



Título: WS - Assignment 1

Autores: Gonçalo Silva 103668, Guilherme Antunes 103600, Pedro Rasinhas 103541

Data: 16/04/2024

1. Introdução.....	2
2. Dados e Relações.....	2
3. Operações sobre os Dados.....	4
4. Funcionalidades da Aplicação.....	11
1. Listar jogadores e ordená-los/filtrá-los por diversas opções.....	12
2. Listar e Pesquisar Ligas, Equipas e Jogadores.....	13
3. Jogar o jogo da adivinha.....	14
4. Criar squads com jogadores à escolha do utilizador.....	14
5. Executar Aplicação.....	15
6. Trabalho Futuro.....	16
7. Conclusão.....	17

1. Introdução

Este trabalho foi desenvolvido no âmbito da Unidade Curricular de Web Semântica e visa o desenvolvimento de um sistema de informação baseado na Web.

O tema escolhido para explorar os objetivos propostos para o projeto foi o videojogo FIFA 24, oficialmente conhecido como EA SPORTS FC24, mais especificamente as cartas dos jogadores do mesmo. Cada carta corresponde a um jogador real e aos atributos jogáveis que o mesmo tem dentro do jogo. Cada jogador tem associado a si, para além dos atributos pessoais, um clube, uma nacionalidade, uma liga, um género, uma posição, etc.

A implementação descrita neste relatório procura explorar da maneira mais versátil possível as capacidades do GraphDB, do SPARQL e da modelação dos dados em RDF.

Para a realização deste projeto foi usada uma combinação de tecnologias tais como HTML, CSS e JavaScript para o frontend, Django para o backend, RDF N3 como estrutura para descrever os dados, GraphDB como base de dados e SPARQL como query language usada para fazer pesquisas sobre os dados.

2. Dados e Relações

Para o desenvolvimento deste trabalho obtivemos os dados relativos às cartas do FIFA 24 através de uma API [2], que disponibiliza os dados pretendidos, e dois websites que serviram de guia para o display dessa informação [1][3]. As entidades contempladas foram as seguintes:

- Player
- Gender
- Nationality
- Position
- Team
- League
- Squad
- Squad_Player

Na pasta *dataset* é possível encontrar o script *fifa.py* utilizado para recuperar todos os dados pretendidos, assim como a construção da estrutura RDF N3. Para além dos dados recuperados da API [2] foram selecionadas manualmente imagens dos logótipos das ligas que também são incluídas no ficheiro final. A opção pelo RDF N3 deveu-se a ser uma representação relativamente mais fácil para a compreensão humana do que a RDF XML e por conseguir condensar muito mais informação que a N-Triples. Na figura abaixo podemos observar o diagrama da estrutura dos dados.

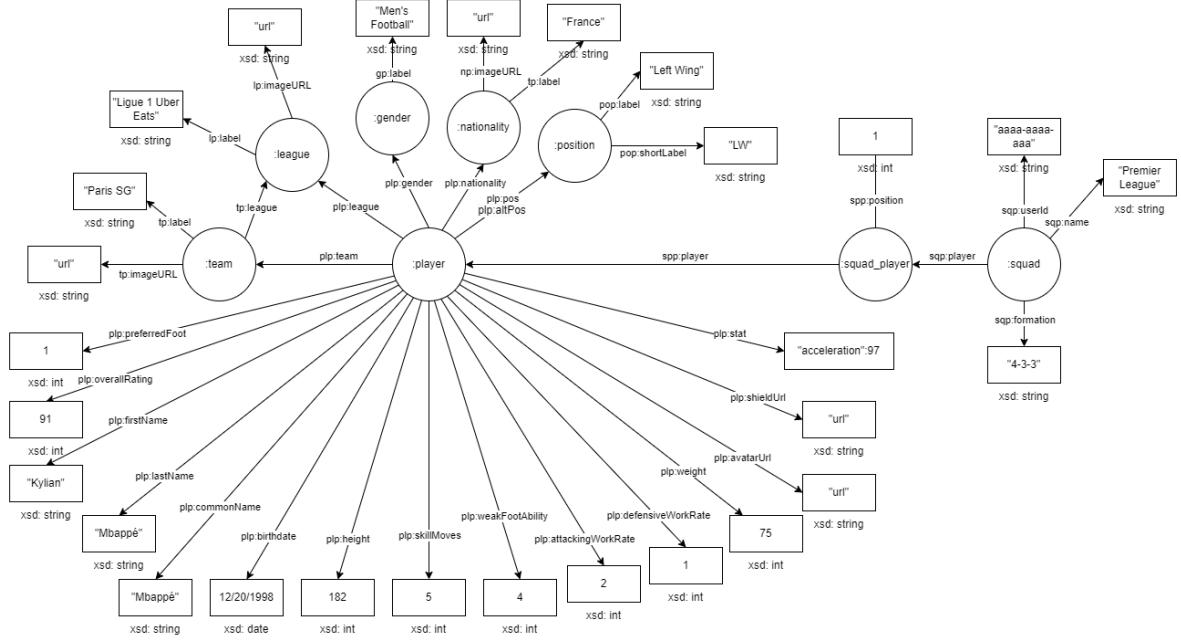


Figura 1: Esquema da Base de Dados

Nesta estrutura podemos observar a existência de um predicado “stat” referente às entidades player com um tipo de dados pouco ortodoxo. A decisão de tornar “stat” um objeto “literal” deveu-se à verificação de tempos insuportáveis na execução de queries que selecionavam todas as stats referentes a um jogador. Por isso passámos de uma estrutura como a da Figura 2 para uma com a da Figura 3.

