

ENGENHARIA DE DADOS E CONHECIMENTO

Trabalho Prático 1 - Relatório



xPand
Your Music Hub.

Trabalho realizado por:

André Rodrigues 73152
Cristiano Vagos 65169

Professor:

- Helder Zagalo

Aveiro, 26 de Outubro de 2017

Índice

Introdução	2
Dados e Suas Fontes	3
API Last.fm	3
RSS Feed - Music News	5
Esquemas de Dados	6
Dados obtidos da API Last.fm	6
Artistas	6
Álbuns	7
Tags (Palavras-chave)	8
Faixas	8
Dados obtidos do RSS Feed	8
Base de dados	9
Transformações XSL	9
Validação usando XML Schema	10
Operações sobre os dados (XQuery & XQuery Update)	10
Funcionalidades da Aplicação (UI)	11
Barra de navegação	11
Página Inicial	11
Página de Notícias	12
Página de Resultados da Pesquisa	13
Página de Artista	14
Página de Álbum	15
Página de Tags	16
Página Top Artists	17
Página Top Tracks	18
Página Top Tags	19
Footer	20
Conclusões	20
Configurações para Executar a Aplicação	20

Introdução

Neste trabalho pretendeu-se desenvolver uma aplicação Web com o objectivo de agregar todos os conceitos abordados nas aulas, utilizando a plataforma Django, bem como o uso de XML e outras ferramentas baseadas em ou envolvendo XML para o desenvolvimento da aplicação Web.

A ideia/tema do grupo foi desenvolver uma aplicação Web relacionada com Música, isto é, agregar informação sobre vários géneros musicais, nomeadamente artistas, álbuns, músicas, entre outros, e para tal, utilizámos uma API de acesso a uma fonte externa de informação no formato XML, tal como abordado nas aulas. Também utilizámos um RSS feed para fornecer notícias diárias do mundo da música.

O nome escolhido para a aplicação Web foi “xPand”, retirado da palavra inglesa “*expand*”, porque a nossa aplicação Web permite expandir conhecimento relativo a artistas, os seus álbuns e músicas, bem pelo facto de trabalharmos em torno do XML (daí o uso da letra “x”). O portal criado tem como objetivo mostrar informação importante sobre cada um dos artistas e álbuns, bem como mostrar rankings de forma a mostrar ao utilizador quais os trabalhos musicais e artistas que estão “na moda”.

Dados e Suas Fontes

Por forma a desenvolver a nossa aplicação Web, recorreremos a APIs externas de modo a obter dados seguros, confiáveis e reais sobre o nosso tema.

Resolvemos então usar a API de desenvolvedor do website [Last.fm](https://last.fm), um website conhecido e reconhecido como um grande agregador de dados acerca de Música. Para as notícias (usando um feed RSS) o grupo optou por utilizar o feed disponibilizado pelo website [Music News](https://musicnews.com).

Vamos agora descrever essas fontes de dados.

API Last.fm

Relativamente à API de desenvolvedor do [Last.fm](https://last.fm), esta é relativamente simples de utilizar e fácil de obter informação a partir da mesma, utilizando os métodos disponíveis no website da API.

API Methods

Album

- Album.addTags
- Album.getInfo
- Album.getTags
- Album.getTopTags
- Album.removeTag
- Album.search

Artist

- Artist.addTags
- Artist.getCorrection
- Artist.getInfo
- Artist.getSimilar
- Artist.getTags
- Artist.getTopAlbums
- Artist.getTopTags
- Artist.getTopTracks
- Artist.removeTag
- Artist.search

Auth

- Auth.getMobileSession
- Auth.getSession
- Auth.getToken

Chart

- Chart.getTopArtists
- Chart.getTopTags
- Chart.getTopTracks

Geo

- Geo.getTopArtists
- Geo.getTopTracks

Library

- Library.getArtists

Tag

- Tag.getInfo
- Tag.getSimilar
- Tag.getTopAlbums
- Tag.getTopArtists
- Tag.getTopTags
- Tag.getTopTracks
- Tag.getWeeklyChartList

Track

- Track.addTags
- Track.getCorrection
- Track.getInfo
- Track.getSimilar
- Track.getTags
- Track.getTopTags
- Track.love
- Track.removeTag
- Track.scrobble
- Track.search
- Track.unlove
- Track.updateNowPlaying

User

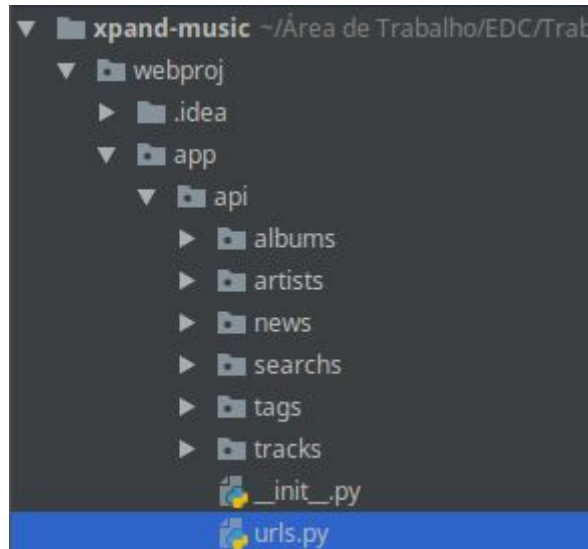
- User.getArtistTracks
- User.getFriends
- User.getInfo
- User.getLovedTracks
- User.getPersonalTags
- User.getRecentTracks
- User.getTopAlbums
- User.getTopArtists
- User.getTopTags
- User.getTopTracks
- User.getWeeklyAlbumChart
- User.getWeeklyArtistChart
- User.getWeeklyChartList
- User.getWeeklyTrackChart

