



Instituto Politécnico
de Viana do Castelo

Licenciatura em ENGENHARIA INFORMÁTICA / Degree INFORMATICS ENGINEERING

INTEGRAÇÃO DE SISTEMAS

Trabalho Prático 2

Escrita de APIs, importação de dados e integração entre bases de dados e outros serviços

N.º 20749– Basílio Barbosa;

N.º 22340 – Tiago Soares

Orientador(es):

- Professor Doutor Jorge Ribeiro, Professor Doutor Luís Teófilo

■ Índice

1. Introdução e Objectivos
2. Tecnologias e ferramentas utilizadas
3. Fundamentos teóricos
4. Criação do Schema
5. Base de Dados
6. Conversão do ficheiro CSV em XML
7. Visualização de dados geográficos
8. Frontend
9. Bibliografia e Referências Web

■ 1. Introdução e Objetivos



No âmbito da unidade curricular de **Integração de Sistemas** foi nos proposta uma implementação que consiste no uso da **API Nominatim**, **Docker**, **pgAdmin** e **XML** usando os módulos de arquitetura que foram disponibilizados no enunciado do trabalho. A **API Nominatim** é usada para obter os dados das **coordenadas** relativas às **cidades** que se encontram no **ficheiro XML**. Em suma, este trabalho tem como objetivo o aprofundamento dos conhecimentos obtidos em sala de aula relativos à utilização de linguagens de anotação, APIs e base de dados GIS com integração de sistemas.

■ Dataset (CSV)

https://www.kaggle.com/datasets/heesoo37/120-years-of-olympic-history-athletes-and-results?select=athlete_events.csv

Guardar Automaticamente athlete_events.csv

Ficheiro Base Inserir Esquema da Página Fórmulas Dados Rever Ver Ajuda

Calibri 11 A A

Cortar Copiar Colar Área de Transferência Pincel de Formatação

Formato de Texto

Normal

Formatação Condicional Formatar como Tabela Neutro

POSSÍVEL PERDA DE DADOS Poderá perder algumas funcionalidades se guardar este livro no formato delimitado por vírgulas (.csv). Para preservar estas funcionalidades, guarde o livro num formato de ficheiro .xlsx

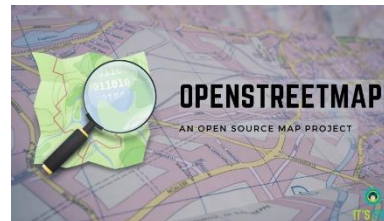
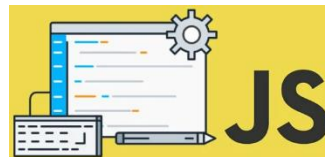
A1 ID,"Name","Sex","Age","Height","Weight","Team","NOC","Games","Year","Season","City","Sport","Event","Medal"

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	ID,"Name","Sex","Age","Height","Weight","Team","NOC","Games","Year","Season","City","Sport","Event","Medal"																	
2	1,"A Dijkstra","M",24,180,80,"China","CHN",1992 Summer,1992,"Summer","Barcelona","Basketball","Basketball Men's Basketball",NA																	
3	2,"A Lamusi","M",23,170,60,"China","CHN",2012 Summer,2012,"Summer","London","Judo","Judo Men's Extra-Lightweight",NA																	
4	3,"Gunnar Nielsen Aabye","M",34,NA,NA,"Denmark","DEN",1920 Summer,1920,"Summer","Antwerpen","Football","Football Men's Football",NA																	
5	4,"Edgar Lindén Aabye","M",21,185,82,"Netherlands","NED",1988 Winter,1988,"Winter","Calgary","Tug-Of-War","Tug-Of-War Men's Tug-Of-War",Gold																	
6	5,"Christine Jacobs Aafink","F",21,185,82,"Netherlands","NED",1988 Winter,1988,"Winter","Calgary","Speed Skating","Speed Skating Women's 500 metres",NA																	
7	6,"Christine Jacobs Aafink","F",21,185,82,"Netherlands","NED",1988 Winter,1988,"Winter","Calgary","Speed Skating","Speed Skating Women's 1,000 metres",NA																	
8	7,"Christine Jacobs Aafink","F",25,185,82,"Netherlands","NED",1992 Winter,1992,"Winter","Albertville","Speed Skating","Speed Skating Women's 500 metres",NA																	
9	8,"Christine Jacobs Aafink","F",25,185,82,"Netherlands","NED",1992 Winter,1992,"Winter","Albertville","Speed Skating","Speed Skating Women's 1,000 metres",NA																	
10	9,"Christine Jacobs Aafink","F",27,185,82,"Netherlands","NED",1994 Winter,1994,"Winter","Lillehammer","Speed Skating","Speed Skating Women's 500 metres",NA																	
11	10,"Christine Jacobs Aafink","F",27,185,82,"Netherlands","NED",1994 Winter,1994,"Winter","Lillehammer","Speed Skating","Speed Skating Women's 1,000 metres",NA																	
12	6,"Per Knut Aaland","M",31,188,75,"United States","USA",1992 Winter,1992,"Winter","Albertville","Cross Country Skiing","Cross Country Skiing Men's 10 kilometres",NA																	
13	6,"Per Knut Aaland","M",31,188,75,"United States","USA",1992 Winter,1992,"Winter","Albertville","Cross Country Skiing","Cross Country Skiing Men's 50 kilometres",NA																	
14	6,"Per Knut Aaland","M",31,188,75,"United States","USA",1992 Winter,1992,"Winter","Albertville","Cross Country Skiing","Cross Country Skiing Men's 10/15 kilometres Pursuit",NA																	
15	6,"Per Knut Aaland","M",31,188,75,"United States","USA",1992 Winter,1992,"Winter","Albertville","Cross Country Skiing","Cross Country Skiing Men's 4 x 10 kilometres Relay",NA																	
16	6,"Per Knut Aaland","M",33,188,75,"United States","USA",1994 Winter,1994,"Winter","Lillehammer","Cross Country Skiing","Cross Country Skiing Men's 10 kilometres",NA																	
17	6,"Per Knut Aaland","M",33,188,75,"United States","USA",1994 Winter,1994,"Winter","Lillehammer","Cross Country Skiing","Cross Country Skiing Men's 30 kilometres",NA																	
18	6,"Per Knut Aaland","M",33,188,75,"United States","USA",1994 Winter,1994,"Winter","Lillehammer","Cross Country Skiing","Cross Country Skiing Men's 10/15 kilometres Pursuit",NA																	
19	6,"Per Knut Aaland","M",33,188,75,"United States","USA",1994 Winter,1994,"Winter","Lillehammer","Cross Country Skiing","Cross Country Skiing Men's 4 x 10 kilometres Relay",NA																	
20	7,"John Aalberg","M",31,183,72,"United States","USA",1992 Winter,1992,"Winter","Albertville","Cross Country Skiing","Cross Country Skiing Men's 10 kilometres",NA																	
21	7,"John Aalberg","M",31,183,72,"United States","USA",1992 Winter,1992,"Winter","Albertville","Cross Country Skiing","Cross Country Skiing Men's 50 kilometres",NA																	
22	7,"John Aalberg","M",31,183,72,"United States","USA",1992 Winter,1992,"Winter","Albertville","Cross Country Skiing","Cross Country Skiing Men's 10/15 kilometres Pursuit",NA																	
23	7,"John Aalberg","M",31,183,72,"United States","USA",1992 Winter,1992,"Winter","Albertville","Cross Country Skiing","Cross Country Skiing Men's 4 x 10 kilometres Relay",NA																	
24	7,"John Aalberg","M",33,183,72,"United States","USA",1994 Winter,1994,"Winter","Lillehammer","Cross Country Skiing","Cross Country Skiing Men's 10 kilometres",NA																	
25	7,"John Aalberg","M",33,183,72,"United States","USA",1994 Winter,1994,"Winter","Lillehammer","Cross Country Skiing","Cross Country Skiing Men's 30 kilometres",NA																	
26	7,"John Aalberg","M",33,183,72,"United States","USA",1994 Winter,1994,"Winter","Lillehammer","Cross Country Skiing","Cross Country Skiing Men's 10/15 kilometres Pursuit",NA																	
27	7,"John Aalberg","M",33,183,72,"United States","USA",1994 Winter,1994,"Winter","Lillehammer","Cross Country Skiing","Cross Country Skiing Men's 4 x 10 kilometres Relay",NA																	
28	8,"Cornelia Cor Aalten (-Strannood)","F",18,168,NA,"Netherlands","NED",1932 Summer,1932,"Summer","Los Angeles","Athletics","Athletics Women's 100 metres",NA																	
29	8,"Cornelia Cor Aalten (-Strannood)","F",18,168,NA,"Netherlands","NED",1932 Summer,1932,"Summer","Los Angeles","Athletics","Athletics Women's 4 x 100 metres Relay",NA																	
30	9,"Antti Sami Aalto","M",26,186,96,"Finland","FIN",2002 Winter,2002,"Winter","Salt Lake City","Ice Hockey","Ice Hockey Men's Ice Hockey",NA																	
31	10,"Einar Ferdinand Einar Aalto","M",26,NA,NA,"Finland","FIN",1952 Summer,1952,"Summer","Helsinki","Swimming","Swimming Men's 400 metres Freestyle",NA																	
32	11,"Jorma Ilmari Aalto","M",22,182,76,5,"Finland","FIN",1980 Winter,1980,"Winter","Lake Placid","Cross Country Skiing","Cross Country Skiing Men's 30 kilometres",NA																	
33	12,"Jyrki Tapani Aalto","M",31,172,70,"Finland","FIN",2000 Summer,2000,"Summer","Sydney","Badminton","Badminton Men's Singles",NA																	
34	13,"Minna Maarit Aalto","F",34,159,55,5,"Finland","FIN",1996 Summer,1996,"Summer","Atlanta","Sailing","Sailing Women's Windsurfer",NA																	
35	13,"Minna Maarit Aalto","F",34,159,55,5,"Finland","FIN",2000 Summer,2000,"Summer","Sydney","Sailing","Sailing Women's Windsurfer",NA																	
36	14,"Pirjo Hannele Aalto (Mattila)","F",32,171,65,"Finland","FIN",1994 Winter,1994,"Winter","Lillehammer","Biathlon","Biathlon Women's 7.5 kilometres Sprint",NA																	
37	15,"Arvo Ossiian Aaltonen","M",22,NA,NA,"Finland","FIN",1912 Summer,1912,"Summer","Stockholm","Swimming","Swimming Men's 200 metres Breaststroke",NA																	
38	15,"Arvo Ossiian Aaltonen","M",22,NA,NA,"Finland","FIN",1912 Summer,1912,"Summer","Stockholm","Swimming","Swimming Men's 400 metres Breaststroke",NA																	
39	15,"Arvo Ossiian Aaltonen","M",20,NA,NA,"Finland","FIN",1920 Summer,1920,"Summer","Antwerpen","Swimming","Swimming Men's 200 metres Breaststroke",Bronze																	