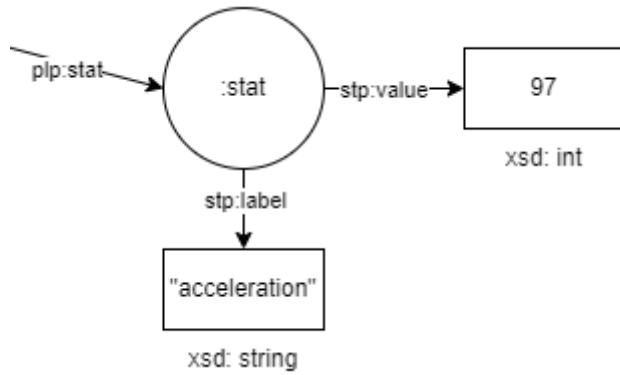


Figura 2: Entidade “stat” antiga



Figura 3: Atributo “stat”

Enquanto as entidades Squad e Squad_Player dependem diretamente do id do utilizador do django, obtido através de autenticação django, e podem ser criadas e

eliminadas, todas as informações relativas a outras entidades do sistema são exclusivas de leitura, pelo que o sistema não implementa queries de update sobre esses dados.

3. Operações sobre os Dados

No desenvolvimento deste projeto, as operações sobre a base de dados foram feitas através de SPARQL, uma standard query language usada em RDF triplestores.

No pasta *myFifa24/app/api* estão localizados os ficheiros python onde as queries são construídas e executadas pela biblioteca s4api.

As várias queries estão divididas em diferentes ficheiros, cada um focado em entidades distintas.

- game.py - queries para o jogo
- leagues.py - queries de ligas
- players.py - queries de jogadores e filtros de jogadores
- squads.py - queries de squads
- teams.py - queries de equipas