Métodos disponíveis na API Last.fm

O website [Last.fm](https://last.fm) disponibiliza documentação relativa à utilização da API, dos parâmetros de utilização e dos dados devolvidos por esta de acordo com a pesquisa efetuada, a partir de um URL. É possível obter dados a partir desta API via XML e via JSON. Como o abordado nas aulas foi o uso do XML e ferramentas e tecnologias usando o XML, fizemos uso da API em XML para obter os dados e modelá-los ao nosso gosto.

Foi então necessário criar uma conta de desenvolvedor neste website para obter uma API Key, e indicar qual seria o uso da mesma (pelo que indicamos que seria para uso académico), pois sem a API Key não seria possível obter dados a partir deste website.

Por forma a facilitar o uso da API na nossa aplicação Web tendo em conta os vários parâmetros existentes para cada tipo de pedido à API, foi criado um ficheiro em Python (ver ficheiro `urls.py` na pasta `api/`) que devolve cada URL da API de acordo com os requisitos e parâmetros disponíveis para personalizar cada pesquisa tendo em conta o que pretendemos pesquisar. Este ficheiro é usado no contexto pedido na aplicação Web.



Localização do ficheiro `urls`, com os URLs da API disponíveis para uso na aplicação e seus parâmetros

A API permite-nos obter dados relativos a Artistas, Álbuns, Músicas, Tags (palavras-chave) e aos seus Charts (Tops/Rankings), que utilizamos e mostramos ao longo da navegação na nossa aplicação Web. Esta API é a fonte principal de dados da nossa aplicação.

```

<album>
  <name>Believe</name>
  <artist>Cher</artist>
  <id>2026126</id>
  <mbid>61bf0388-b8a9-48f4-81d1-7eb02706dfb0</mbid>
  <url>http://www.last.fm/music/Cher/Believe</url>
  <releasedate>6 Apr 1999, 00:00</releasedate>
  <image size="small">...</image>
  <image size="medium">...</image>
  <image size="large">...</image>
  <listeners>47602</listeners>
  <playcount>212991</playcount>
  <toptags>
    <tag>
      <name>pop</name>
      <url>http://www.last.fm/tag/pop</url>
    </tag>
    ...
  </toptags>
  <tracks>
    <track rank="1">
      <name>Believe</name>
      <duration>239</duration>
      <mbid/>
      <url>http://www.last.fm/music/Cher/_/Believe</url>
      <streamable fulltrack="0">1</streamable>
      <artist>
        <name>Cher</name>
        <mbid>bfcc6d75-a6a5-4bc6-8282-47aec8531818</mbid>
        <url>http://www.last.fm/music/Cher</url>
      </artist>
    </track>
    ...
  </tracks>
</album>

```

Exemplo de resposta da API Last.fm (caso do Álbum "Believe" da artista "Cher")

RSS Feed - Music News

Como um dos conceitos abordados nas aulas foi o uso de RSS Feed e Atom, achámos por bem incorporar um destes conceitos na nossa aplicação de forma a deixá-la mais completa. Sendo a nossa aplicação web baseada no tema Música, faz todo o sentido usarmos um RSS Feed de um website de música, pelo que escolhemos o website [Music News](#).

Optámos por este RSS Feed pelo facto da informação estar bem estruturada e abordar vários estilos musicais, facilitando a introdução da funcionalidade na nossa aplicação Web. Quanto à opção do uso do RSS em detrimento do Atom, foi uma opção baseada unicamente pelo facto do RSS ser o standard, daí termos decidido usá-lo para obter as notícias e demonstrar como podemos usar essa tecnologia no contexto da nossa aplicação.

Esquemas de Dados

Uma vez que mostrámos como acedemos aos dados, é altura de mostrarmos como lidamos com os dados.

Para o “*parse*” e interpretação dos XMLs obtidos via API e RSS no decorrer da aplicação utilizamos a biblioteca “[ElementTree](#)” do Python, bem como a biblioteca “[lxml](#)” para o uso de transformações XSL e validação com XML Schema.

A abordagem que tivemos no desenvolvimento da aplicação foi a seguinte: obter os dados pretendidos, transformá-los para a forma e estrutura final em que os pretendemos (usando transformações XSL), verificar se a transformação feita foi válida tendo em conta a estrutura que pretendemos guardar (usando XML Schemas), e caso esta tenha sido válida então guardamos os dados finais já transformados na base de dados, para que da próxima vez que estes dados sejam pretendidos não seja necessário obtê-los novamente, usando então os dados guardados na base de dados para o efeito.