athlete_events

Pronto Acessibilidade: Indisponível

■ 2. Tecnologias e ferramentas utilizadas

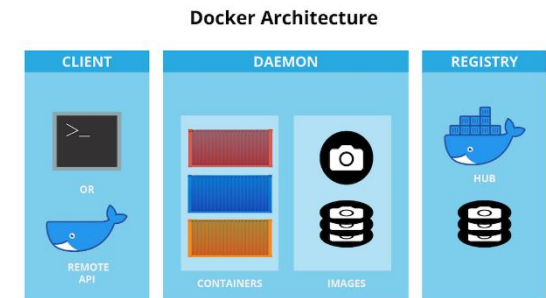
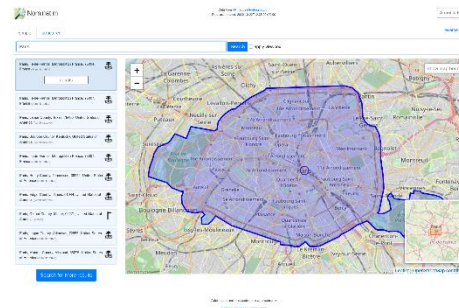
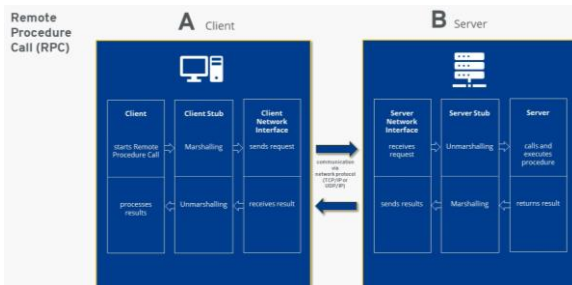


3. Fundamentos teóricos

Um **RPC** é uma tecnologia de **comunicação** entre **processos** que permite a um **programa** solicitar um **procedimento** de outro **computador** ligados em **rede**(envolve um **cliente** e um **servidor**).

O **XML** é uma **linguagem de marcação** para desenvolver documentos organizados **hierarquicamente**. Uma **API** é um conjunto de padrões de uma **interface** que permite que duas **aplicações conversem** entre si. Ou seja, é o “**mensageiro**” que regista o **pedido** e **informa** o **sistema** que que fazer.

O **Docker** **virtualiza** um **sistema operacional** para entregar software em **pacotes** denominados “**containers**”.



4. Criação do Schema

Scripts da criação dos schemas.

EXPLORER

main.py ...migrator main.py ...entities 1 initial_schema.sql ...

TP2-IS-20749-22340

- images
 - db
 - rel
 - Dockerfile
 - initial_schema.sql
 - xml
 - Dockerfile
 - initial_schema.sql
 - python
 - web
 - IS-TP2-testes
 - src
 - api
 - entities

pg-xml

```

1 CREATE TABLE public.imported_documents (
2   id          serial PRIMARY KEY,
3   file_name   VARCHAR(250) UNIQUE NOT NULL,
4   xml         XML NOT NULL,
5   created_on  TIMESTAMP NOT NULL DEFAULT NOW(),
6   updated_on  TIMESTAMP NOT NULL DEFAULT NOW()
7 );
8
9 CREATE TABLE public.converted_documents (
10  id          serial PRIMARY KEY,
11  src         VARCHAR(250) UNIQUE NOT NULL,
12  file_size   BIGINT NOT NULL,
13  dst         VARCHAR(250) UNIQUE NOT NULL,
14  created_on  TIMESTAMP NOT NULL DEFAULT NOW(),
15  updated_on  TIMESTAMP NOT NULL DEFAULT NOW()
16 );
17

```

EXPLORER

main.py ...migrator main.py ...entities 1 initial_schema.sql ...

TP2-IS-20749-22340

- images
 - db
 - rel
 - Dockerfile
 - initial_schema.sql
 - xml
 - Dockerfile
 - initial_schema.sql
 - python
 - web
 - IS-TP2-testes
 - src
 - api
 - entities
 - _pycache_
 - _init_.py
 - atlethe.py
 - season.py
 - team.py
 - main.py
 - run.sh
 - gis
 - graphql
 - proc
 - daemon
 - gis-updater
 - importer
 - migrator
 - main.py
 - run.sh
 - test.py
 - frontend