A fim de maximizar a exploração das capacidades da linguagem SPARQL tentámos aplicar a maior variedade de instruções possíveis em cada query, sem desvirtuar as funcionalidades propostas para o sistema.

```
def get_players_by_prop(start: int = 0, limit: int = 30, props: dict = None) -> list[dict]:  
  
    order = "DESC(?ovr) ?name"  
    name = ""  
    gender = ""  
    nationality = ""  
    team = ""  
    position = ""  
    extra_position = ""  
  
    if props:  
        for prop in props:  
            match prop["prop"]:  
                case "name":  
                    name += f'FILTER REGEX(?name, "{prop["value"]}")',  
"i")'  
                case "nationality":  
                    nationality += f'FILTER(?nationalityid =  
fifang:{prop["value"]})'  
                case "league":  
                    team = f'?teamid fifatp:league  
fifalg:{prop["value"]}'.  
                case "team":  
                    team = f'FILTER(?teamid = fifatg:{prop["value"]})'  
                case "gender":  
                    gender = f'FILTER(?genderid =  
fifagg:{prop["value"]})'  
                case "position":  
                    position = f'FILTER(?positionid =  
fifapog:{prop["value"]})'
```

```

extra_position = f"""UNION {{  

?playerid fifaplp:gender ?genderid .  

?genderid fifagp:label ?gender .  

?playerid fifaplp:altPos fifapog:{prop["value"]} .  

?playerid fifaplp:position ?positionid .  

?positionid fifapop:shortLabel ?position .  

?playerid fifaplp:nationality ?nationalityid .  

?nationalityid fifanp:imageUrl ?flag .  

?nationalityid fifanp:label ?nationality .  

?teamid fifatp:label ?team .  

?playerid fifaplp:team ?teamid .  

?teamid fifatp:imageUrl ?logo .  

?playerid fifaplp:overallRating ?ovr .  

?playerid fifaplp:firstName ?fName .  

?playerid fifaplp:lastName ?lName .  

OPTIONAL {{ ?playerid fifaplp:commonName ?cName .  

}}}  

        BIND(COALESCE(?cName, CONCAT(?fName, " ", ?lName))  

AS ?name)  

?playerid fifaplp:skillMoves ?skills .  

?playerid fifaplp:weakFootAbility ?weakfoot .  

?playerid fifaplp:attackingWorkRate ?attwr .  

?playerid fifaplp:defensiveWorkRate ?defwr .  

?playerid fifaplp:avatarUrl ?image .  

?playerid fifaplp:stat ?stat .  

} }"""  

    case "order":  

        signal = "ASC" if prop["value"][0] == "-" else  

"DESC"  

        ord = prop["value"][1:]  

        if ord == "ovr":  

            order = f"{signal}({?ovr}) ?name"  

        elif ord == "name":  

            order = f"{signal}({?name}) DESC({?ovr})"  

        else:  

            order = f"{signal}({{?ord}}) DESC({?ovr}) ?name"  

query = f"""  

PREFIX fifaplp: <http://fifa24/player/pred/>  

PREFIX fifang: <http://fifa24/nationality/guid/>  

PREFIX fifanp: <http://fifa24/nationality/pred/>  

PREFIX fifalg: <http://fifa24/league/guid/>  

PREFIX fifatg: <http://fifa24/team/guid/>  

PREFIX fifatp: <http://fifa24/team/pred/>  

PREFIX fifapog: <http://fifa24/position/guid/>  

PREFIX fifapop: <http://fifa24/position/pred/>  

PREFIX fifagg: <http://fifa24/gender/guid/>  

PREFIX fifagp: <http://fifa24/gender/pred/>  

SELECT ?playerid ?name ?flag ?nationality ?teamid ?team ?logo  

?position ?ovr ?gender ?image ?skills ?weakfoot ?attwr ?defwr  

(CONCAT("{{", GROUP_CONCAT(?stat; separator=", "), "}}") AS ?stats)  

WHERE {{  

    {{  

?playerid fifaplp:gender ?genderid .  

{gender}  

?genderid fifagp:label ?gender .  

?playerid fifaplp:position ?positionid .  

}}
```

```

{position}
?positionid fifapop:shortLabel ?position .
?playerid fifaplp:nationality ?nationalityid .
{nationality}
?nationalityid fifanp:imageUrl ?flag .
?nationalityid fifanp:label ?nationality .
?playerid fifaplp:team ?teamid .
{team}
?teamid fifatp:label ?team .
?teamid fifatp:imageUrl ?logo .
?playerid fifaplp:overallRating ?ovr .
?playerid fifaplp:firstName ?fName .
?playerid fifaplp:lastName ?lName .
OPTIONAL {{ ?playerid fifaplp:commonName ?cName . } }
{name}
BIND(COALESCE(?cName, CONCAT(?fName, " ", ?lName)) AS
?name)
?playerid fifaplp:skillMoves ?skills .
?playerid fifaplp:weakFootAbility ?weakfoot .
?playerid fifaplp:attackingWorkRate ?attwr .
?playerid fifaplp:defensiveWorkRate ?defwr .
?playerid fifaplp:avatarUrl ?image .
?playerid fifaplp:stat ?stat .
}}
{extra_position}
}}
GROUP BY ?playerid ?name ?flag ?nationality ?teamid ?team ?logo
?position ?ovr ?gender ?image ?skills ?weakfoot ?attwr ?defwr
ORDER BY {order}
OFFSET {start}
LIMIT {limit}
"""

```

Esta query gerada dinamicamente é responsável por filtrar os jogadores existentes no sistema pelos parâmetros de género, posição, nacionalidade, equipa, liga e nome. Em termos de particularidades esta query é capaz de ordenar os resultados por qualquer parâmetro selecionado, que existe na query, e ordena sempre por overall e nome como opções seguintes, tudo isto usando ASC e DESC. O OFFSET e o LIMIT também são variáveis pelo que é possível implementar paginação através desta query.

Para filtrar por um valor específico de cada opção ou guid são utilizados FILTERs que são adicionados à query mediante as opções de filtragem. Para filtrar por nome é ainda utilizado REGEX para verificar todos os jogadores que têm no nome a string introduzida, ignorando maiúsculas e minúsculas com a opção “I”. O nome por sua vez é obtido com recurso às funcionalidades OPTIONAL, BIND, COALESCE e CONCAT.

Existem jogadores com o predicado *commonName* que consiste no nome pelo qual são conhecidos, no entanto nem todos os jogadores têm esse predicado, assim utilizamos o OPTIONAL para recuperar os commonNames de todos os jogadores sem ocorrer um erro, uma vez que o OPTIONAL quando não encontra o predicado vai atribuir um valor nulo nessa variável. Aos jogadores que não têm *commonName* é considerado que o seu nome é a concatenação, utilizando o CONCAT, do seu primeiro nome com o último. Finalmente a variável ?name é atribuído através da instrução BIND que vai atribuir, pelos critérios anteriores o valor do nome a ?name. Esses critérios são aplicados pela COALESCE que entre o *commonName* e o *fullName* vai atribuir a ?name o primeiro entre eles que não for

nulo, garantindo assim que todos os jogadores com commonName tenham esse valor como nome.

Cada jogador tem uma posição principal, mas também pode ter posições alternativas. Quando filtramos por uma posição queremos que essa filtragem devolva todos os jogadores cuja posição principal seja a selecionada, mas também aqueles que têm como posição alternativa essa mesma posição. Para fazer isto utilizamos a mesma query anteriormente descrita para filtrar a posição principal e adicionamos, através do operador UNION, uma nova “query” que vai verificar todos os jogadores que têm a posição filtrada como posição alternativa, unindo assim ambos os conjuntos de dados.