Vamos agora descrever os passos necessários no desenvolvimento desta abordagem.

Dados obtidos da API Last.fm

Tal como descrito acima, criamos um ficheiro Python que devolve todos os URLs da API tendo em conta os seus parâmetros possíveis, para que depois da obtenção de dados os possamos interpretar e usar ao nosso gosto.

Foram então criadas classes em Python (localizadas em `app/model`) para obter os dados diretamente de uma forma mais fácil, usando os conceitos de programação orientada a objetos e métodos que permitam obter os dados quando pedido, no lado da view nos templates do Django (localizadas na pasta `app`). Assim, quando pretendido na aplicação basta criar um objeto e chamar os métodos que quisermos dele, minimizando a parte lógica nos templates.

Os objetos criados foram o Artist, Album, Track e Tag, no entanto o objeto Track não foi utilizado por falta de tempo.

Assim que o objeto é criado, aquando da sua criação é feita uma verificação à base de dados sobre a existência dele na base de dados. Caso exista, as variáveis do objeto são populadas com os dados vindos da base de dados, caso contrário teremos que fazer o procedimento descrito acima - obter o XML vindo da API, transformar para o formato que queremos, validar e então gravar na base de dados. Isto é feito para os objetos Artist e Album. O objeto Tag não é guardado na base de dados, pois apresenta uma abordagem diferente. Iremos explicar isso adiante.

Vejamos então em detalhe como funcionam as nossas classes:

Artistas

Todos os dados relativos a um determinado Artista são colocados na classe Artist, por exemplo a sua biografia, os melhores álbuns e músicas tendo em conta a classificação calculada pela Last.fm no seu website, e artistas relacionados com o mesmo.

Neste objeto existem também os métodos para obter os valores das variáveis criadas (por exemplo obter biografia), bem como suporte ao sistema de comentários (ver Funcionalidades da UI) e funções de verificação e introdução de dados na base de dados.

Assim, quando é criado um novo objeto Artista verificamos se ele existe na base de dados. Se existir, vamos à base de dados obter os valores para todas as variáveis do objeto, e para tal usamos XQuery para obtermos o nó Artista correspondente da base de dados, e aí fazemos uso de XPath e da biblioteca ElementTree para navegação ao longo do nó obtido, com as devidas verificações de validade dos dados.

Caso o Artista não esteja na base de dados vamos obter o XML vindo da API referente ao Artista pretendido, e então aplicamos 3 transformações XSL diferentes, uma vez que para criar um Artista precisamos de dados vindos de 3 XMLs diferentes. São aplicadas transformações XSL no XML vindo da API Last.fm "[artist.getInfo](#)" (ficheiro "transformArtist.xml" na pasta app/xml), outra transformação para o XML vindo da API Last.fm "[artist.getTopAlbums](#)" para obtermos os melhores álbuns do Artista (ficheiro "transformArtistTopAlbums.xml" na pasta app/xml), e a transformação final que é transformar o XML vindo da API Last.fm "[artist.getTopTracks](#)" para obtermos as melhores músicas do Artista (ficheiro "transformArtistTopTracks.xml" na pasta app/xml). Depois, tendo as 3 transformações concretizadas, verificamos se o XML gerado a partir delas está conforme pretendido, usando o XML Schema feito para os XMLs finais (ficheiros "schArtist.xsd", "schTopAlbums.xsd" e "schTopTracks.xsd" respetivamente). Tanto as transformações XSL como as validações de XML Schema são feitas pela biblioteca "lxml". Caso a validação seja positiva, então inserimos o nó XML Artista na base de dados usando XQuery/XUpdate e depois o XML dos TopAlbums e TopTracks no nó Artista já criado na base de dados. A nossa operação da base de dados fica assim completa.

Depois, resta-nos atribuir os valores das variáveis aos dados guardados na base de dados para posteriormente podermos utilizar o objeto onde quisermos. Para tal usamos a função "fetchInfoDatabase()". O objeto Artista fica assim com dados disponíveis para serem utilizados onde quisermos.

Álbuns

Os dados referentes aos Álbuns estão na classe Album.

Tal como descrito acima relativamente à classe Artist, a classe Album comporta-se e foi criada de forma semelhante na lógica de obtenção/introdução de dados na base de dados e no objeto propriamente dito, tendo apenas diferença nas variáveis e nos dados obtidos. Uma das diferenças principais é que cada Álbum diz respeito a um só Artista, pelo que a introdução de um Álbum na base de dados pressupõe a criação de um Artista, para que haja dados sobre ele na base de dados e possamos acrescentar um Álbum a esse Artista.

Aqui existem também os métodos para obter os valores das variáveis criadas, bem como suporte ao sistema de comentários (ver Funcionalidades da UI) e funções de verificação e introdução de dados na base de dados, tal como na classe Artist.

Nesta classe apenas é feita uma transformação XSL pois apenas é retirado um XML de um URL da API Last.fm sobre o Álbum, em "[album.getInfo](#)", que depois é transformado pelo ficheiro "transformAlbum.xml" na pasta app/xml, e a validação XML Schema pelo ficheiro "schAlbum.xsd" também disponível na mesma pasta. A lógica é a mesma, se a transformação for válida, o Álbum é então acrescentado à base de dados no nó Artista correspondente, e a função

“fetchInfoDatabase()” é chamada de forma a inserir valores nas variáveis do objeto para posterior uso na aplicação.