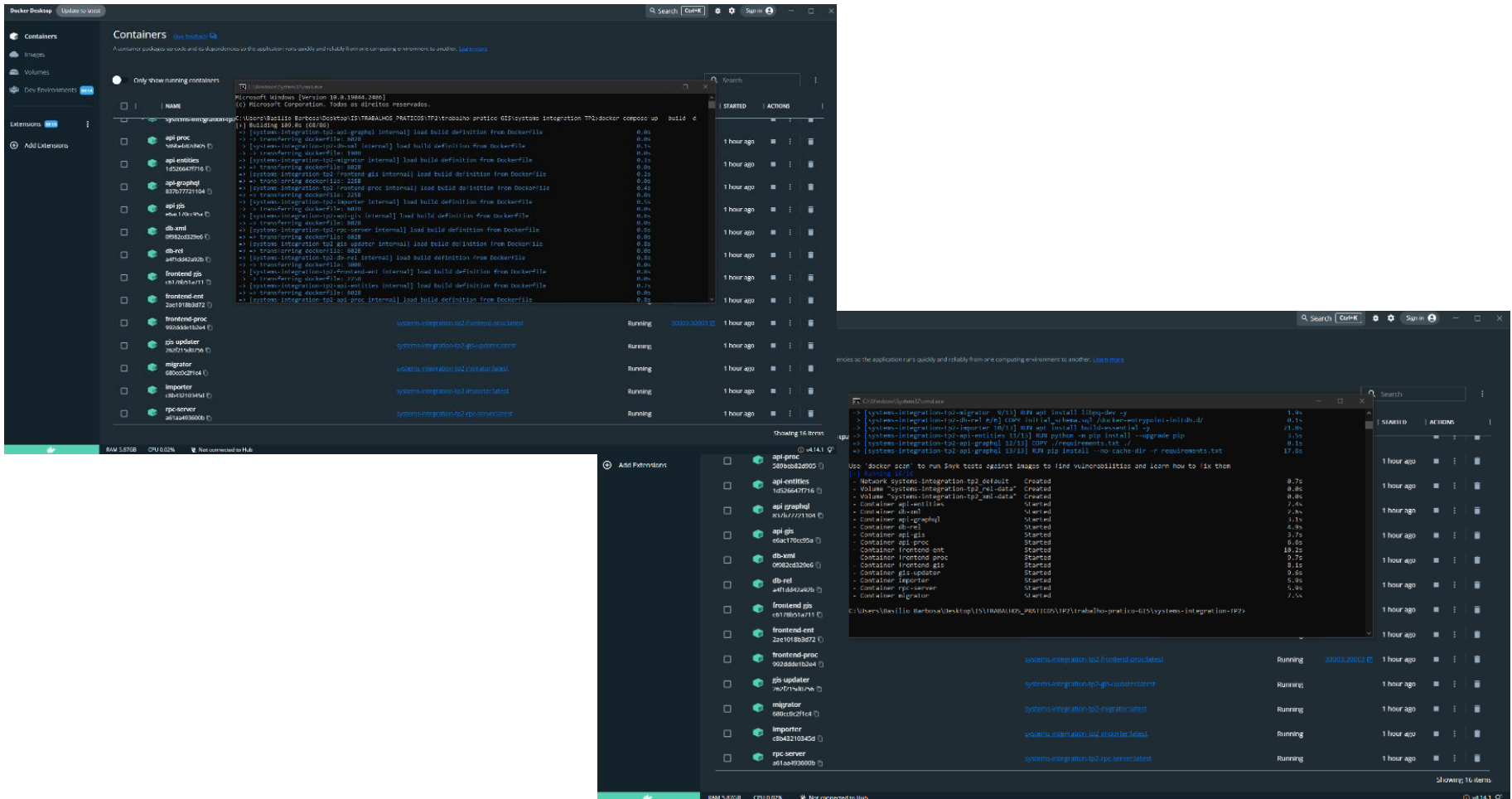
pg-rel

```

41
42
43 CREATE TABLE public.season (
44   id          uuid PRIMARY KEY DEFAULT uuid_generate_v4(),
45   season      VARCHAR(250) NOT NULL,
46   created_on  TIMESTAMP NOT NULL DEFAULT NOW(),
47   updated_on  TIMESTAMP NOT NULL DEFAULT NOW()
48 );
49
50 CREATE TABLE public.atlethe (
51   id          uuid PRIMARY KEY DEFAULT uuid_generate_v4(),
52   name        VARCHAR(250) NOT NULL,
53   sex         VARCHAR(250) NOT NULL,
54   age         VARCHAR(250) NOT NULL,
55   height      VARCHAR(250) NOT NULL,
56   weight      VARCHAR(250) NOT NULL,
57   team        VARCHAR(250) NOT NULL,
58   noc         VARCHAR(250) NOT NULL,
59   games       VARCHAR(250) NOT NULL,
60   year        VARCHAR(250) NOT NULL,
61   season      VARCHAR(250) NOT NULL,
62   city        VARCHAR(250) NOT NULL,
63   lat         VARCHAR(250) NOT NULL,
64   lon         VARCHAR(250) NOT NULL,
65   sport       VARCHAR(250) NOT NULL,
66   event       VARCHAR(250) NOT NULL,
67   medal       VARCHAR(250) NOT NULL,
68   geom        GEOMETRY,
69   season_id   uuid,
70   created_on  TIMESTAMP NOT NULL DEFAULT NOW(),
71   updated_on  TIMESTAMP NOT NULL DEFAULT NOW()
72 );
73
74 ALTER TABLE atlethe
75   ADD CONSTRAINT atlethe_season_id_fk
76   FOREIGN KEY (season_id) REFERENCES season
77   ON DELETE SET NULL;
78

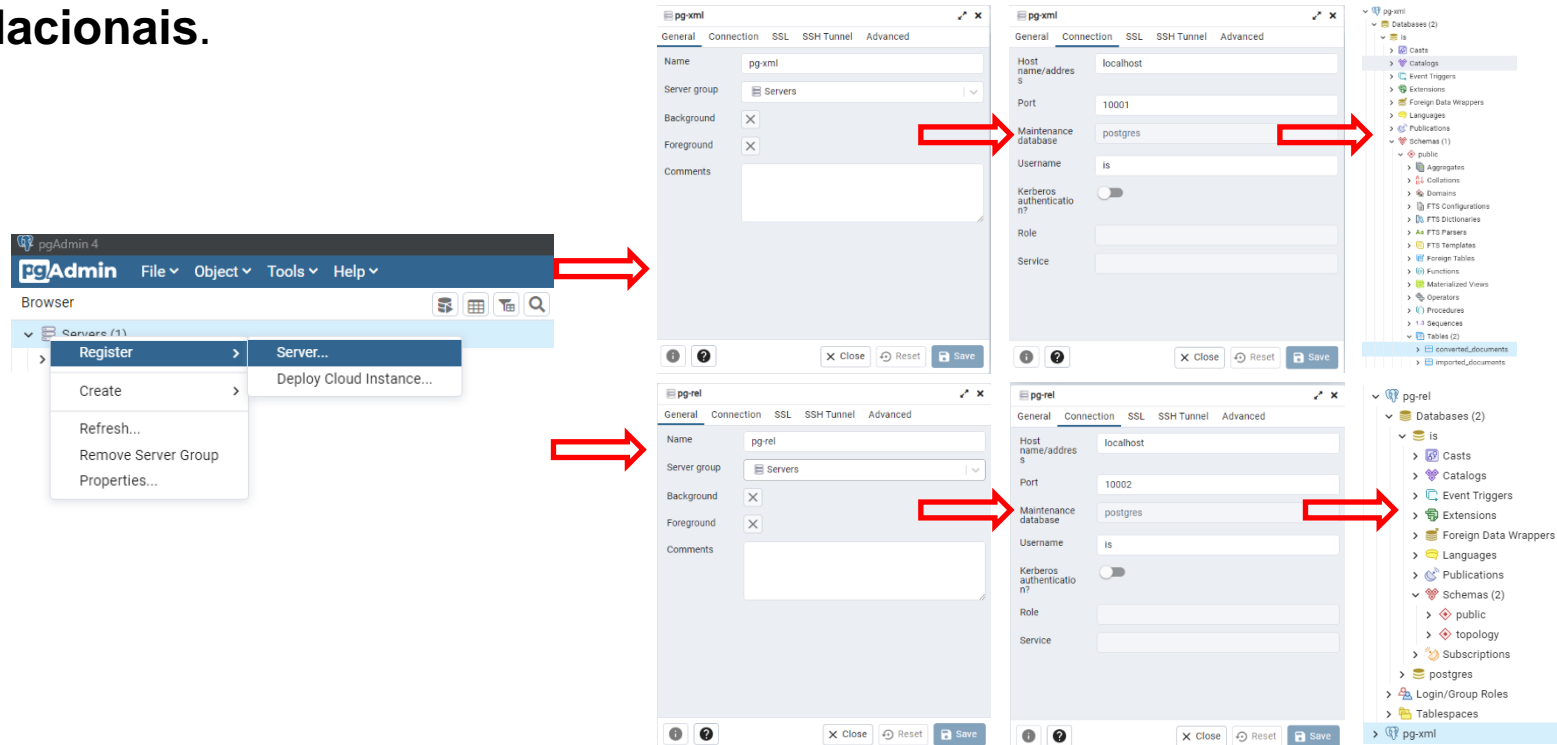
```

Ao dar **build** no projeto os **containers** ficam a correr e são **criadas** as **tabelas**.



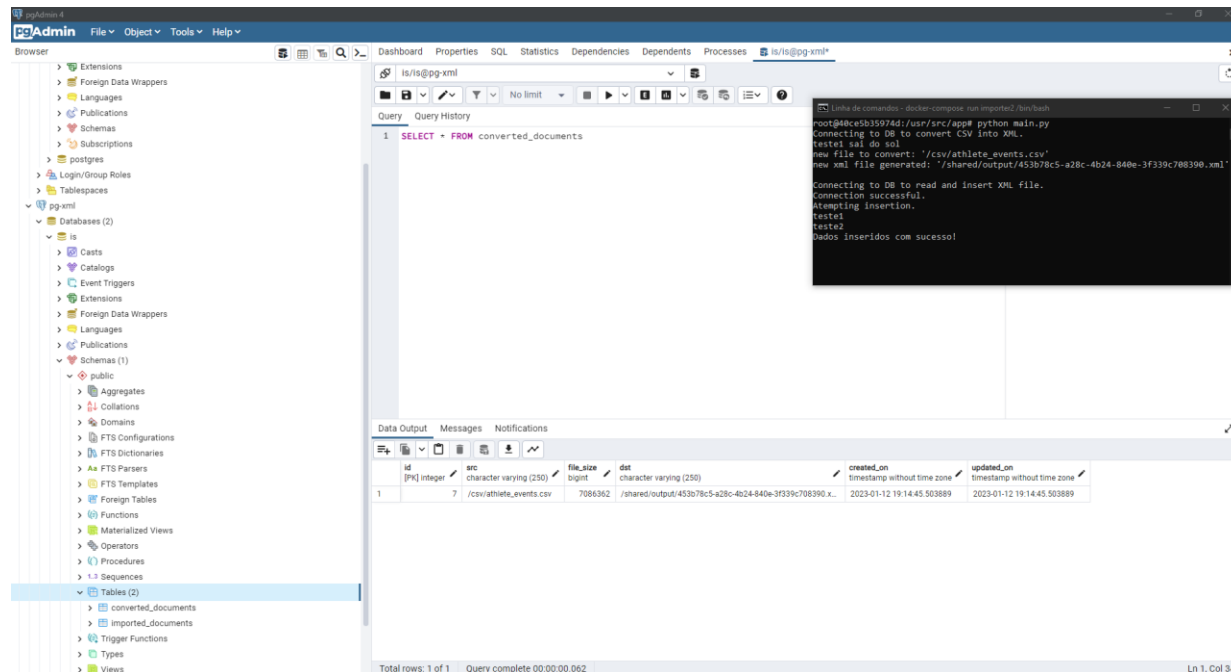
4. Criação do Schema

Foram criados 2 **schemas**, o **pg-xml** servirá para **armazenar os ficheiros CSV e XML**(usando uma **conversão de ficheiros**) em **tabelas** e o **pg-rel** servirá para **armazenar a informação** contida no **XML** em **tabelas relacionais**.



5. Base de dados

Criação da tabela **converted_documents** com os atributos **id** incrementado automaticamente, **src** que **guarda o diretório do ficheiro CSV**, **file_size** que nos apresenta o **tamanho do ficheiro**, **dst** que nos dá o **diretório do ficheiro XML** resultante da conversão, a **data de criação(created_on)** e a **data de atualização(updated_on)**. Executou-se ainda um **SELECT** para verificarmos o que temos na tabela.



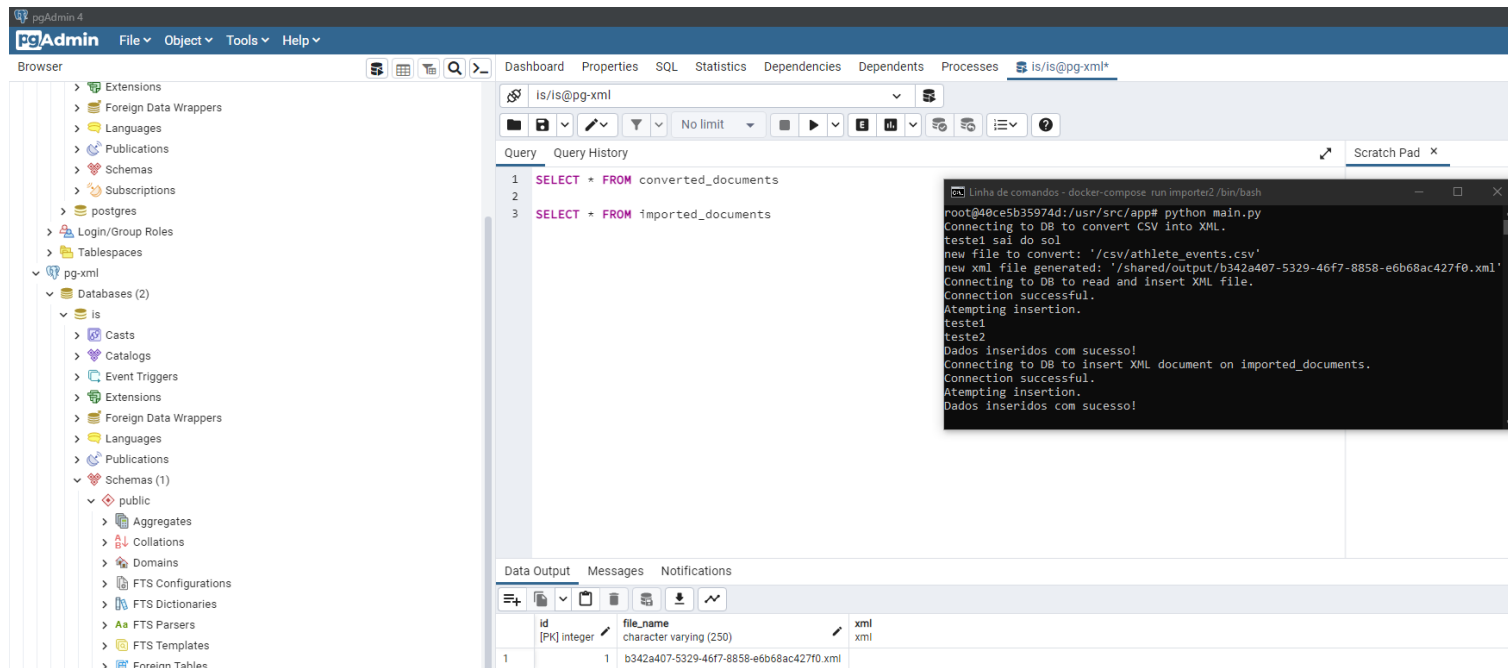
The screenshot shows the pgAdmin 4 interface. On the left, the 'pg.xml' database is selected, showing a list of tables including 'converted_documents'. The main pane displays a SQL query: `SELECT * FROM converted_documents`. The results pane shows a single row of data:

id	src	file_size	dst	created_on	updated_on
1	/usr/src/app/athlete_events.csv	7086362	/shared/output/453b78c5-a28c-4b24-840e-3f339c708390.xml	2023-01-12 19:14:45.503889	2023-01-12 19:14:45.503889

Below the table, it indicates 'Total rows: 1 of 1' and 'Query complete 00:00:00.062'.

5. Base de dados

Criação da tabela **imported_documents** com os atributos **id** que é gerado, **file_name** que guarda o **diretório** do ficheiro **XML** e **xml** que armazena os **dados do ficheiro XML**. Executou-se ainda um **SELECT** para verificarmos o que temos na tabela.



The screenshot shows the pgAdmin 4 interface. On the left, the 'pg-xml' database is expanded, showing the 'public' schema. The 'imported_documents' table is visible in the 'Schemas' list. The main pane shows the 'Query' editor with the following SQL queries:

```
1 SELECT * FROM converted_documents
2
3 SELECT * FROM imported_documents
```

Below the queries, the 'Data Output' tab shows the results of the last query (SELECT * FROM imported_documents). The table has three columns: 'id' (integer), 'file_name' (character varying (250)), and 'xml' (xml). The results show one row with the following values:

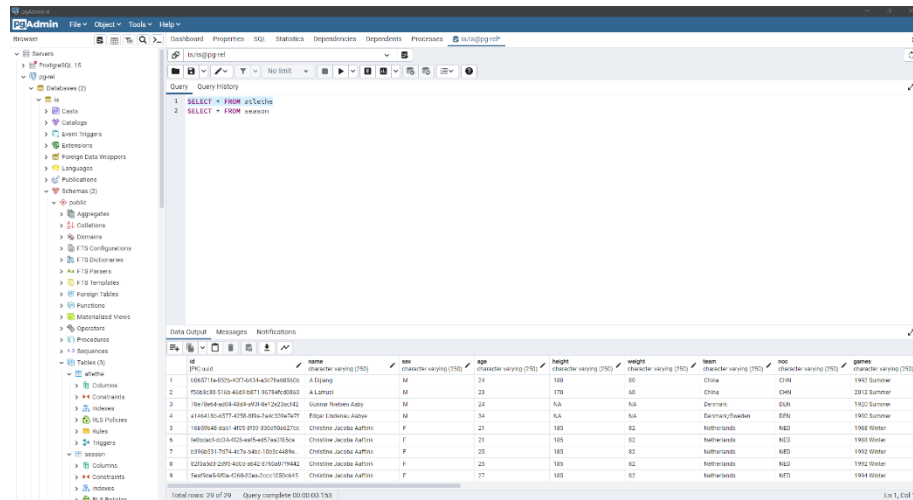
id	file_name	xml
1	b342a407-5329-46f7-8858-e6b68ac427f0.xml	

On the right, a terminal window shows the execution of a script. The script connects to the database, reads an XML file, and inserts it into the 'imported_documents' table. The output shows successful connections and insertions.

```
Linha de comandos - docker-compose run importer2/bin/bash
root@40ce5b35974d:/usr/src/app# python main.py
Connecting to DB to convert CSV into XML.
teste1 sai do sol
new file to convert: '/csv/athlete_events.csv'
new xml file generated: '/shared/output/b342a407-5329-46f7-8858-e6b68ac427f0.xml'
Connecting to DB to read and insert XML file.
Connection successful.
Attempting insertion.
teste1
teste2
Dados inseridos com sucesso!
Connecting to DB to insert XML document on imported_documents.
Connection successful.
Attempting insertion.
Dados inseridos com sucesso!
```

5. Base de dados

Executou-se **SELECT** para verificarmos o que foi armazenado nas tabelas relacionais como se pode averiguar em seguida.

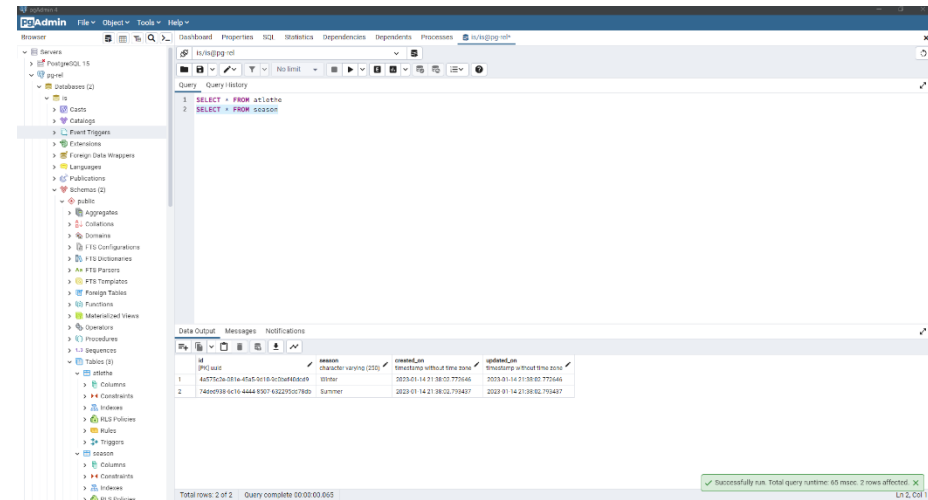


pgAdmin 4 interface showing a SQL query and its results. The query is:

```
1 SELECT * FROM seasons
2 SELECT * FROM seasons
```

The results are displayed in a table with the following columns: id, name, sex, age, height, weight, team, noc, games. The table contains 9 rows of data.

id	name	sex	age	height	weight	team	noc	games
1	1980 Winter	M	24	188	60	China	CHN	1980 Summer
2	1984 Winter	M	24	179	60	China	CHN	2012 Summer
3	1988 Winter	M	24	N/A	N/A	Denmark	DEN	1980 Summer
4	1992 Winter	M	24	N/A	N/A	Denmark/Canada	DEN	1980 Summer
5	1996 Winter	F	21	165	62	Netherlands	NED	1980 Winter
6	2000 Winter	F	21	165	62	Netherlands	NED	1980 Winter
7	2004 Winter	F	25	165	62	Netherlands	NED	1980 Winter
8	2008 Winter	F	25	165	62	Netherlands	NED	1980 Winter
9	2012 Winter	F	27	165	62	Netherlands	NED	1980 Winter



pgAdmin 4 interface showing a SQL query and its results. The query is:

```
1 SELECT * FROM seasons
2 SELECT * FROM seasons
```

The results are displayed in a table with the following columns: id, name, sex, age, height, weight, team, noc, games. The table contains 9 rows of data.

id	name	sex	age	height	weight	team	noc	games
1	1980 Winter	M	24	188	60	China	CHN	1980 Summer
2	1984 Winter	M	24	179	60	China	CHN	2012 Summer
3	1988 Winter	M	24	N/A	N/A	Denmark	DEN	1980 Summer
4	1992 Winter	M	24	N/A	N/A	Denmark/Canada	DEN	1980 Summer
5	1996 Winter	F	21	165	62	Netherlands	NED	1980 Winter
6	2000 Winter	F	21	165	62	Netherlands	NED	1980 Winter
7	2004 Winter	F	25	165	62	Netherlands	NED	1980 Winter
8	2008 Winter	F	25	165	62	Netherlands	NED	1980 Winter
9	2012 Winter	F	27	165	62	Netherlands	NED	1980 Winter