Como foi referido nos Dados e Relações a entidade Stat passou a ser um atributo com um tipo de dados do tipo “chave”:“valor” em string. Como cada jogador tem mais do que uma stat elas são agrupadas por jogador usando o GROUP BY e são todas juntas numa estrutura {“chave”:“valor”,“chave”:“valor”,...} utilizando o GROUP_CONCAT e o CONCAT. Esta estrutura será facilmente convertível para dicionário python utilizando a função json.loads(), que converte aquela string em dicionário.

```
query = """
PREFIX fifanp: <http://fifa24/nationality/pred/>

SELECT ?nationality ?label
WHERE {
    ?nationality fifanp:label ?label .
}
ORDER BY ?label
"""
```

Para obter os valores a introduzir nas opções de <select> no frontend são utilizadas queries simples como a anterior, que retorna por ordem alfabética todas as nacionalidades e guids correspondentes.

```
query = f"""
PREFIX fifaplp: <http://fifa24/player/pred/>
PREFIX fifang: <http://fifa24/nationality/guid/>
PREFIX fifalg: <http://fifa24/league/guid/>
PREFIX fifatg: <http://fifa24/team/guid/>
PREFIX fifapog: <http://fifa24/position/guid/>
PREFIX fifagg: <http://fifa24/gender/guid/>

SELECT (COUNT(?playerid) AS ?total)
WHERE {{{
    ?playerid fifaplp:gender ?genderid .
    {gender}
    ?playerid fifaplp:position ?positionid .
    {position}
    ?playerid fifaplp:nationality ?nationalityid .
    {nationality}
    ?playerid fifaplp:team ?teamid .
    {team}
    ?playerid fifaplp:firstName ?fName .
    ?playerid fifaplp:lastName ?lName .
    OPTIONAL {{ ?playerid fifaplp:commonName ?cName . }}}
    {name}
}}}
```

```

        {extra_position}
    } }
"""

```

Para obter o número total de jogadores no sistema para utilizar na paginação é utilizada a query anterior que conta o número total de guids de jogadores, utilizando exatamente os mesmos critérios de filtragem que os inseridos na query maior.

Todas as filtragens por nome existentes no sistema utilizam a mesma estratégia que na query de filtragem de jogadores, exceptuando no jogo onde é utilizada uma query que retorna diretamente o nome do jogador e o mesmo é normalizado utilizando o unidecode() do python, de maneira a permitir que o utilizador consiga acertar no nome do jogador, mesmo que o teclado não esteja na linguagem em que o nome dele é escrito. Exemplo, letras e acentos de países escandinavos.

Para colocar o jogo em funcionamento são selecionados todos os jogadores não guarda-redes com um overall superior a 74 e através da função random.choice() do python é selecionado um. O objetivo seria efetuar esta seleção aleatória diretamente na query mas o mesmo revelou-se não ser possível.

```

def guess_stat(player_guid: str, stat: str, value: str) -> bool:

    obj = None
    if stat == "ovr":
        obj = f"\\"{value}\\"^^xsd:int"
    elif stat != "name":
        obj = f"\\"\\\"{stat}\\\"\\\":{value}\\""

    if stat == "ovr":
        args = f"<{player_guid}> fifaplp:overallRating {obj} ."
    elif stat == "name":
        return guess_name(player_guid, value)
    else:
        args = f"<{player_guid}> fifaplp:stat \\"{obj}\\" ."

    query = f"""
PREFIX fifaplp: <http://fifa24/player/pred/>

ASK {{ 
    {args}
}}}
"""

    return ask(query)

```

Para verificar se a tentativa do utilizador está correta quando este tenta adivinhar o valor de uma stat são utilizadas queries ASK como a mostrada acima que retornam verdadeiro ou falso mediante se a tentativa está correta.

Também são utilizadas ASK queries antes e depois de INSERTs e DELETEs para verificar a correta inserção/remoção/atualização dos valores em causa.

```

def create_squad(user_id: str, squad: dict) -> dict:

```

```

        status = ask(f"ASK {{ <http://fifa24/squad/guid/{squad["id"]}> ?p
?o } }")

    if status:
        return False

    squad_players = ""
    for player in squad["players"]:
        squad_players += f"fifasqg:{squad["id"]} fifasqp:player
fifaspq:{squad["id"]+player["pos"]} .\n"
        squad_players += f"fifaspq:{squad["id"]+player["pos"]} 
fifaspp:player {player["id"]} .\n"
        squad_players += f"fifaspq:{squad["id"]+player["pos"]} 
fifaspp:position \"{player["pos"]}\\"^^xsd:int .\n"

    query = """
PREFIX fifasqg: <http://fifa24/squad/guid/>
PREFIX fifasqp: <http://fifa24/squad/pred/>
PREFIX fifaspq: <http://fifa24/squad_player/guid/>
PREFIX fifaspp: <http://fifa24/squad_player/pred/>

INSERT DATA {{

    fifasqg:{squad["id"]} fifasqp:name
"squad["name"]"}^^xsd:string .
    fifasqg:{squad["id"]} fifasqp:formation
"squad["formation"]"}^^xsd:string .
    fifasqg:{squad["id"]} fifasqp:userId "{user_id}"^^xsd:string .
    {squad_players}
} }
"""

    update(query)

    return ask(f"ASK {{ <http://fifa24/squad/guid/{squad["id"]}> ?p ?o
} }")