Tags (Palavras-chave)

Os dados relativos às Tags estão na classe Tag.

A classe Tag tem a mesma estrutura que as classes Artist e Album. A sua principal diferença é a não introdução dos dados obtidos pelo XML na base de dados, pelo que nesta classe não foi feita nenhuma transformação XSL nem validação XML Schema pois os Artistas e Álbuns em destaque poderão variar diariamente.

Então, quando pedido, a classe Tag vai sempre obter dados à API da Last.fm, em “[tag.getInfo](#)”, “[tag.getTopAlbums](#)” e “[tag.getTopTracks](#)”. Depois o procedimento é usar a biblioteca ElementTree em conjunto com XPath para obter os valores das variáveis e então atribuir valores às variáveis da classe para serem usadas na aplicação.

Faixas

Para obtermos os dados referentes às faixas de cada Álbum foi criada a classe Track, criada com o mesmo intuito das restantes: mostrar detalhes relativamente a cada faixa de cada Álbum. No entanto, o tempo restante e a aproximação do dia de apresentação / entrega não nos permitiu criar uma funcionalidade extra e fazer uso da classe no contexto da aplicação. A classe ficou completa tendo em conta a obtenção de dados da API Last.fm em “[track.getInfo](#)”, mas não é utilizada.

Dados obtidos do RSS Feed

Para que fosse possível a disponibilização de notícias relacionadas com o mundo da música e com os artistas (conforme iremos explicar em Funcionalidades da UI), como já foi mencionado inicialmente, recorreremos a utilização de um feed RSS. Não optamos por guardar nenhuma informação das notícias na base de dados uma vez que o feed RSS é atualizado diariamente, ou seja, apenas disponibiliza as notícias mais recentes e não faria sentido mostrar ao utilizador notícias desatualizadas, assim evitamos o crescimento rápido da base de dados.

A obtenção dos dados relativos a cada notícia foi possível através do RSS feed do website [Music News](#) e uma vez que toda a informação sobre a notícia não era disponibilizada no RSS feed decidimos fornecer um link para a notícia original.

Tal como nas classes descritas acima, usamos a biblioteca ElementTree em conjunto com XPath para obter os dados vindos do XML do RSS feed.

Base de dados

Após a obtenção dos dados vindos da API Last.fm, é necessário guardá-los para posterior utilização na aplicação. Como descrito, apenas fazemos uso da base de dados para o Artista e para o Álbum (e respectivos comentários - ver Funcionalidades da UI). A base de dados criada chama-se “xpand-db” e é criada no arranque do servidor da aplicação (ver ficheiros “apps.py” na pasta app, e “create.py” na pasta “app/db”), para não ter que executar um comando extra antes do arranque do servidor. No entanto se pretender apagar a base de dados, pode fazê-lo a partir da interface gráfica do programa BaseX ou então executando o ficheiro “drop.py” na pasta “app/db”.

A estrutura da base de dados é a seguinte:

```
<artists>
  <artist>
    <name></name>
    (...)
    (outros dados referentes ao artista - biografia, etc)
    <albums>
      <album>
        (dados referentes ao álbum)
      <album>
        (mais albuns... com a mesma estrutura)
    </albums>
    <topAlbums>
      (nome e imagem dos melhores albuns)
    </topAlbums>
    <topTracks>
      (nome e imagem das melhores músicas)
    </topTracks>
    <comments>
      (comentários feitos pelos utilizadores ao artista)
    </comments>
  </artist>
  (mais artistas... com a mesma estrutura que acima)
</artists>
```

O nó principal é o “artists”. Resolvemos usar este nó como o principal uma vez que a estrutura mais *top-level* seria os Artistas.

Cada Artista pode ter vários álbuns, e cada álbum está sempre ligado a um só Artista, pelo que decidimos usar esta estrutura para guardar os dados de forma a termos uma estrutura simples de entender e otimizada para usar menos recursos e evitar itens duplicados.

Transformações XSL

Para que possamos ter os dados de acordo com a estrutura definida acima para introdução na base de dados, temos que transformar os dados obtidos do XML retornado pela

API Last.fm. Conforme explicado acima, as transformações são apenas usadas nas classes Artist e Album pois são elas as únicas que irão ter conteúdo inserido na base de dados.

Os ficheiros das transformações e respetivo resultado derivado da transformação foram testados no website FreeFormatter.com, e então utilizados no contexto da aplicação conforme explicado. Esses ficheiros estão na pasta app/xml.

Validação usando XML Schema

Os ficheiros XML Schema foram criados no mesmo contexto que os das Transformações XSL, e também testados no website FreeFormatter.com antes de serem utilizados no contexto da aplicação. Esses ficheiros estão também na pasta app/xml.