6. Conversão do ficheiro CSV em XML

Realizou-se a **conversão** do ficheiro **CSV** em **XML**, inserindo a informação das coordenadas através da **API**.

```
C:\Windows\System32\cmd.exe - docker-compose run importer2 /bin/bash
C:\Users\Basilio Barbosa\Desktop\IS\TRABALHOS_PRATICOS\TP2\tp2-IS-20749-22340>docker-compose start
[*] Running 12/15
- Container api-graphql2 Started
- Container db-xm12 Started
- Container api-proc2 Started
- Container api-entitles2 Started
- Container api-gis2 Started
- Container db-rel2 Started
- Container rpc-server2 Started
- Container importer2 Started
- Container migrator2 Started
- Container frontend-proc2 Started
- Container frontend-ent2 Started
- Container frontend-gis2 Started
- Container gis-updater2 Started

[*] Running 1/0o Barbosa\Desktop\IS\TRABALHOS_PRATICOS\TP2\tp2-IS-20749-22340>docker-compose run importer2 /bin/bash
- Container db-xm12 Running
root@885644b3510:/usr/src/app# python main.py
Traceback (most recent call last):
  File "/usr/src/app/main.py", line 11, in <module>
    from utils.convertXML import convertXML
  File "/usr/src/app/utils/convertXML.py", line 2, in <module>
    import requests
ModuleNotFoundError: No module named 'requests'
root@885644b3510:/usr/src/app# pip install requests
Collecting requests
  Downloading requests-2.28.2-py3-none-any.whl (62 kB)
    02.8/62.8 kB 3.6 MB/s eta 0:00:00
Collecting charset-normalizer<4,>=2
  Downloading charset-normalizer-3.0.1-cp39-cp39-manylinux_2_17_x86_64_manylinux2014_x86_64.whl (198 kB)
    198.0/198.0 kB 0.1 MB/s eta 0:00:00
Collecting urllib3<1.27,>=1.21.1
  Downloading urllib3-1.26.14-py2.py3-none-any.whl (140 kB)
    140.0/140.0 kB 8.4 MB/s eta 0:00:00
Collecting certifi>=2017.4.17
  Downloading certifi-2022.12.7-py3-none-any.whl (155 kB)
    155.3/155.3 kB 9.1 MB/s eta 0:00:00
Collecting idna<4,>=2.5
  Downloading idna-3.4-py3-none-any.whl (61 kB)
    61.5/61.5 kB 6.3 MB/s eta 0:00:00
Installing collected packages: charset-normalizer, urllib3, idna, certifi, requests
Successfully installed certifi-2022.12.7 charset-normalizer-3.0.1 idna-3.4 requests-2.28.2 urllib3-1.26.14
root@885644b3510:/usr/src/app# python main.py
Connecting to DB to convert CSV into XML.
new file to convert: '/csv/athlete_events.csv'
A cidade e: Barcelona
A cidade e: London
A cidade e: Antwerpen
A cidade e: Paris
A cidade e: Calgary
A cidade e: Calgary
A cidade e: Albertville
A cidade e: Albertville
A cidade e: Lillehammer
A cidade e: Lillehammer
A cidade e: Albertville
A cidade e: Albertville
A cidade e: Albertville
A cidade e: Albertville
A cidade e: Albertville
A cidade e: Lillehammer
```

```
A cidade e: Lillehammer
A cidade e: Albertville
A cidade e: Albertville
A cidade e: Albertville
A cidade e: Albertville
A cidade e: Lillehammer
A cidade e: Lillehammer
A cidade e: Lillehammer
A cidade e: Los Angeles
A cidade e: Los Angeles
A cidade e: Salt Lake City
new xml file generated: '/shared/output/a1a5e393-03a4-4c3a-8094-3a2f2e2ce20d.xml'
Connecting to DB to read and insert XML file.
Connection successful.
Attempting insertion.
Dados inseridos com sucesso!
Connecting to DB to insert XML document on imported_documents.
Connection successful.
Attempting insertion.
Dados inseridos com sucesso!
```

6. Conversão do ficheiro CSV em XML

api-entities

```

src > api > entities > main.py > ...
25 @app.route('/api/season/', methods=['GET'])
26 def get_seasons():
27     rel_cursor = connect_db_rel()
28     rel_cursor.execute("SELECT id, season FROM season")
29     return [Season(row[1]).to_json() for row in rel_cursor]
30
31
32 @app.route('/api/season/create', methods=['POST'])
33 def post_seasons():
34     data = request.get_json()
35     season = str(data.get('season'))
36     connection = psycopg2.connect(
37         host='db-rel', database='is', user='is', password='is')
38     cursor = connection.cursor()
39     try:
40         cursor.execute("INSERT INTO season (season) VALUES(%s)", (season,))
41         connection.commit()
42         return jsonify({'status': 'success', 'your_data': season}), 201
43     except:
44         pass
45
46
47 @app.route('/api/athlete', methods=['GET'])
48 def get_athletes():
49     rel_cursor = connect_db_rel()
50     rel_cursor.execute(
51         "SELECT name, sex, age, height, weight, team, noc, games, year, season, city, lat, lon, sport, event, medal, geom, id FROM athlete")
52     return [Athlete(row[0], row[1], row[2], row[3], row[4], row[5], row[6], row[7], row[8], row[9], row[10], row[11], row[12], row[13], row[14], row[15], row[16], row[17]).to_json() for row in rel_cursor]
53
54
55 @app.route('/api/athlete/create', methods=['POST'])
56 def post_athletes():
57     data = request.get_json()
58     connection = psycopg2.connect(
59         host='db-rel', database='is', user='is', password='is')
60     cursor = connection.cursor()
61
62     query = f"INSERT INTO athlete(name, sex, age, height, weight, team, noc, games, year, season, city, lat, lon, sport, event, medal, geom) VALUES ('{data['name']}', '{data['sex']}', '{data['age']}', '{data['height']}', '{data['weight']}', '{data['team']}', '{data['noc']}', '{data['games']}', '{data['year']}', '{data['season']}', '{data['city']}', '{data['lat']}', '{data['lon']}', '{data['sport']}', '{data['event']}', '{data['medal']}', '{data['geom']}')"
63     cursor.execute(query)
64     connection.commit()
65     return data
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99

```

Na **main.py** do **api-entities** foi criado o **endpoint** da **API**, ou seja, a sua **rota** e fizemos a **conexão** á Base de Dados **db-rel** e as repetivas **queries**.

6. Conversão do ficheiro CSV em XML

api-entities

ThunderClient interface showing a GET request to `localhost2001/api/atlethe/`. The response is a JSON array of athlete data. The terminal shows the command `docker-compose up` being executed.

Google Chrome browser showing the JSON response from the API endpoint `localhost2001/api/atlethe/`. The response is a JSON array of athlete data, including details like age, city, event, games, geom, id, lat, lon, height, medal, name, noc, season, sex, sport, team, weight, and year.

Demonstração dos **endpoint's** a funcionar, através do **ThunderClient** e do browser **Google Chrome**.

6. Conversão do ficheiro CSV em XML

api-entities

The image shows two screenshots demonstrating API functionality. The top screenshot is from ThunderClient, showing a GET request to `localhost:20001/api/season/` with a status of 200 OK. The response is a JSON array of two objects, each containing an `id` and a `season` (Winter or Summer). The bottom screenshot is from Google Chrome, showing the same URL in the browser, displaying the same JSON array of two objects.

Demonstração dos **endpoint's** a funcionar, através do **ThunderClient** e do browser **Google Chrome**.

6. Conversão do ficheiro CSV em XML

migrator

```

src > daemon > migrator > migrator.py ...
18
19 if __name__ == "__main__":
20     db_org = psycopg2.connect(
21         host="db-xml2", database="is", user="is", password="is")
22     db_rel = psycopg2.connect(
23         host="db-rel2", database="is", user="is", password="is")
24
25     while True:
26
27         # Connect to both databases
28         db_org = None
29         db_rel = None
30
31         try:
32             db_org = psycopg2.connect(
33                 host="db-xml2", database="is", user="is", password="is")
34             db_rel = psycopg2.connect(
35                 host="db-rel2", database="is", user="is", password="is")
36         except OperationalError as err:
37             print(psycopg2_exception(err))
38
39         if db_rel is None or db_org is None:
40             continue
41
42         xml_cursor = db_org.cursor()
43         print("Checking updates...")
44         # TODO: is Execute a SELECT query to check for any changes on the table
45         resp = requests.get(
46             url="http://api-entities2:8080/api/season", params={})
47         # Val buscar os dados da tag season
48         seasons = {x.get("season"): x.get("id") for x in resp.json()}
49         query = """(with athletes as (select unnest(xpath('//athlete',xml)) as athletes from imported_documents)
50             select DISTINCT(xpath('//athlete/competition/season/text()',athletes))[1]:text as Season
51             FROM athletes
52             GROUP BY Season)"""
53         xml_cursor.execute(query)
54
55         for row in xml_cursor:
56             if row[0] not in seasons.keys():
57                 x = row[0]
58                 print(x)
59
60                 resp = requests.post(
61                     url="http://api-entities2:8080/api/season/create", json={"season": x})
62                 resp = requests.get(
63                     url="http://api-entities2:8080/api/season", params={})
64                 seasons = {x.get("season"): x.get("id") for x in resp.json()} #
65
66                 respRest = requests.get(
67                     url="http://api-entities2:8080/api/athlete", params={})
68                 athletes = {x.get("name") for x in respRest.json()} # : x.get("id")
69                 query2 = """ with athletes as (select unnest(xpath('//athlete/sex/..',xml)) as athletes_with_sex from imported_documents)
70                     select(xpath('//athlete/@name',athletes_with_sex))[1]:text as name,
71                         (xpath('//athlete/sex/text()',athletes_with_sex))[1]:text as sex,
72                         (xpath('//athlete/age/text()',athletes_with_sex))[1]:text as age,
73                         (xpath('//athlete/height/text()',athletes_with_sex))[1]:text as height,
74                         (xpath('//athlete/weight/text()',athletes_with_sex))[1]:text as weight,
75                         (xpath('//athlete/country/team/text()',athletes_with_sex))[1]:text as team,
76                         (xpath('//athlete/country/noc/text()',athletes_with_sex))[1]:text as noc,
77                         (xpath('//athlete/competition/games/text()',athletes_with_sex))[1]:text as games,
78                         (xpath('//athlete/competition/year/text()',athletes_with_sex))[1]:text as year,
79                         (xpath('//athlete/competition/season/text()',athletes_with_sex))[1]:text as season,
80                         (xpath('//athlete/competition/city/text()',athletes_with_sex))[1]:text as city,
81                         (xpath('//athlete/competition/coordinates/lat/text()',athletes_with_sex))[1]:text as lat,
82                         (xpath('//athlete/competition/coordinates/lon/text()',athletes_with_sex))[1]:text as lon,
83                         (xpath('//athlete/competition/statsBySport/sport/text()',athletes_with_sex))[1]:text as sport,
84                         (xpath('//athlete/competition/statsBySport/event/text()',athletes_with_sex))[1]:text as event,
85                         (xpath('//athlete/competition/statsBySport/medal/text()',athletes_with_sex))[1]:text as medal
86                     FROM athletes;"""

```

Foi realizada a **conexão** á base de dados **db-xml** e **db-rel**, foi feita a **query Xpath** para as tabelas da **db-rel** e verificou-se se existiam novos dados a enviar.

