



# RELATÓRIO TP1

## Stand Automóvel

Engenharia de Dados e Conhecimento

Universidade de Aveiro

Departamento de Eletrónica, Telecomunicações e Informática  
Mestrado Integrado em Engenharia de Computadores e Telemática  
2020/2021

Eduardo Coelho (88867)

Joaquim Ramos (88812)

Rúben Menino (89185)

eduardoluis33@ua.pt

joaquimramos@ua.pt

ruben.menino@ua.pt

# Índice

Introdução ao Tema	2
Dados e suas Fontes	3
Esquema dos dados (XML Schema)	5
Transformações sobre os dados (XSLT)	8
Operações sobre os dados (XQuery e XQuery Update)	9
Funcionalidades da Aplicação (UI)	10
Conclusões	13
Configuração para Executar a Aplicação	14

# Introdução

O relatório descreve uma aplicação web efetuada no âmbito da unidade curricular de Engenharia de Dados e Conhecimento. Neste primeiro trabalho pretendeu-se desenvolver uma aplicação web com um tema à escolha com o objetivo de utilizar todas as tecnologias lecionadas, usando o XML que se tornou útil porque permitiu-nos definir uma linguagem específica para descrever os dados dos anúncios, XQueries para pesquisa e gestão dos dados na base de dados e muito mais outras tecnologias.

Esta aplicação web é uma plataforma com a intuição de parecer um stand automóvel e que permite a visualização e manipulação de anúncios de vendas de veículos.

A informação dos anúncios disponíveis são: a marca, modelo, combustível, quilómetros, ano e mês de registo, preço e muitas mais características que são relacionadas tanto com os automóveis, como os seus vendedores.

# Dados e suas Fontes

Após procurarmos por fontes de dados XML sobre o tema em questão, sem sucesso, decidimos construir o nosso ficheiro XML que especificasse os dados dos anúncios.

Exemplo do ficheiro XML :

```
<?xml version="1.0" encoding="utf-8"?>
<anuncios xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="anuncios.xsd">
  <anuncio>
    <id>1</id>
    <venda>naovendido</venda>
    <preco>22000</preco>
    <nome>João</nome>
    <contacto>234234234</contacto>
    <email>utilizador1@email.pt</email>
    <marca>renault</marca>
    <modelo>megane</modelo>
    <cilindrada>2000</cilindrada>
    <combustivel>gasoleo</combustivel>
    <transmissao>>manual</transmissao>
    <registo>2017</registo>
    <mes>maio</mes>
    <estado>usado</estado>
    <potencia>220</potencia>
    <quilometros>7000</quilometros>
    <numportas>5</numportas>
    <lotacao>5</lotacao>
    <imagens><imagem>1.png</imagem></imagens>
    <descricao>Carro cinco estrelas, muito bem mantido, venha ver e apreciar.</descricao>
  </anuncio>
  <anuncio>
    <id>2</id>
    <venda>naovendido</venda>
    <preco>8950</preco>
    <nome>Nuno Bruno</nome>
    <contacto>234234600</contacto>
    <email>utilizador2@email.pt</email>
    <marca>citroen</marca>
    <modelo>berlingo</modelo>
    <cilindrada>1600</cilindrada>
    <combustivel>gasoleo</combustivel>
    <transmissao>>manual</transmissao>
    <registo>2015</registo>
    <mes>maio</mes>
    <estado>usado</estado>
    <potencia>100</potencia>
    <quilometros>98000</quilometros>
    <numportas>3</numportas>
    <lotacao>3</lotacao>
```

```
<imagens><imagem>20.png</imagem></imagens>  
<descricao>Preço negociável, com a/c manual, não fumador, tem gps, estofos em tecido, duas chaves originais.</  
descricao>  
</anuncio>  
</anuncios>
```

Os dados das notícias provêm de um Feed RSS disponibilizado pelo website <https://www.autoblog.com/>.  
Optámos por este Feed RSS, pelo facto da informação estar relacionada com o mundo automóvel, que é a base do nosso projeto.