Operações sobre os dados (XQuery & XQuery Update)

Como falado acima, fazemos uso de uma base de dados BaseX para guardar informação disponibilizada na nossa aplicação Web. Cada uma das queries utilizadas na aplicação Web foi testada primeiro na interface gráfica do BaseX para teste e verificação dos resultados da mesma, antes de aplicada na aplicação. Utilizamos algumas das expressões FLWOR (For, Let, Where, Order by, Return) para obtermos o pretendido, bem como a inserção, remoção e alteração de nós conforme solicitado, sendo que estas duas últimas operações são apenas aplicadas aos comentários.

Sempre que um Artista ou Álbum é solicitado na execução da aplicação web, este é obtido a partir da API, e então guardado na base de dados por forma a no próximo pedido destes podermos obter a informação já guardada, evitando recorrer novamente à API.

Quando essa solicitação é feita, é executada uma query à base de dados por forma a verificar se já existe informação na base de dados sobre o pretendido. Caso a informação não exista na base de dados é então obtida a informação à API e guardada na base de dados também utilizando uma query.

Nas páginas de artista e em cada álbum é possível efetuar comentários, bem como apagar e modificar cada comentário. Aqui também são utilizadas queries de forma a introduzir cada comentário diretamente no nó pretendido (artista ou álbum).

Funcionalidades da Aplicação (UI)

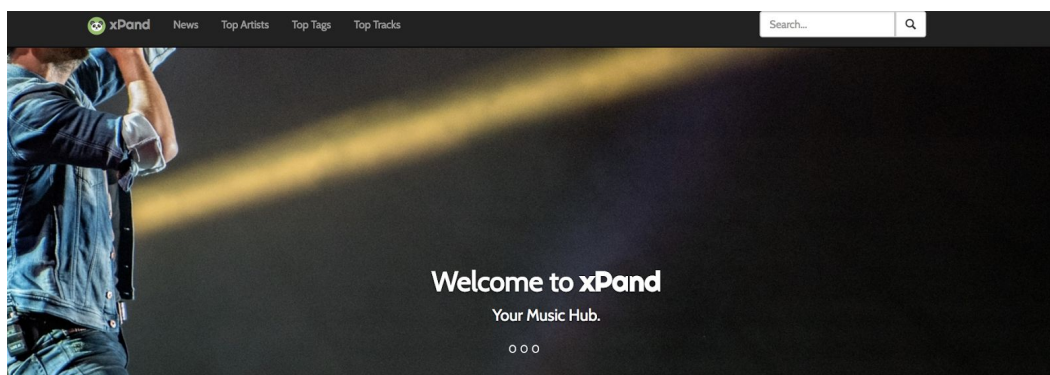
A UI da nossa aplicação é simples, intuitiva e fácil de utilizar. Decidimos usar o esqueleto fornecido pelo professor nas aulas práticas com a framework Bootstrap do Twitter, que nos oferece a possibilidade de obter uma interface limpa e agradável. Tentámos ao máximo disponibilizar uma interface capaz de dar uma experiência de navegação agradável e intuitiva ao utilizador, e acreditamos que esse objetivo foi cumprido.

Barra de navegação

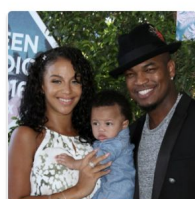


Na barra de navegação da aplicação é possível vermos o logótipo da nossa aplicação, bem como ligações para algumas páginas da aplicação. Também contém uma barra de pesquisa que permite procurar por qualquer artista ou álbum.

Página Inicial



Latest News



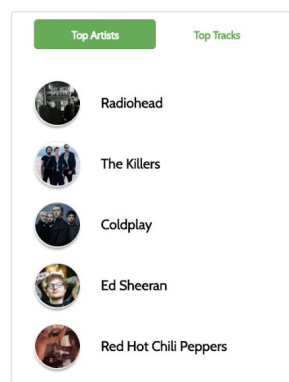
Ne-Yo to become a father of four
The baby news is a pleasant surprise for the couple.

[Read more](#)