6. Conversão do ficheiro CSV em XML

migrator

```

grator > main.py ...
(xpath('//atlethe/competition/statsBySport/event/text()',atlethes_with_sex))[1]:text as
(xpath('//atlethe/competition/statsBySport/medal/text()',atlethes_with_sex))[1]:text as
FROM atlethes;"""
xml_cursor.execute(query2)
for row in xml_cursor: # nome das colunas do xml
    if row[0] not in atlethes:
        x = row[0]
        data = {
            "name": row[0],
            "sex": row[1],
            "age": row[2],
            "height": row[3],
            "weight": row[4],
            "team": row[5],
            "noc": row[6],
            "games": row[7],
            "year": row[8],
            "season": seasons.get(row[9]),
            "city": row[10],
            "lat": row[11],
            "lon": row[12],
            "sport": row[13],
            "event": remove_special_characters(row[14]),
            "medal": row[15],
        }
        url = "http://api-entities2:8080/api/atlethe/create"
        # headers = {"Content-Type": "application/json"}
        response = requests.post(url, json=data)
        print(response.json())
# !TODO: 2- Execute a SELECT queries with xpath to retrieve the data we want to store in the relational db
# !TODO: 3- Execute INSERT queries in the destination db
# !TODO: 4- Make sure we store somehow in the origin database that certain records were already migrated.
# Change the db structure if needed.

db_org.close()
db_dst.close()

time.sleep(POLLING_FREQ)

```

Docker Desktop interface showing the logs of the 'api-entities2' container. The logs display a series of HTTP POST requests to the '/api/atlethe/create' endpoint, each returning a 200 status code. The requests are made from the 'migrator' container, which is running on the host. The logs also show the container's resource usage: RAM 3.40GB and CPU 0.09%.

Como se pode verificar o envio dos dados está a ser executado.

7. Visualização de dados geográficos

api-gis

```

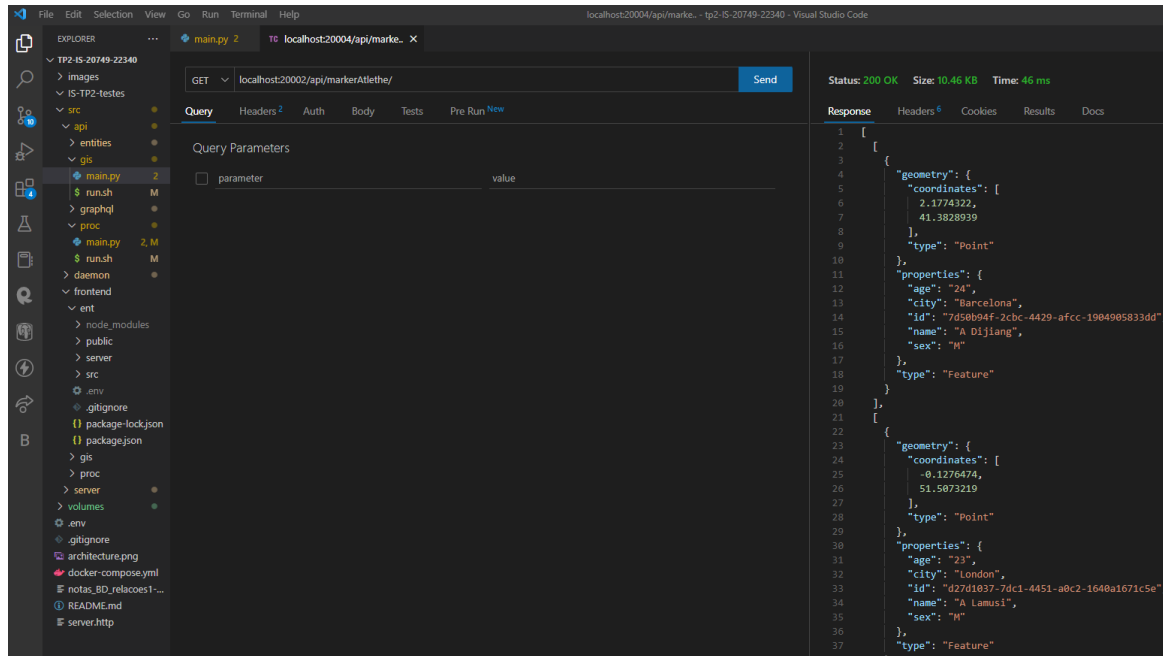
1  import sys
2  import psycopg2
3
4  from flask import Flask, request
5  from flask_cors import CORS
6
7  PORT = int(sys.argv[1]) if len(sys.argv) >= 2 else 9000
8
9  app = Flask(__name__)
10 CORS(app)
11 app.config["DEBUG"] = True
12
13 @app.route('/api/markerAtlethe/', methods=['GET'])
14 def get_markers_atlethes():
15
16     connection = psycopg2.connect(
17         host="db-rel2", database='is', user='is', password='is')
18     cursor = connection.cursor()
19
20     query = f"SELECT (SELECT jsonb_build_object('type', 'Feature', 'geometry', ST_AsGeoJSON(atlethe.geom)::jsonb, 'properties', to_jsonb(t.*) - 'geom') AS json FROM (VALUES (atlethe.id, atlethe.name, atlethe.c"
21
22     cursor.execute(query)
23     connection.commit()
24     data = cursor.fetchall()
25
26     connection.close()
27     return data
28
29
30
31 if __name__ == '__main__':
32     app.run(host="0.0.0.0", port=PORT)
33

```

Foi realizada a **conexão** á base de dados **db-rel** para realizar a query e respetivo **fetchall**(retorna uma lista de tuplas) destes de forma a que fossem obtidos na **rota** da **API**.

7. Visualização de dados geográficos

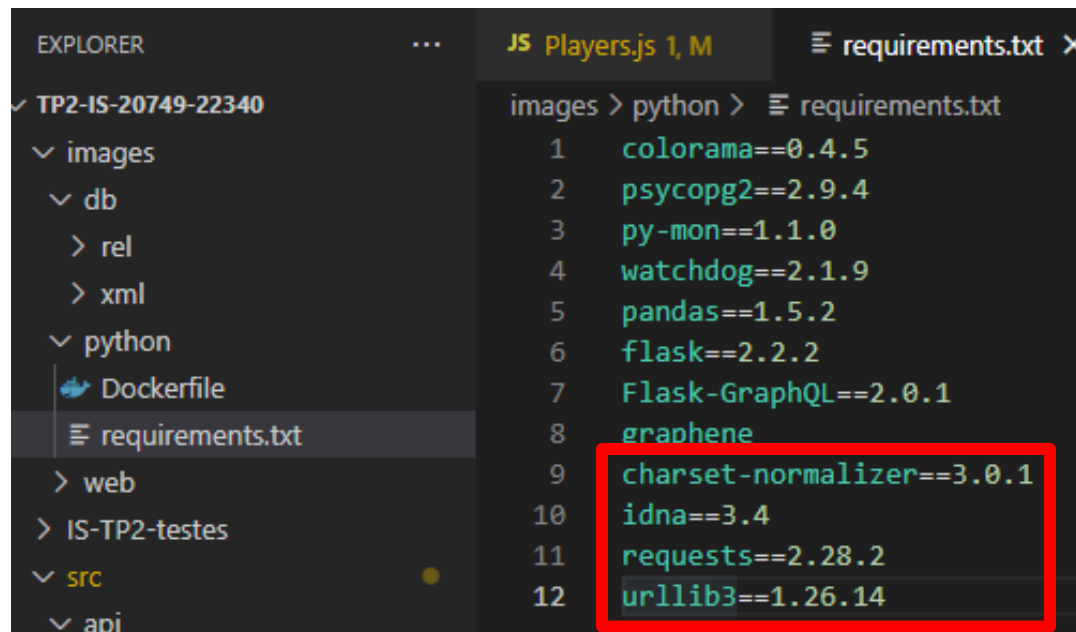
api-gis



Demonstração dos **endpoint's** a funcionar, através do **ThunderClient** e do browser **Google Chrome**.