# Esquema dos dados (XML Schema)

A estrutura dos dados foi inicialmente gerada a partir do ficheiro XML, com os dados a serem usados na aplicação, através da ferramenta do PyCharm que permite gerar os ficheiros XSD. Após análise do esquema gerado concluímos que ainda não estava completamente de acordo com a estrutura dos dados requerida para manter a integridade dos mesmos, e por isso foi alterada até chegar à versão que é aqui apresentada:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified" xmlns:xs="http://www.w3.org/2001/XMLSchema"
a">
  <xs:element name="anuncios" type="anunciosType"/>
  <xs:complexType name="imagensType">
    <xs:sequence>
      <xs:element type="xs:string" name="imagem"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="anuncioType">
    <xs:sequence>
      <xs:element type="xs:integer" name="id"/>
      <xs:element name="venda">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="vendido"/>
            <xs:enumeration value="naovendido"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
      <xs:element type="xs:integer" name="preco"/>
      <xs:element type="xs:string" name="nome"/>
      <xs:element name="contacto">
        <xs:simpleType>
          <xs:restriction base="xs:integer">
            <xs:pattern value="[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
      <xs:element type="xs:string" name="email"/>
      <xs:element type="xs:string" name="marca"/>
      <xs:element type="xs:string" name="modelo"/>
      <xs:element type="xs:integer" name="cilindrada"/>
      <xs:element name="combustivel">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="gasoleo"/>
            <xs:enumeration value="gasolina"/>
            <xs:enumeration value="híbrido(gasolina)"/>
            <xs:enumeration value="híbrido(gasoleo)"/>
            <xs:enumeration value="elétrico"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>

```

```

</xs:element>
<xs:element name="transmissao">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:enumeration value="manual"/>
      <xs:enumeration value="automático"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
<xs:element name="registro">
  <xs:simpleType>
    <xs:restriction base="xs:integer">
      <xs:pattern value="[0-9][0-9][0-9][0-9]"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
<xs:element name="mes">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:enumeration value="janeiro"/>
      <xs:enumeration value="fevereiro"/>
      <xs:enumeration value="março"/>
      <xs:enumeration value="abril"/>
      <xs:enumeration value="maio"/>
      <xs:enumeration value="junho"/>
      <xs:enumeration value="julho"/>
      <xs:enumeration value="agosto"/>
      <xs:enumeration value="setembro"/>
      <xs:enumeration value="outubro"/>
      <xs:enumeration value="novembro"/>
      <xs:enumeration value="dezembro"/>
      <xs:enumeration value="Janeiro"/>
      <xs:enumeration value="Fevereiro"/>
      <xs:enumeration value="Março"/>
      <xs:enumeration value="Abril"/>
      <xs:enumeration value="Maio"/>
      <xs:enumeration value="Junho"/>
      <xs:enumeration value="Julho"/>
      <xs:enumeration value="Agosto"/>
      <xs:enumeration value="Setembro"/>
      <xs:enumeration value="Outubro"/>
      <xs:enumeration value="Novembro"/>
      <xs:enumeration value="Dezembro"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
<xs:element name="estado">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:enumeration value="usado"/>
      <xs:enumeration value="novo"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>

```