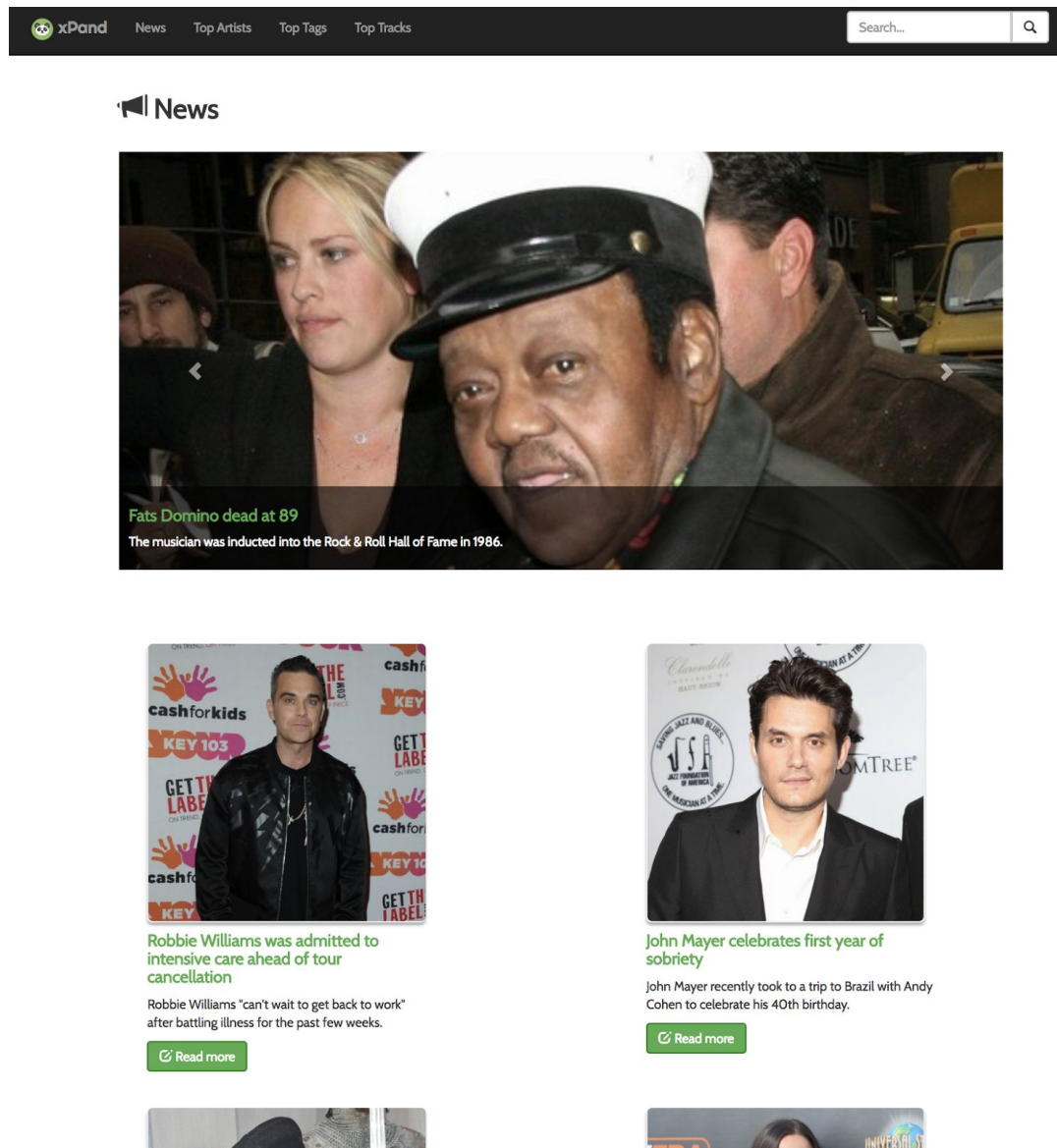
Fats Domino dead at 89
The musician was inducted into the Rock & Roll Hall of Fame in 1986.

[Read more](#)





Na página inicial é possível ver um slider com imagens referentes a música para dar uma ligação direta ao tema da aplicação web, e as últimas notícias referentes ao mundo da música obtidas via Feed RSS. Clicando nas notícias (botão, título da notícia ou imagem) irá abrir o link respetivo da notícia na fonte da notícia. Na lateral da página podemos ver um Top 5 de Artistas e Músicas, e link para os mesmos.

Página de Notícias



Na página de notícias são apresentadas as últimas notícias retiradas do RSS feed. O slider contém as últimas 6 notícias do feed, e as restantes mostradas abaixo, disponibilizando uma imagem, titulo da noticia e uma breve descrição. Clicando no botão ou no título da notícia obtém uma ligação externa para a notícia original.


Página de Resultados da Pesquisa

 [News](#) [Top Artists](#) [Top Tags](#) [Top Tracks](#) 


Hey! Good news!

Here's what we found for "fergie":


Artists & Tracks




Fergie



Fergie, Q-Tip & GoonRock




Nelly feat. Fergie




Fergie Feat. Sean Kingston


Albums




The Dutchess
by Fergie



Fergie Hit Pac - 5 Series
by Fergie




Double Dutchess
by Fergie




Music From Baz Luhrmann's Film The Great Gatsby
by Fergie

É possível a utilização de pesquisa (Search) como o utilizador desejar. Esta irá fornecer os dados encontrados que vão de encontro com o pedido do utilizador, sendo que esses dados poderão ser artistas, músicas e/ou álbuns, podendo então aceder a eles clicando no nome ou na imagem. Caso a procura seja inválida (por exemplo acedendo ao link da pesquisa sem ter preenchido a barra de pesquisa) ou não haja resultados da pesquisa, ser-lhe-á mostrada uma mensagem com o sucedido de forma a informar-lhe.






Página de Artista

 [News](#) [Top Artists](#) [Top Tags](#) [Top Tracks](#)

Artist / [Fergie](#)




Fergie


    

Fergie Duhamel (born Stacy Ann Ferguson March 27, 1975 in Hacienda Heights, California) rose to prominence when she joined the hip-hop group Black Eyed Peas in 2003. While the group took a brief hiatus, she released her first solo album, "The Dutchess", she was also featured in Slash's debut album "Slash" singing two rock songs "Beautiful Dangerous (feat. Fergie)" and "Paradise City (feat. Cypress Hill & Fergie)". The Feel Alive Songfacts states that she teamed up with Pitbull and DJ Poet for "Feel Alive".