```

Nesta query é possível verificar como são introduzidos novos Squads, criando novas entidades do tipo Squad_Player e Squad e associando este último ao id do utilizador na base de dados do Django. A garantia de um id por utilizador é feita através do sistema de autenticação próprio do Django. Como referido é possível observar que antes e depois da inserção de dados é efetuado um ASK para verificar o correto funcionamento da função.

```

def delete_squad(guid: str) -> dict:
    query = """
PREFIX fifasqg: <http://fifa24/squad/guid/>
PREFIX fifasqp: <http://fifa24/squad/pred/>

DELETE {{

    fifasqg:{guid} ?p1 ?o1 .
    ?squadPlayerId ?p2 ?o2 .
} }
WHERE {{

    fifasqg:{guid} ?p1 ?o1 .
    fifasqg:{guid} fifasqp:player ?squadPlayerId .
    ?squadPlayerId ?p2 ?o2 .
} }
"""

```

```

    update(query)

    return not ask(f"ASK {{ <http://fifa24/squad/guid/{guid}> ?p ?o
}}")

```

Esta query por sua vez é responsável por eliminar da base de dados o Squad e os Squad_Players criados pela query anterior a esta.

```

def update_squad(guid: str, squad: dict) -> dict:

    delete = ""
    insert = ""

    new_players = []
    old_players = []

    old_squad = get_squad_by_guid(guid)

    if not old_squad:
        return False

    for old_player, new_player in zip(old_squad["players"], squad["players"]):
        if old_player["id"] != new_player["id"]:
            old_players.append(old_player)
            new_players.append(new_player)

            delete += f"""
                fifasgg:{guid} fifasqp:player
fifasp:{old_player["squadPlayerId"]} .
                fifasp:{old_player["squadPlayerId"]} ?p{old_player["pos"]} }
?o{old_player["pos"]} .
"""

            if new_player["id"]:
                insert += f"""
                    fifasgg:{guid} fifasqp:player
fifasp:{guid+new_player["pos"]} .
                    fifasp:{old_player["squadPlayerId"]} fifaspp:player
fifasp:{new_player["id"]} .
                    fifasp:{old_player["squadPlayerId"]} fifaspp:position
"{new_player["pos"]}"^^xsd:int .
"""

    query = f"""
PREFIX fifasgg: <http://fifa24/squad/guid/>
PREFIX fifasqp: <http://fifa24/squad/pred/>
PREFIX fifasp:{<http://fifa24/squad_player/guid/>}
PREFIX fifaspp: <http://fifa24/squad_player/pred/>

DELETE {{ {delete} }} }
WHERE {{ {delete} }} }
INSERT DATA{{

```

```

        {insert}
    }
"""

update(query)

query = ""

for player in new_players:
    if not ask(f"ASK {{ <{player['squadPlayerId']}> fifaspp:player
<{player['id']}> }}"):
        return False

for player in old_players:
    if ask(f"ASK {{ <{player['squadPlayerId']}> fifaspp:player
<{player['id']}> }}"):
        return False

return True

```

Por fim temos uma query de atualização que conjuga INSERTs e DELETEs para atualizar os Squad_Players de uma Squad. Essa atualização é feita obtendo o estado guardado da Squad a alterar, compará-la com o estado atual e eliminando os jogadores guardados que já não existem, substituindo-os pelos jogadores novos no estado atual. Mais uma vez são utilizados ASKs para verificar o correto funcionamento do sistema.

4. Funcionalidades da Aplicação

Em termos de funcionalidades a nossa aplicação conta com as seguintes:

1. Listar jogadores e ordená-los/filtrá-los por diversas opções

The screenshot shows the 'Players' section of the MyFifa24 website. At the top, there are four dropdown menus: 'Nationality' (Select Nationality), 'Team' (Select Team), 'Gender' (Select Gender), and 'Position' (Select Position). Below these is a search bar labeled 'Search by name'. The main area is a table listing seven players:

		Team	Nationality	OVR	Position	Gender	Skill Moves	Weak Foot Ability	Work Rate	PAC	SHO	PAS	DRI	DEF	PHY
	Alexia Putellas			91	CM	W	5★	5★	H\M	82	90	91	92	72	78
	Erling Haaland			91	ST	M	3★	3★	H\M	89	93	66	80	45	88
	Kevin De Bruyne			91	CM	M	4★	5★	H\M	72	88	94	87	65	78
	Kylian Mbappé			91	ST	M	5★	4★	H\L	97	90	80	92	36	78
	Aitana Bonmatí			90	CM	W	4★	5★	H\H	81	84	83	91	75	73
	Caroline Graham Hansen			90	RW	W	5★	4★	H\M	89	86	88	90	47	75
	Harry Kane			90	ST	M	3★	5★	H\H	69	93	84	83	49	83

A página de “Players” é usada como página principal depois de efetuar o login. Aqui estão disponíveis todos os jogadores da base de dados, masculinos e femininos, e é possível filtrá-los através dos filtros localizados no topo da tabela. Para além desses filtros é possível encontrar paginação no fundo da página e é possível trocar a ordem de ordenação por overall carregando livremente na label OVR. O botão que se assemelha a um caixote do lixo serve apenas para dar reset a todos os filtros através de um click.