```

        </xs:restriction>
    </xs:simpleType>
</xs:element>
<xs:element type="xs:integer" name="potencia"/>
<xs:element type="xs:integer" name="quilometros"/>
<xs:element type="xs:integer" name="numportas"/>
<xs:element type="xs:integer" name="lotacao"/>
<xs:element type="imagensType" name="imagens"/>
<xs:element type="xs:string" name="descricao"/>
</xs:sequence>
</xs:complexType>
<xs:complexType name="anunciosType">
    <xs:sequence>
        <xs:element type="anuncioType" name="anuncio" maxOccurs="unbounded" minOccurs="0"/>
    </xs:sequence>
</xs:complexType>
</xs:schema>

```

É com esta versão que são validados os dados que são inseridos pelo o utilizador na aplicação, na página de criar anúncio, se os dados estiverem de acordo com a estrutura são inseridos na base de dados, se não estiverem é lançado um erro na página a notificar o utilizador para inserir os dados da forma correta.



## Transformações sobre os dados (XSLT)

Houve três instâncias neste projeto, onde decidimos utilizar a linguagem XSL para a transformação de XML em XHTML devido à sua utilidade e adequação ao objetivo que pretendíamos.

Numa primeira instância, utilizamos XSLT para apresentar os anúncios na página principal. Primeiramente, efetuamos uma query ao BaseX que nos devolveu todos os anúncios resultantes dos possíveis filtros aplicados. Depois aplicamos ao resultado da query o 'anuncios.xsd', que percorre com um for-each os anúncios encontrados e apresenta-os da forma que achamos mais elegante.

A segunda utilização foi para apresentar as páginas individuais de cada anúncio, fazendo uma query que pesquisa todos os elementos de um anúncio com um certo id apresentando-os na página.

Por fim, fizemos um pequeno snippet para apresentar os 6 carros mais caros que ainda não foram vendidos. Esta aplicação de XSLT foi apenas feita com um intuito de complementar o tema, dando uma perspectiva mais realista ao website, destacando alguns anúncios com um suposto objetivo de promover a venda dos automóveis.

Para obter as notícias a partir do feed RSS fizemos uma pesquisa nos elementos <item> onde obtivemos as informações do título, imagem e outras informações da notícia.

Usamos a biblioteca ElementTree, com o Xpath para obter os dados vindo do Feed RSS.

# Operações sobre os dados (XQuery e XQuery Update)

De forma a permitir o funcionamento da aplicação é necessário realizar queries para recolher, pesquisar, inserir, remover e alterar informação. É exposto de seguida algumas das queries que permitiram o funcionamento da aplicação:

Query que faz uso de todos os campos de filtragem, e retorna os veículos de acordo com os parâmetros passados pelo utilizador, quando não está algum dos campos filtragem selecionado, são feitas outras queries sem esses campos.

```
<root> {  
  for $d in collection('anuncios')//anuncio  
  where $d/marca="\\" + str(  
    selectedmarca).lower() + "\\" and $d/modelo =\\" + str(  
    selectedmodelo).lower() + "\\" and $d/combustivel =\\" + str(  
    selectedcombustivel).lower() + "\\" and $d/preco <" + str(selectedpreco) + " and $d/quilometros <" + str(  
    selectedquilometros) + " and $d/registo >" + str(  
    selectedregisto) + "  
  return <elem> {$d/id} {$d/preco} {$d/imagens/imagem} {$d/modelo} {$d/marca} </elem>} </root>
```

Query que pesquisa pelos seis veículos mais caros ainda não vendidos.

```
let $tmp:=(for $d in collection('anuncios')//anuncio  
where $d/venda = 'naovendido' order by number($d/preco) descending return $d )  
  
return <root>{ let $b := $tmp[position()<7] for $f in $b return <elem> {$f/id} {$f/marca} {$f/modelo}  
{$f/preco} {$f/registo} {$f/transmissao} {$f/quilometros} {$f/imagens/imagem} </elem> }</root>
```

Query que atualiza o estado do veículo para vendido.

```
for $d in collection('anuncios')//anuncio  
where $d/id='"+id+"'<br>return replace node $d/venda/text() with 'vendido'
```

Query que apaga o veículo com o id especificado pelo utilizador.

```
for $d in collection('anuncios')//anuncio  
where $d/id='"+id+"'<br>return delete node $d
```

Query que retorna todas as marcas distintas de veículos disponíveis na base de dados.

```
<root> { for $d in distinct-values(doc('anuncios')//marca)  
return <elem> {$d} </elem>} </root>
```

Query que insere o anúncio com os valores especificados pelo utilizador, após o último anúncio na base de dados, só após serem validados pelo schema.