Top Albums




The Dutchess




Fergie Hit Pac - 5 Series



Music From Baz Luhrmann's Film The Great Gatsby




L.A.LOVE (la la)




The Dutchess Deluxe (Explicit)


Top Songs




Fergalicious




Glamorous



Big Girls Don't Cry




London Bridge




Clumsy


Related




Ke\$ha




Katy Perry



Dua Lipa




Iggy Azalea



Miley Cyrus

Latest News from Fergie




Fergie: 'I wanted to stay married forever'

Fergie and Josh Duhamel went public with their separation in September (17), months after they actually split up.



Source: Newsdesk (Tue, 24 Oct 2017 15:45:00 +0100)

1 Comment



User commented

adoro!

Leave a comment...

Type your message

Comment

Na página referente a cada artista é possível ver os dados relativos ao Artista, nomeadamente a sua imagem, biografia, tags, bem como os seus melhores trabalhos (Álbuns e Músicas) e Artistas semelhantes. Caso haja notícias do RSS Feed relativas ao artista, estas são disponibilizadas abaixo. Também é possível ver e deixar um comentário ao artista referido, bem como apagar e modificar o comentário. O título dos comentários varia de acordo com o número de comentários feitos.

Página de Álbum

xPand

News

Top Artists


Top Tags

Top Tracks

Search...

Q

Fergie / Albums / The Dutchess




The Dutchess













by **Fergie**


pop albums I own Fergie female vocalists Hip-Hop

The Dutchess is the debut studio album by American musical recording artist Fergie. It was released on September 13, 2006, by A&M Records and the will.i.am Music Group as her first solo album since the break from her band The Black Eyed Peas. The album was recorded between The Black Eyed Peas' tour in 2005, and the songs were written throughout the last 8 years that preceded its release. While developing the album, Fergie wanted to create an autobiographical album that would be more intimate between her and the listener.


Tracks in The Dutchess:


 Fergalicious


 Clumsy
 All That I Got (The Make Up Song)
 London Bridge
 Pedestal
 Voodoo Doll
 Glamorous
 Here I Come
 Velvet
 Big Girls Don't Cry (Personal)
 Mary Jane Shoes
 Losing My Ground
 Finally


 Get Your Hands Up

More albums by **Fergie**:


 Fergie Hit Pac - 5 Series

 Music From Baz Luhrmann's Film The Great Gatsby

 L.A.LOVE (la la)

 The Dutchess Deluxe (Explicit)

1 Comment



User commented

o meu album favorito!

Change

Remove

Leave a comment...


Type your message

Comment

Nesta página é possível ver os dados relativos ao Álbum pretendido, como a biografia (se disponível), a capa do mesmo, bem como as músicas que o constituem. Na lateral da página pode ver também outros álbuns deste artista, sendo também possível ver esses álbuns clicando neles para aceder ao outro álbum. É também possível ver e deixar um comentário ao artista referido,

bem como apagar e modificar o comentário. O título dos comentários varia de acordo com o número de comentários feitos.

Página de Tags


 [News](#) [Top Artists](#) [Top Tags](#) [Top Tracks](#)

Tag / pop


pop

"Pop music" is a broad-term for many different types of music: The term is flexible, and the music labeled "pop" changes frequently. It does, however, usually refer to popular, mainstream music with emphasis on a catchy melody and accessible style.


Top Artists




Lady Gaga




Rihanna



Britney Spears




Madonna




Katy Perry


Top Albums




"Born This Way"
by Lady Gaga




"Teenage Dream"
by Katy Perry



"Bionic"
by Christina Aguilera




"One of the Boys"
by Katy Perry




"Dangerous Woman"
by Ariana Grande


Top Tracks




"Wonderwall"
by Oasis




"Somebody Told Me"
by The Killers



"Clocks"
by Coldplay




"Viva la Vida"
by Coldplay



"The Scientist"
by Coldplay













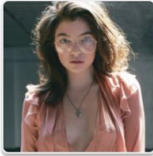












Ao carregar numa tag disponibilizada na descrição de um artista e/ou de um álbum, é possível obter uma breve descrição da tag, e os melhores artistas, álbuns e músicas que correspondem à tag seleccionada. Clicando num artista ou álbum irá direccioná-lo para a sua página respectiva.

Página Top Artists

 [News](#) [Top Artists](#) [Top Tags](#) [Top Tracks](#)


Top / Top Artists

Top Artists

				
Radiohead	The Killers	Coldplay	Ed Sheeran	Red Hot Chili Peppers
				
Kendrick Lamar	David Bowie	Taylor Swift	Imagine Dragons	Linkin Park
				
Maroon 5	The Beatles	Lorde	Queen	Foo Fighters
				
Arctic Monkeys	Kanye West	Calvin Harris	Nirvana	Lana Del Rey
				





















A página Top Artists mostra a classificação dos artistas de acordo com o website Last.fm. No fundo desta página é possível ir para a página seguinte para ver os próximos artistas de acordo com o ranking. Clicando num artista leva-o para a sua página respectiva.