Cada jogador tem uma página própria com toda a sua informação que pode ser acedida carregando no nome do jogador.

The screenshot shows the detailed profile of player Alexia Putellas. At the top left is her gold-rated card with an overall rating of 91 and position CM. Her stats include Pace (82), Shooting (90), Passing (91), Dribbling (92), Defending (72), and Physicality (78). The main table lists her detailed attributes:

	Pace	Shooting	Passing
Acceleration	81	Positioning	91
Sprint Speed	82	Finishing	91
		Shot Power	86
		Long Shots	89
		Volley	90
		Penalties	91
			Curve
	Dribbling	Defending	Physicality
Agility	90	Interceptions	78
Balance	89	Heading Accuracy	74
Reactions	92	Def. Awareness	60
Ball Control	94	Standing Tackle	81
Dribbling	92	Sliding Tackle	64
Composure	92		

Below the table, her player statistics are listed:

Name	Alexia Putellas
Team	
Nation	Spain
Gender	Women's Football
Skills	5★
Weak Foot	5★
Foot	Left
Height	172 cm

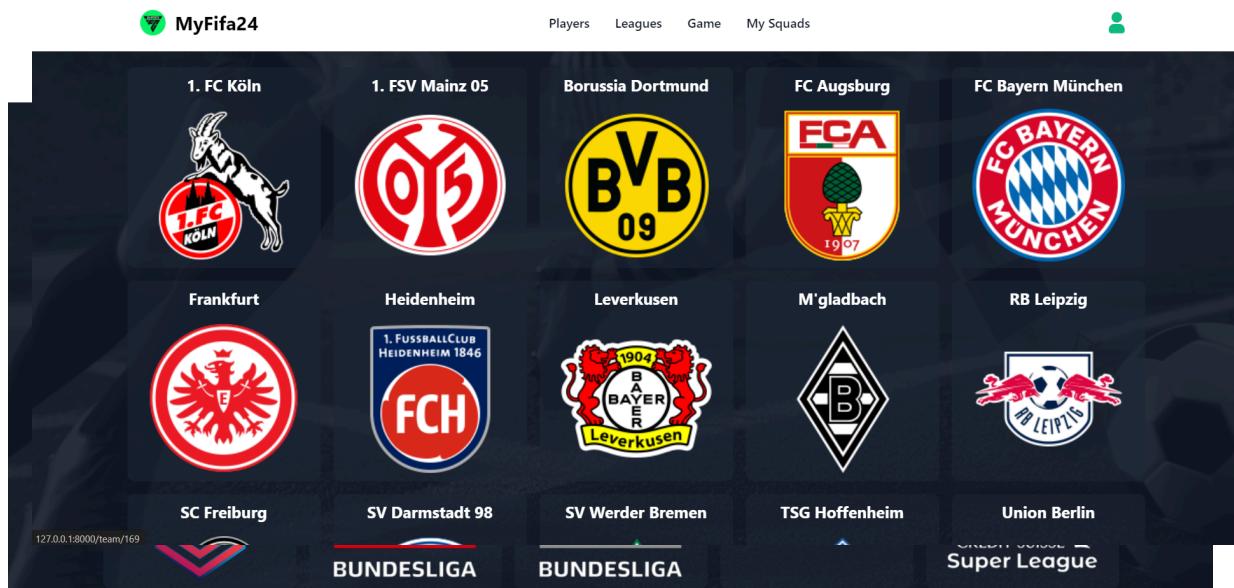
Como existem cerca de 17 mil jogadores na base de dados, cada query executada nesta página pode demorar largos segundos a ser executada. A fim de minimizar o tempo de espera todas as queries são guardadas durante 5 minutos em cache a fim de otimizar o

tempo de espera, evitando assim demorar muito tempo em páginas que já foram vistas recentemente.

2. Listar e Pesquisar Ligas, Equipas e Jogadores

Na tab “Leagues” é dado display de todas as ligas disponíveis na app, podendo estas ser filtradas por nome através da search bar.

Quando encontrarmos a liga que queremos saber mais informações basta carregar no logo que nos vai redirecionar para a lista de equipas pertencentes à liga correspondente.



Seguindo a lógica anterior, quando encontrarmos a equipa que desejamos basta carregar no seu logo que nos levará para a listagem de jogadores dessa mesma equipa.