```
let $d := doc('anuncios')//anuncio[last()] \  
return insert nodes ( \  
  <anuncio> \  
    <id>" + str(res) + "</id>\<br>  
    <venda>naovendido</venda> \  
    <preco>" + str(preco) + "</preco> \  
    <nome>" + str(nome) + "</nome>\<br>  
    <contacto>" + str(contacto) + "</contacto>\<br>  
    <email>" + str(email) + "</email>\<br>  
    <marca>" + str(marca).lower() + "</marca>\<br>  
    <modelo>" + str(modelo).lower() + "</modelo>\<br>  
    <cilindrada>" + str(cilindrada) + "</cilindrada>\</anuncio> )
```

```
<combustivel>" + str(combustivel) + "</combustivel>\n
<transmissao>" + str(transmissao) + "</transmissao>\n
<registro>" + str(registo) + "</registro>\n
<mes>" + str(mes) + "</mes>\n
<estado>" + str(estado) + "</estado>\n
<potencia>" + str(potencia) + "</potencia>\n
<quilometros>" + str(quilometros) + "</quilometros>\n
<numportas>" + str(numportas) + "</numportas>\n
<lotacao>" + str(lotacao) + "</lotacao>\n
<imagens><imagem>" + str(res) + ".png</imagem></imagens>\n
<descricao>" + str(descricao) + "</descricao>\n
    </anuncio>\n
) after $d
```

# Funcionalidades da Aplicação (UI)

Com o decorrer da realização do projeto, tentámos projetar uma interface simples e objetiva que não saísse muito fora do comum e que fosse de fácil interpretação.

Depois de executada a aplicação, é apresentada a página principal do projeto, contendo bastante informação e ações que são necessárias fazer para mostrar o funcionamento dela.

Começamos logo por ver o início da página inicial, com o respetivo logótipo, seguido de um slider de 6 carros com os carros premium disponíveis. Carros estes que estão atribuídos a uma query para apresentar os 6 carros mais caros.

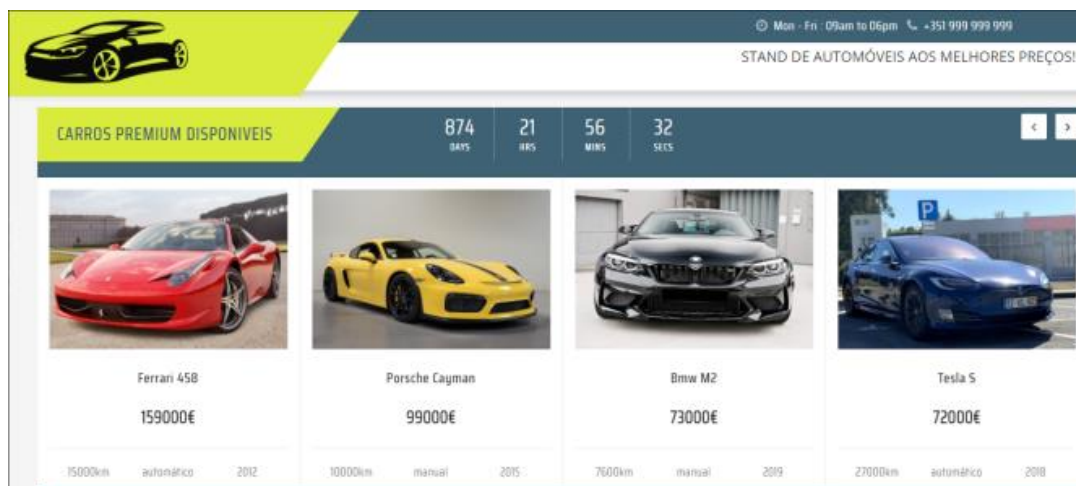


Figura 1 – lista de carros mais caros

Mais em baixo, temos uma lista com todos os carros onde podemos ver algumas das suas informações. Caso queiramos saber todas as informações dum anúncio, incluindo a do respetivo vendedor, basta clicar no mesmo, que irá ser redirecionado para a página individual do anúncio através do id.

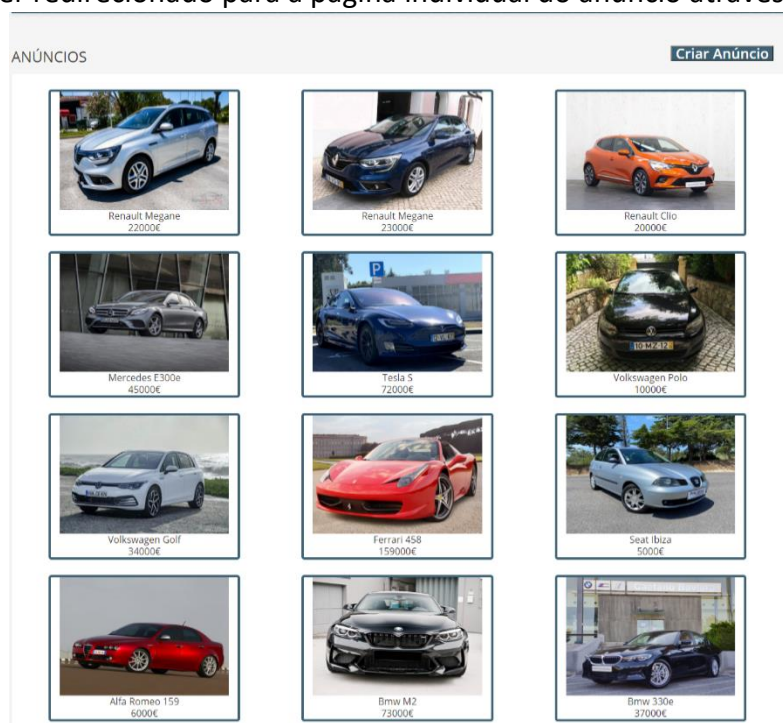


Figura 2 – lista com todos os anúncios do website

Dentro de cada anúncio podemos optar por comprar ou apagar o respetivo carro. Apagando o carro, ele irá ser eliminado da lista dos carros. Se a opção for comprar, é-nos informado que esse carro está vendido.



**Citroen Berlingo**

**8950€** [Comprar](#) [Apagar](#)

**Dados do vendedor:**  
Nome: Nuno Bruno  
Email: pepino@limonada.pt  
Contacto Telefónico (+351): 234234600

**Mais informações:**  
Estado de Venda: Não Vendido  
Combustível: gasoleo  