■ Alteração no ficheiro requirements

Requirements.txt => sempre que corrámos algum container tínhamos de fazer **pip install** para fazer algumas instalações, de modo que fosse possível correr a **main.py**. Então decidimos fazer as seguintes alterações para não acontecer o erro.



```

EXPLORER
TP2-IS-20749-22340
  images
    db
      rel
      xml
    python
      Dockerfile
      requirements.txt
    web
    IS-TP2-testes
  src
    api

JS Players.js 1, M
requirements.txt
images > python > requirements.txt
1  colorama==0.4.5
2  psycopg2==2.9.4
3  py-mon==1.1.0
4  watchdog==2.1.9
5  pandas==1.5.2
6  flask==2.2.2
7  Flask-GraphQL==2.0.1
8  graphene
9  charset-normalizer==3.0.1
10 idna==3.4
11 requests==2.28.2
12 urllib3==1.26.14
  
```

■ Instalações para realizar o frontend

Sempre que corríamos o **container** tínhamos de realizar o comando **pip install** para fazer algumas instalações, de modo que fosse possível visualizar o **conteúdo do frontend** desenvolvido.

```
C:\Windows\System32\cmd.exe - docker-compose run frontend-ent2 /bin/bash
C:\Users\Basilio Barbosa\Desktop\IS\TRABALHOS_PRATICOS\TP2\tp2-IS-20749-22340>docker-compose run frontend-ent2 /bin/bash
[+] Running 1/0
  Container api-entities2 Created                                0.0s
  Container frontend-ent2 Started                               1.0s
root@dec51440181b:/usr/src/app# npm i react-data-table-component styled-components
added 10 packages, and audited 1520 packages in 6s
236 packages are looking for funding
  run `npm fund` for details
7 high severity vulnerabilities
To address issues that do not require attention, run:
  npm audit fix
To address all issues (including breaking changes), run:
  npm audit fix --force
Run `npm audit` for details.
npm notice New minor version of npm available! 9.2.0 -> 9.3.0
npm notice Changelog: https://github.com/npm/cli/releases/tag/v9.3.0
npm notice Run `npm install -g npm@9.3.0` to update!
npm notice
root@dec51440181b:/usr/src/app#
```

```
C:\Windows\System32\cmd.exe - docker-compose run frontend-ent2 /bin/bash
- Container gis-updater2 Started                                2.9s
- Container frontend-proc2 Started                              2.7s
C:\Users\Basilio Barbosa\Desktop\IS\TRABALHOS_PRATICOS\TP2\tp2-IS-20749-22340>docker-compose run frontend-ent2 /bin/bash
[+] Running 1/0
  Container api-entities2 Running                               0.0s
root@c87a7ebbd8:/usr/src/app# npm i react-fetch-hook
added 1 package, and audited 1503 packages in 3s
234 packages are looking for funding
  run `npm fund` for details
7 high severity vulnerabilities
To address issues that do not require attention, run:
  npm audit fix
To address all issues (including breaking changes), run:
  npm audit fix --force
Run `npm audit` for details.
npm notice New minor version of npm available! 9.2.0 -> 9.3.0
npm notice Changelog: https://github.com/npm/cli/releases/tag/v9.3.0
npm notice Run `npm install -g npm@9.3.0` to update!
npm notice
root@c87a7ebbd8:/usr/src/app#
```

```
C:\Windows\System32\cmd.exe - docker-compose run frontend-ent2 /bin/bash
- Container gis-updater2 Started                                4.0s
- Container frontend-gis2 Started                               4.1s
- Container frontend-proc2 Started                              3.0s
C:\Users\Basilio Barbosa\Desktop\IS\TRABALHOS_PRATICOS\TP2\tp2-IS-20749-22340>docker-compose run frontend-ent2 /bin/bash
[+] Running 1/0
  Container api-entities2 Running                               0.0s
root@b1ba5bc01bfa:/usr/src/app# npm install eslint-plugin-react-hooks@next
added 1 package, changed 1 package, and audited 1521 packages in 4s
236 packages are looking for funding
  run `npm fund` for details
7 high severity vulnerabilities
To address issues that do not require attention, run:
  npm audit fix
To address all issues (including breaking changes), run:
  npm audit fix --force
Run `npm audit` for details.
npm notice New minor version of npm available! 9.2.0 -> 9.3.0
npm notice Changelog: https://github.com/npm/cli/releases/tag/v9.3.0
npm notice Run `npm install -g npm@9.3.0` to update!
npm notice
root@b1ba5bc01bfa:/usr/src/app#
```

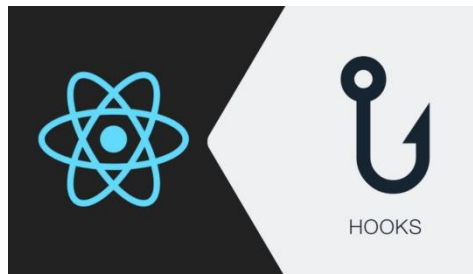
```
C:\Windows\System32\cmd.exe - docker-compose run --rm frontend-ent2 /bin/bash
root@3ec7fca464a3:/usr/src/app# npm install axios
up to date, audited 1507 packages in 3s
234 packages are looking for funding
  run `npm fund` for details
7 high severity vulnerabilities
To address issues that do not require attention, run:
  npm audit fix
To address all issues (including breaking changes), run:
  npm audit fix --force
Run `npm audit` for details.
npm notice New minor version of npm available! 9.2.0 -> 9.3.0
npm notice Changelog: https://github.com/npm/cli/releases/tag/v9.3.0
npm notice Run `npm install -g npm@9.3.0` to update!
npm notice
root@3ec7fca464a3:/usr/src/app# npm install hook
added 3 packages, and audited 1510 packages in 5s
234 packages are looking for funding
  run `npm fund` for details
7 high severity vulnerabilities
To address issues that do not require attention, run:
  npm audit fix
To address all issues (including breaking changes), run:
  npm audit fix --force
Run `npm audit` for details.
root@3ec7fca464a3:/usr/src/app# npm i react-fetch-hook
```

■ Funcionalidades Frontend

Axios => é um cliente HTTP promise-based para node.js e para o navegador.
No lado do servidor, ele usa o módulo http node.js nativo, enquanto no cliente (navegador) usa XMLHttpRequests

A X I O S

Async Hooks => é um módulo principal no Node.js que fornece uma API para rastrear o tempo de vida de recursos assíncronos num aplicativo Node.



■ Funcionalidades Frontend

useEffect Hook => permite executar efeitos colaterais nos componentes, tais como buscar dados, atualizar diretamente o DOM e temporizadores.

useState => é um **React Hook** que permite adicionar uma variável de estado ao seu componente.

MUI Core => contém bibliotecas de componentes fundamentais do **React UI** para enviar novos recursos mais rapidamente.

Material UI => é uma biblioteca abrangente de componentes que apresenta a nossa implementação do sistema **Material Design do Google**.



8. Frontend-ent

```

File Edit Selection View Go Run Terminal Help
EXPLORER
TP2-IS-20749-22340
  > images
  > IS-TP2-testes
  > src
    > api
    > entities
    > gis
    > main.py
    > run.sh
    > graphql
    > proc
    > main.py
    > run.sh
    > daemon
    > frontend
  > TP2-IS-20749-22340
    > src
      > frontend
        > ent
          > src
            > Tables
              > JS
                > Athlete.js
                  > AthleteTables.jsx 1
                    1 import React from "react";
                    2 import AthleteTables from "../AthleteTables";
                    3
                    4 function Athlete() {
                    5   return (
                    6     <div className="d-flex flex-column align-items-center">
                    7       <h1>Dados API Athletes</h1>
                    8     </div>
                    9     <AthleteTables />
                    10   );
                    11 }
                    12
                    13 export default Athlete;
  
```

Codificação para mostrar a **tabela** dos **dados da API Atheletes**.

```

File Edit Selection View Go Run Terminal Help
EXPLORER
TP2-IS-20749-22340
  > images
  > IS-TP2-testes
  > src
    > api
    > entities
    > gis
    > main.py
    > run.sh
    > graphql
    > proc
    > main.py
    > run.sh
    > daemon
    > frontend
      > ent
        > node_modules
        > public
        > server
        > src
          > Layout
          > JS
            > Content.js
            > Menu.js
            > Sections.js
            > Tables
              > JS
                > Athlete.js
                > AthleteTa... 1
                > JS
                  > Season.js
                    1 import { useEffect, useState } from "react";
                    2 import axios from "axios";
                    3 import {
                    4   Paper,
                    5   Table,
                    6   TableBody,
                    7   TableCell,
                    8   TableContainer,
                    9   TableHead,
                    10   TableRow
                    11 } from "@mui/material";
                    12
                    13 function Season() {
                    14   const [season, setSeasons] = useState([])
                    15
                    16   useEffect(() => {
                    17     fetchData()
                    18   }, [])
                    19
                    20   const fetchData = async () => {
                    21     const { data } = await axios.get("http://localhost:20001/api/season")
                    22
                    23     setSeasons(data)
                    24   }
                    25
                    26   return (
                    27     <div>
                    28       <h1>Dados API Season</h1>
                    29       <TableContainer component={Paper}>
                    30         <Table sx={{ minWidth: 650 }} aria-label="simple table">
                    31           <TableHead>
                    32             <TableRow>
                    33               <TableCell>Season Name</TableCell>
                    34             </TableRow>
                    35           </TableHead>
                    36           <TableBody>
                    37             {
                    38               season.map(row => (
                    39                 <TableRow
                    40                   key={row.id}
                    41                   style={{ background: "gray", color: "black" }}
                    42                 >
                    43                   <TableCell>{row.name}</TableCell>
                    44                 </TableRow>
                    45               )
                    46             )
                    47           </TableBody>
                    48         </Table>
                    49       </TableContainer>
                    50     </div>
                    51   );
                    52 }
  
```

8. Frontend-ent

Codificação para passar os dados
na tabela.

```

1  import React, { useEffect, useState } from 'react'
2  import axios from 'axios';
3  import DataTable from 'react-data-table-component'
4
5
6  const AtletheTables = () => {
7    const [search, setSearch] = useState("");
8    const [atlethe, setAtlethes] = useState([]);
9    const [filterAtlethe, setFilterAtlethe] = useState({});
10
11
12    const getAtlethes = async () => {
13      try {
14        const response = await axios.get("http://localhost:20001/api/atlethe");
15        setAtlethes(response.data);
16        setFilterAtlethe(response.data);
17      } catch (error) {
18        console.log(error);
19      }
20    };
21
22    const columns = [
23      {
24        name: <h4>Atlethe Name</h4>,
25        selector: row => row.name
26      },
27      {
28        name: <h4>Age</h4>,
29        selector: row => row.age
30      },
31      {
32        name: <h4>Height</h4>,
33        selector: row => row.height
34      },
35      {
36        name: <h4>Weight</h4>,
37        selector: row => row.weight
38      },
39      {
40        name: <h4>Team</h4>,
41        selector: row => row.team
42      },
43    ];
44  }
45
46  export default AtletheTables
  