A screenshot of the MyFifa24 mobile application interface, showing the player statistics for the Union Berlin team. The table lists 10 players, each with their name, nationality, OVR (Overall Rating), position, skill moves, week foot ability, work rate, and various performance metrics like PAC, SHO, PAS, DRI, DEF, and PHY. The players listed are Florian Kainz, Marvin Schwäbe, Timo Hübers, Dejan Ljubićić, Julian Chabot, Mark Uth, Benno Schmitz, Eric Martel, and Linton Maina.

Name	Nationality	OVR	Position	Skill Moves	Week Foot Ability	Work Rate	PAC	SHO	PAS	DRI	DEF	PHY
Florian Kainz	ATL	78	CAM	4★	5★	H\H	70	75	79	79	44	64
Marvin Schwäbe	GER	77	GK	1★	3★	M\H	79	74	76	81	47	73
Timo Hübers	GER	77	CB	2★	4★	M\H	76	40	56	59	79	75
Dejan Ljubićić	ATL	76	RM	3★	3★	M\H	76	70	71	75	72	78
Julian Chabot	GER	75	CB	2★	2★	M\H	49	28	43	48	77	80
Mark Uth	GER	75	CAM	3★	3★	M\L	67	77	74	76	40	62
Benno Schmitz	GER	74	RB	2★	4★	M\H	61	50	71	70	75	74
Eric Martel	GER	74	CDM	2★	3★	M\H	63	51	60	69	75	80
Linton Maina	GER	74	LM	3★	4★	M\H	86	62	68	76	29	55

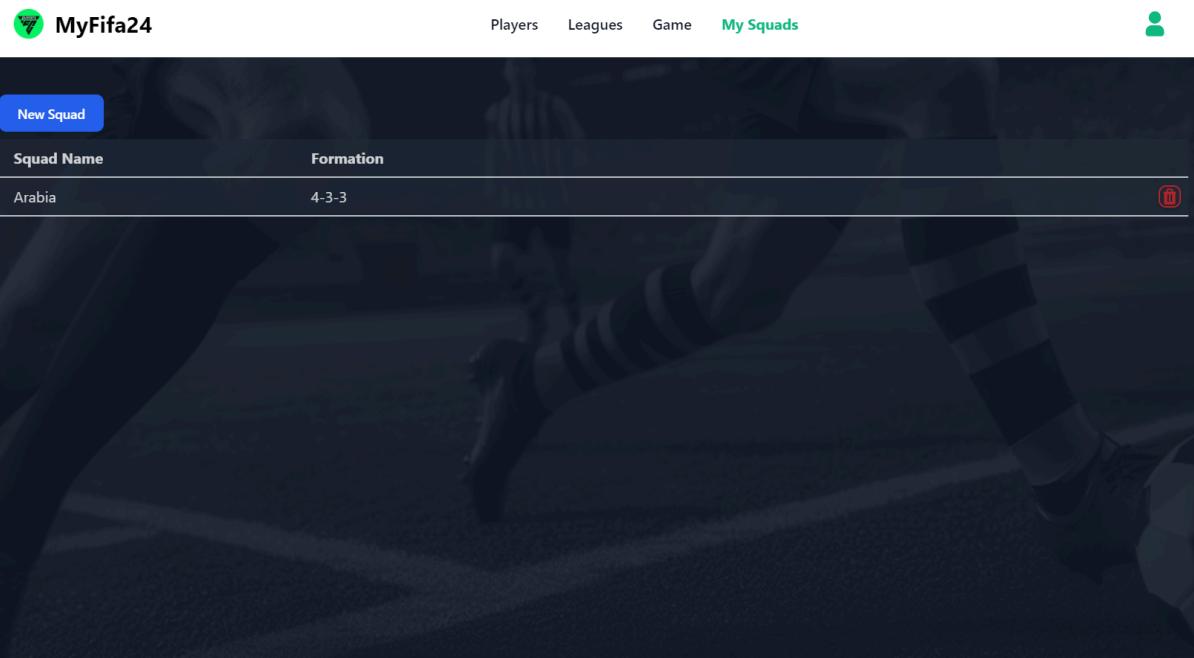
3. Jogar o jogo da adivinha

Na tab “Game” podemos aceder a um jogo que consiste em tentar adivinhar a estatística ou nome em falta ao jogador apresentado pela sua imagem. Em caso de acerto o valor inserido será colocado a verde, em caso de erro o valor correto aparecerá a vermelho. Para tentar novamente, deve carregar-se na tab Game para dar reset corretamente.