Cilindrada: 1600cc  
Tipo de transmissão: manual  
Ano e mês de registo: maio de 2015  
Estado do automovel: usado  
Potencia: 100cv  
Quilómetros: 98000km  
Número de portas: 3  
Lotação: 3

**Descrição do Vendedor:**  
Preço negociável, com a/c manual, não fumador, tem gps, estofos em tecido, duas chaves originais.

Figura 3 – informações detalhadas do anúncio

Seguidamente, podemos ver os filtros de procura para os carros na base de dados.

Com estes filtros, conseguimos de forma rápida e eficaz, obter os carros que se relacionam com essa pesquisa. Temos à disposição vários dropdowns e textboxes onde podemos escolher e limitar, por exemplo, um preço máximo. Depois de feita a escolha, basta clicar no botão “procurar veículo” que irá filtrar os veículos à nossa filtragem de dados.

### ENCONTRE O SEU AUTOMÓVEL

Marca...

Modelo...

Combustível...

Quilómetros (abaixo de:)

Registo (acima de:)

Preço (abaixo de:)


PROCURAR VEÍCULO

Figura 4 – filtro de pesquisa


Podemos também criar um anúncio adicionando-o à lista de anúncios ao clicar no botão “Criar Anúncio”.

ANÚNCIOS


[Criar Anúncio](#)



Renault Megane  
22000€



Renault Megane  
23000€



Renault Clio  
20000€

Figura 5 – botão “criar anúncio” para oferecer a possibilidade de criar um novo anúncio

Para completar a criação, basta preencher um simples form com todas as informações do veículo, como marca, modelo... e ainda escolher uma imagem, que ficará como imagem base do anúncio, caso os dados que foram inseridos não estejam de acordo com a estrutura o utilizador é notificado para inserir os dados da forma correta.

Dados da viatura:


Marca: <input type="text"/>	Combustível: <input type="text" value="Gasoleo"/>
Modelo: <input type="text"/>	Transmissão: <input type="text" value="Manual"/>
Cilindrada (cc): <input type="text"/>	Mês do Registo: <input type="text" value="Janeiro"/>
Ano do Registo: <input type="text"/>	Estado: <input type="text" value="Novo"/>
Potência (cv): <input type="text"/>	Image: <input type="text" value="Escolher ficheiro"/> Nenhum ficheiro selecionado
Quilómetros (km): <input type="text"/>	Seus dados:
Numero de portas: <input type="text"/>	Seu nome: <input type="text"/>
Lotação: <input type="text"/>	Email: <input type="text"/>
Preço: <input type="text"/>	Contacto Telefónico (+351): <input type="text"/>
Descrição (opcional): <input type="text"/>	

**Criar Anúncio**

Figura 6 – form do “criar anúncio” para criar o anúncio detalhado


Para ser possível a visualização de notícias necessitamos de criar uma função que retorna o resultado esperado a partir do URL do respetivo RSS. Os elementos title, link e description correspondem aos elementos padrão do RSS <title>, <link> e <description>.