Página Top Tracks

 [News](#) [Top Artists](#) [Top Tags](#) [Top Tracks](#)


Top / Top Tracks

Top Tracks

 <p>"New Rules" by Dua Lipa</p>	 <p>"Look What You Made Me Do" by Taylor Swift</p>	 <p>"Rockstar" by Post Malone</p>	 <p>"Feel It Still" by Portugal. The Man</p>	 <p>"HUMBLE." by Kendrick Lamar</p>
 <p>"Havana" by Camila Cabello</p>	 <p>"Sorry Not Sorry" by Demi Lovato</p>	 <p>"Too Good at Goodbyes" by Sam Smith</p>	 <p>"Shape of You" by Ed Sheeran</p>	 <p>"Thunder" by Imagine Dragons</p>
 <p>"Mr. Brightside" by The Killers</p>	 <p>"What About Us" by P!nk</p>	 <p>"Mi Gente" by J Balvin</p>	 <p>"Dusk Till Dawn - Radio Edit" by Zayn</p>	 <p>"What Lovers Do (feat. SZA)" by Maroon 5</p>
 <p>"Bodak Yellow" by Cardi B</p>	 <p>"1-800-273-8255" by Logic</p>	 <p>"Feels" by Calvin Harris</p>	 <p>"Smells Like Teen Spirit" by Marshmello</p>	 <p>"Silence" by Marshmello</p>

A página Top Tracks mostra a classificação das músicas, mostrando também os respetivos artistas de acordo com o website Last.fm. No fundo desta página é possível ir para a página seguinte para ver as próximas músicas de acordo com o ranking. Clicando num artista leva-o para a sua página respetiva.


Página Top Tags


 [News](#) [Top Artists](#) [Top Tags](#) [Top Tracks](#)


Top / Top Tags


Top Tags

rock



Coldplay



Linkin Park



Red Hot Chili Peppers



David Bowie

electronic



Daft Punk



Depeche Mode



Crystal Castles



Björk

seen live


The National


Editors

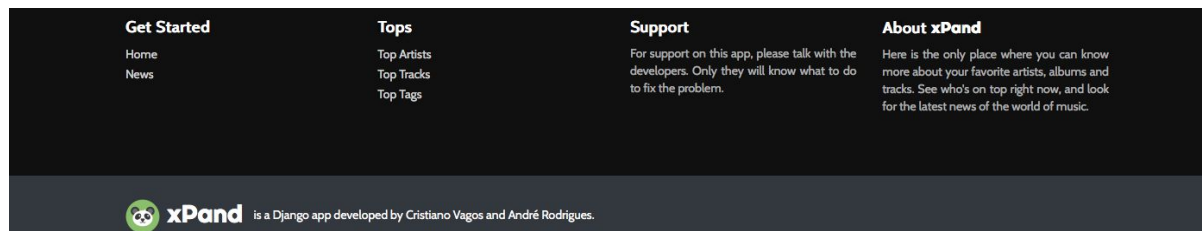

Biffy Clyro


Major Lazer

alternative

A página Top Tags tem como objectivo mostrar as melhores tags (palavras-chave) e os melhores artistas de cada tag. Clicando numa Tag leva-o à página respetiva de cada tag, clicando num artista leva-o à sua página respetiva.

Footer



O footer, situado no final de cada página fornece o acesso à página principal, à página de notícias e aos Tops (Artistas, Tracks e Tags), ainda disponibiliza suporte da aplicação e uma breve descrição sobre a aplicação (About).

Conclusões

Acreditamos que este projeto foi de encontro aos objetivos do mesmo: o uso da framework Django, XML e de ferramentas utilizando esta forma de representação de dados e a obtenção, modelação e validação dos mesmos, conforme abordado e lecionado nas aulas desta unidade curricular. Temos a certeza de que fizemos um bom trabalho e que cumprimos com o objetivo.

No entanto, pelo facto de termos sido duas pessoas a fazer o projeto é possível verificar que em algumas partes do projeto este esteja incompleto, como por exemplo lidar com erros vindos da API Last.fm (na documentação é possível saber quais os códigos de erro), criar páginas de erro e adicionar outras funcionalidades (por exemplo a possibilidade de haver utilizadores, sistemas de rating/pontuação, obter o vídeo de YouTube da música via API do YouTube, entre outros).

Muitas ideias estavam pensadas e a estrutura do código preparada para a adição de mais funcionalidades de forma a completar a aplicação, no entanto tendo em conta o tempo disponível para o desenvolvimento, o projeto que hoje apresentamos foi o possível, e acabamos este projeto com o sentimento de dever cumprido pois aprendemos bastante com ele, tendo compreendido os conceitos lecionados nesta unidade curricular e o seu uso num contexto real, pois esta aplicação poderia ser facilmente alojada e estar disponível online.

Configurações para Executar a Aplicação

Para executar a aplicação, tem de ter a versão 3.6 do Python, e ter instalado o Django, bem como o BaseX, e a biblioteca lxml ([link](#)).

Depois tem de executar no terminal ou linha de comandos o comando “basexserver” para iniciar o servidor BaseX para comunicação com a base de dados. Depois é executar a aplicação Django ou a partir do PyCharm ou a partir da linha de comandos com o comando habitual