```

```

1  import React, { useEffect, useState } from 'react'
2  import axios from 'axios';
3  import DataTable from 'react-data-table-component'
4
5
6  const AtletheTables = () => {
7    const [search, setSearch] = useState("");
8    const [atlethe, setAtlethes] = useState([]);
9    const [filterAtlethe, setFilterAtlethe] = useState({});
10
11
12    const getAtlethes = async () => {
13      try {
14        const response = await axios.get("http://localhost:20001/api/atlethe");
15        setAtlethes(response.data);
16        setFilterAtlethe(response.data);
17      } catch (error) {
18        console.log(error);
19      }
20    };
21
22    const columns = [
23      {
24        name: <h4>Atlethe Name</h4>,
25        selector: row => row.name
26      },
27      {
28        name: <h4>Age</h4>,
29        selector: row => row.age
30      },
31      {
32        name: <h4>Height</h4>,
33        selector: row => row.height
34      },
35      {
36        name: <h4>Weight</h4>,
37        selector: row => row.weight
38      },
39      {
40        name: <h4>Team</h4>,
41        selector: row => row.team
42      },
43    ];
44
45    useEffect(() => {
46      getAtlethes();
47    }, []);
48
49    useEffect(() => {
50      const result = atlethe.filter(atlethe => {
51        return atlethe.name.toLowerCase().match(search.toLowerCase()) + atlethe.city.toLowerCase().match(search.toLowerCase()) + atlethe.team.toLowerCase().match(search.toLowerCase());
52      });
53      setFilterAtlethe(result);
54    }, [search]); // [atlethe, search]
55
56    return <DataTable
57      title="Lista de Atletas"
58      columns={columns}
59      data={filterAtlethe}
60      pagination
61      fixedHeader
62      fixedHeaderScrollHeight="450px"
63      highlightOnHover
64      subHeader
65      subHeaderComponent={
66        <input
67          type="text"
68          placeholder="Pesquisa (name/team/city)"
69          className="w-25 form-control"
70          value={search}
71          onChange={(e) => setSearch(e.target.value)}
72        />
73      />
74
75    export default AtletheTables
  
```

8. Frontend-ent

Na **API Athletes** foi introduzida a funcionalidade para **pesquisar** na **tabela** por **nome, equipa ou cidade do atleta**.

The image displays three screenshots of the API Athletes frontend and the DevTools network tab.

Dados API Season: The page shows a form with a 'Season Name' input field. The input field contains the text 'Winter'. Below the input field, there is a 'Submit' button.

Dados API Athletes: The page shows a table of athletes. The table has columns: Athlete Name, Age, Height, Weight, Team, NOC, Games, Year, and City. The table contains 10 rows of data. A search bar is located at the top right of the table, with the placeholder text 'Pesquisa (nome/equipa/city)'.

DevTools Network Tab: The network tab shows the response of the API. The response is a JSON array of objects. The first object is:


```
{
      "id": "4f59-413e-b4c2-4d3958419ad1",
      "season": "Winter",
      "age": 24,
      "city": "Barcelona",
      "event": "Basketball",
      "height": 188,
      "weight": 82,
      "team": "Spain",
      "noc": "ESP",
      "games": 1992,
      "year": 1992,
      "city": "Barcelona"
    }
```

8. Frontend-gis

```

File Edit Selection View Go Run Terminal Help
ObjectMarkers - tp2-IS-20749-22340
EXPLORER
  TP2-IS-20749-22340
    src
      > api
      > entities
      > gis
      > main.py
      > run.sh
      > graphql
      > proc
      > main.py
      > run.sh
      > daemon
      > frontend
      > ent
      > node_modules
      > public
      > server
      > src
      > .env
      > .gitignore
      > package-lock.json
      > package.json
      > gis
      > node_modules
      > public
      > server
      > src
      > Layout
      > JS Content.js
      > JS Menu.js
      > JS Sections.js
      > Map
      > ObjectMar... 1
      > ObjectMar... 2
      > ObjectMarkers...
      > JS App.css
      > JS App.js
      > index.css
      > index.js
    PROBLEMS
    OUTPUT
    DEBUG CONSOLE
    TERMINAL
    COMMENTS
  src > frontend > gis > src > Map > JS ObjectMarkers.js > LIST_PROPERTIES > label
  1
  2
  3
  4
  5
  6
  7
  8
  9
  10
  11
  12
  13
  14
  15
  16
  17
  18
  19
  20
  21
  22
  23
  24
  25
  26
  27
  28
  29
  30
  31
  32
  33
  34
  35
  36
  37
  38
  39
  40
  41
  42
  43
  const LIST_PROPERTIES = [
    { key: "city", label: "city", Icon: FlagIcon },
    { key: "age", label: "age", Icon: ContactsIcon },
    { key: "sex", label: "gender", Icon: PictureInPictureAltIcon },
  ];
  export function ObjectMarker({ geoJSON }) {
    const properties = geoJSON.properties;
    const { id, imgUrl, name } = properties;
    const coordinates = geoJSON.geometry.coordinates;
    return (
      <Marker>
        <position>{coordinates}</position>
        <icon>leafletIcon({
          iconUrl: imgUrl,
          iconRetinaUrl: imgUrl,
          iconSize: point(50, 50),
        })</icon>
      </Marker>
    );
  }
  <Popup>
    <List dense={true}>
      <ListItem>
        <Avatar alt={name} src={imgUrl} />
      </ListItem>
      <ListItemText primary={name} />
    </List>
  </Popup>
  {LIST_PROPERTIES.map(({ key, label, Icon }) => (
    <ObjectMarker key={key} geoJSON={geoJSON} />
  ))}
  </Map>
  </>
  export default ObjectMap;

```

```

File Edit Selection View Go Run Terminal Help
ObjectMarkersAtletes.js
EXPLORER
  TP2-IS-20749-22340
    src
      > api
      > entities
      > gis
      > main.py
      > run.sh
      > graphql
      > proc
      > main.py
      > run.sh
      > daemon
      > frontend
      > ent
      > node_modules
      > public
      > server
      > src
      > .env
      > .gitignore
      > package-lock.json
      > package.json
      > gis
      > node_modules
      > public
      > server
      > src
      > Layout
      > JS Content.js
      > JS Menu.js
      > JS Sections.js
      > Map
      > ObjectMar... 1
      > ObjectMar... 2
      > ObjectMarkers...
      > JS App.css
      > JS App.js
      > index.css
      > index.js
    PROBLEMS
    OUTPUT
    DEBUG CONSOLE
    TERMINAL
    COMMENTS
  src > frontend > gis > src > Map > JS ObjectMarkersAtletes.js > ObjectMarkersGroup
  1
  2
  3
  4
  5
  6
  7
  8
  9
  10
  11
  12
  13
  14
  15
  16
  17
  18
  19
  20
  21
  22
  23
  24
  25
  26
  27
  28
  29
  30
  31
  32
  33
  34
  35
  36
  37
  38
  39
  40
  41
  42
  43
  import React, { useEffect, useState } from "react";
  import { LayerGroup, useMap } from "react-leaflet";
  import { ObjectMarker } from "../ObjectMarker";
  import axios from "axios";
  function ObjectMarkersGroup() {
    const map = useMap();
    const [atlethe, setAtlethe] = useState([]);
    const [bounds, setBounds] = useState(map.getBounds());
    const getMarkersAtlethes = async () => {
      let response = await axios.get("http://localhost:20002/api/markerAtlethe/");
      return response;
    };
    function trocarObjeto(atlethe) {
      return atlethe[0];
    }
    function addImage(atlethe) {
      atlethe.properties.imgUrl =
        "https://cdn-icons-png.flaticon.com/512/6534/6534578.png";
    }
    function changeCoordinates(atlethe) {
      let lon = atlethe.geometry.coordinates[0];
      let lat = atlethe.geometry.coordinates[1];
      let coordinates = [lat, lon];
      atlethe.geometry.coordinates = coordinates;
      return atlethe;
    }
    useEffect(() => {
      let atlethesObj = [];
      getMarkersAtlethes().then((response) => {
        response.data.map(atlethe => {
          atlethe = trocarObjeto(atlethe);
          atlethe = changeCoordinates(atlethe);
          addImage(atlethe);
          atlethesObj.push(atlethe);
        });
        setAtlethes(atlethesObj);
      });
    });
  }

```

A API foi consumida e foi adicionada uma imagem para mostrar os atletas, que por sua vez, contém os dados quando clicamos nesta.

8. Frontend-gis

Mapa com os dados de geolocalização dos atletas.



■ 9. Bibliografia e Referências Web

- <https://mui.com/material-ui/react-table/#custom-pagination-options>
- <https://www.digitalocean.com/community/tutorials/how-to-display-data-from-the-digitalocean-api-with-react>
- <https://react-spectrum.adobe.com/react-aria/useTable.html>
- <https://react-table-v7.tanstack.com/docs/api/useTable>
- <https://www.npmjs.com/package/react-fetch-hook>
- <https://www.npmjs.com/package/hooks>
- <https://hardcoded.medium.com/docker-compose-with-react-node-and-postgresql-a-multi-container-application-with-docker-a11197802e33>
- <https://www.tomray.dev/nestjs-docker-compose-postgres>
- <https://designcode.io/react-hooks-handbook-fetch-data-from-an-api>

■ 9. Bibliografia e Referências Web



- <https://www.positronx.io/how-to-use-fetch-api-to-get-data-in-react-with-rest-api/>
- <https://blog.logrocket.com/modern-api-data-fetching-methods-react/>
- <https://www.copycat.dev/blog/react-fetch/>
- <https://axios-http.com/docs/intro>
- <http://bboxfinder.com/>
- <https://medium.com/@zack.noah/creating-a-geospatial-web-app-using-docker-django-and-postgis-975bc24e7a6b>
- <https://gis-ops.com/flask-geo-api-tutorial-powerful-geospatial-flask-explained/>
- <https://realpython.com/location-based-app-with-geodjango-tutorial/>
- <https://medium.com/@manilwagle/geocoding-the-world-using-google-api-and-python-1f6b6fb6ca48>

o teu • de partida



Instituto Politécnico
de Viana do Castelo

www.ipvc.pt