O elemento principal “products” é composto por elementos referentes a todos os valores contidos na tabela.

**Namespaces:**

Este ficheiro não necessita de importar nenhum outro ficheiro e como tal não recorre há utilização de namespaces nos seus elementos.

**Elementos:**

**prodID -** este elemento consiste num simples ID e como tal apenas requer que seja um numero inteiro e positivo.

**name** – este elemento consiste no nome do produto e apenas requer que seja uma string.

**productNumber** – este elemento consiste no numero pelo qual o produto é identificado. Este numero é constituido por letras e numeros, pelo que, requer que seja uma string.

**color** – este elemento serve para identificar a cor do produto e apenas requer que seja uma string.

**listPrice** – este elemento indica qual o preço listado do produto, sendo que este valor será um decimal positivo.

**sellStartDate** – este elemento indica a data em que foi iniciada a venda e requer que seja uma data do tipo dateTime.

**sellEndDate** – este elemento indica a data em que foi finalizada a venda e requer que seja uma data do tipo dateTime.

# Sales Details

Este ficheiro .xsd consiste na informação referente á tabela SalesDetails e serve para organizar e validar a informação nela contida. O elemento principal “sales” é composto por elementos referentes a todos os valores contidos na tabela. Este ficheiro necessita de informação proveniente de outros ficheiros e como tal usa imports de modo a poder reutilizalos.

**Namespaces:**

Uma vez que este ficheiro utiliza informações utilizadas e definidas em ficheiros anteriores, recorremos á ultização de imports para os reutilizar e a utilização de namespaces para que não haja conflito entre os elementos.

Utilizamos o “:p” para caracterizar elementos do schema productDetails, o “:c” para elementos do schema currencyDetails e o “:u” para elementos do schema userDetails.

**Elementos:**

**reciptID** – este elemento consiste num simples ID e como tal apenas requer que seja um numero inteiro e positivo.

**orderDate** – este elemento indica a data em que foi feita a encomenda e requer que seja uma data do tipo dateTime.

**costumer** – este elemento refere-se ao ID do cliente, que já foi definido em “userDetails” e como tal é do tipo “u:userID”.

**currencyRateID** – este elemento refere-se ao ID da conversão de moeda, que já foi definido em “currencyDetails” e como tal é do tipo “c:currencyID”.

**subTotal** – este elemento refere-se ao subtotal do valor da venda, sendo que este valor será um decimal positivo.

**taxAmt** – este elemento refere-se ao valor da taxa associada á venda, sendo que este valor será um decimal positivo.

**store** – este elemento indica o ID da loja que efetuou a venda, apenas requer que seja um numero inteiro e positivo.

**storeN** – este elemento indica o nome da loja que efetuou a venda, apenas requer que seja uma string.

**reciptLineID** – este elemento consiste num simples ID e como tal apenas requer que seja um numero inteiro e positivo.

**order** – este elemento indica o numero de encomendas do produto que foram efetuada e apenas requer que seja um numero inteiro e positivo.

**productID** – este elemento refere-se ao ID do produto, que já foi definido em “productDetails” e como tal é do tipo “p:prodID”.

**unitPrice** – este elemento indica o valor unitário do produto que foi vendido, sendo que este valor será um decimal positivo.

**lineTotal** – este elemento indica o valor total de tudo o que foi vendido naquela encomenda, sendo que este valor será um decimal positivo.





# Store Details

Este ficheiro .xsd consiste na identificação das lojas BikeOnTrack, tendo como elemento “store” composto pelos elementos referentes à sua identificação. Este ficheiro necessita de informação proveniente de outros ficheiros e como tal usa imports de modo a poder reutilizá-los.

**Namespaces:**

Uma vez que este ficheiro utiliza informações utilizadas e definidas em ficheiros anteriores, recorremos á ultização de imports para os reutilizar e a utilização de namespaces para que não haja conflito entre os elementos.

Utilizamos o “:s” para caracterizar elementos do schema salesDetails.

**Elementos:**

**storeID** – este elemento refere-se ao ID da loja, que já foi definido em “salesDetails” e como tal é do tipo “s:store”.

**storeName** – este elemento refere-se ao nome da loja, que já foi definido em “salesDetails” e como tal é do tipo “s:storeN”.

**nif** – este elemento indica o nif de uma loja. Este código é constituído por 9 números.

**address** – este elemento indica o adereço da loja, apenas requer que seja uma string.

**contacts** – este elemento indica o contacto da loja. É constituído por 9 números e obrigatoriamente tem que ter um valor definido.

**salesInfo** – este elemento refere-se às encomendas da loja, que já foi definido em “salesDetails” e como tal é do tipo “s:sales”.



# BikeOnTrack XML

Este ficheiro .xml tem como propósito a facilidade de compartilhar a informação armazenada via internet.

Foi então construído um documento exemplo com dados que permitam assim experimentar o vocabulário desenvolvido.



# Conclusão

## No presente relatório elaborou-se um conjunto de XML Schemas capazes de validar todas as regras de sintaxe definidas para a linguagem e tipos associados. Foi também elaborado um documento XML, com dados exemplo, permitindo testar a viabilidade do vocabulário construído.

Estamos numa época em que a Informação é um bem bastante precioso e a partilha e controlo dos dados nem sempre é realizada da melhor maneira. É extremamente importante que os detentores de quaisquer dados possuam mecanismos de armazenamento e exploração organizada dos mesmos. Assim, com o propósito de auxiliar a empresa BikeOnTrack no compartilhamento dos seus dados, podemos concluir que a utilização de um vocabulário XML é uma forma eficaz de gerir uma grande quantidade de dados.