Latest News




2021 Kia Sorento price creeps upward with the redesign

**1** READ MORE



California offering \$1,500 point-of-sale rebate on electric cars and plug-in hybrids


**1** READ MORE



Scooters, hoverboards and more on sale for up to 25% off today only

**1** READ MORE

CONTACT US



Universidade de Aveiro, 3810-193  
Aveiro Portugal

+351 999 999 999

standautomoveis@banana.com

Figura 7 – Feed RSS com as notícias atualizadas de www.autoblog.com

Com o objetivo de sumariar as informações de uma webpage, mostramos num simples bloco na parte inferior do nosso website o uso desse feed, bastando clicar no “READ MORE” para obter a notícia na totalidade. Optámos para disponibilizar um total de 3 notícias para não sobrecarregar muito a página.

## Conclusões

Relativamente à realização deste projeto, conseguimos realizar os objetivos pretendidos tais como: programação em Django, utilização de XML para o formato dos dados, XPath e XQuerys para filtragem, pesquisa e gestão de dados na base de dados, como muitas outras tecnologias lecionadas e aplicadas no decorrer deste trabalho.

Todos os passos realizados ao longo do projeto, desde a revisão do relatório à aprendizagem e empenho nos contextos para a realização do mesmo, contribuíram para todos estes resultados e para que os objetivos deste trabalho fossem alcançados. Assim sendo, focando-nos nos objetivos estipulados no momento inicial deste trabalho, podemos afirmar que compreendemos a importância de modelar, representar e recuperar a informação do modo distinto do relacional em sistemas de informação baseado na Web.

## Configuração para Executar a Aplicação

Para executar a aplicação precisamos de algumas dependências, como a plataforma Django, BaseXClient, biblioteca lxml, biblioteca Pillow e o django-crispy-forms.

- `pip3 install Django BaseXClient pillow django-crispy-forms`

Inicialmente tem de se iniciar o “basexserver” para iniciar o servidor com a devida base de dados, seguidamente podemos iniciar tanto no PyCharm como pelo terminal.

- `python3 manage.py runserver`
- Base de dados adicionada tem de ter o nome “anuncios”

Junto da pasta fornecemos um script para instalar todas as dependências para o sistema operativo Windows.

O ficheiro `schema.xsd` que foi usado para validar os dados encontra-se na pasta “`app/static/xml/anuncios.xsd`” do projeto django “proj1EDC” e o ficheiro `xml` que foi usado como input no BaseX encontra-se na mesma pasta mas com o nome “`anuncios.xml`”.