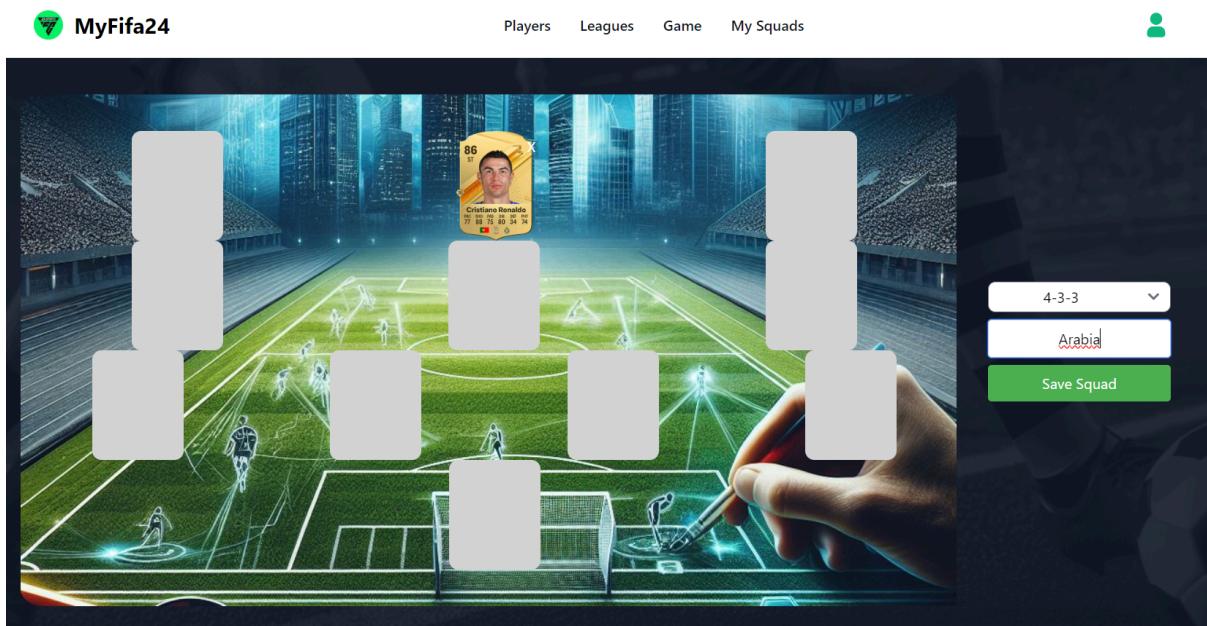
The screenshot shows a player profile for Jonathan Tah (CB). At the top left is the MyFifa24 logo. Below it is a portrait of Jonathan Tah. To his right is the Bayer Leverkusen logo, which features two red lions flanking a shield with the year 1904 and the word BAYER. To the right of the logo are the letters PAC, SHO, PAS, and their respective values: --, 37, and 59. Below these are the letters DRI, DEF, and PHY, with values 61, 81, and 85 respectively. On the far left, the letters OVR and the value 81 are displayed. To the right of the OVR value is the German flag. The background is dark with a subtle geometric pattern.

4. Criar squads com jogadores à escolha do utilizador

Na secção “My Squads” está disponível uma listagem das squads associada ao nosso utilizador e uma opção para criar uma equipa com os jogadores favoritos do usuário. É também possível eliminar squads previamente criadas através do botão vermelho à frente de cada squad.



Para realizar esta tarefa basta carregar no card correspondente à posição do jogador que pretende escolher e realizar uma pesquisa por nome do jogador. É também possível selecionar a formação, atualmente apenas com 4-3-3, mas também é possível escrever o nome que queremos dar à Squad. Quando terminada a criação basta carregar no botão de Save Squad que este será guardado e imediatamente listado na página mostrada anteriormente.



5. Executar Aplicação

Pré-requisitos:

1. Instale os requirements do ficheiro *requirements.txt* na pasta *dataset/* e correr o ficheiro *fifa.py* (pode demorar alguns minutos na primeira execução)
2. Instale os requirements do ficheiro *requirements.txt* na pasta *myFIFA24/*

Preparar o GraphDB:

1. Aceder ao ficheiro *GraphDB Desktop.cfg* na pasta *C:\Users\<User>\AppData\Local\GraphDB Desktop\app* e adicionar na última linha a opção “java-options=-Xmx8g”.
2. Inicialize o GraphDB. Caso não inicialize alterar o valor adicionado no passo anterior para um valor superior a 8.
3. Crie um novo repositório chamado *fifa24*
4. Importe o ficheiro *fifa.n3* da pasta *dataset/* (pode demorar alguns minutos)

Executar a aplicação:

1. No diretório *myFIFA24/*, execute o seguinte comando:

```
python manage.py migrate
```

2. Execute o seguinte comando para iniciar o servidor:

```
python3 manage.py runserver
```

3. Criar uma conta no sistema, fazer login e está pronto para explorar a aplicação. (A página de display de um squad não se encontra a funcionar corretamente)

6. Trabalho Futuro

Com o dataset escolhido temos bastantes dados para trabalhar e pensar em trabalho futuro.

Adicionar mais formações para o criador de squads seria uma boa adição para o website. Implementar uma representação gráfica dos dados também seria um foco de trabalho importante. Criar uma interação entre utilizadores do tipo Squad Battles.

7. Conclusão

Este trabalho proporcionou uma experiência valiosa na consolidação de conhecimentos adquiridos na Unidade Curricular, principalmente em relação a Django, RDF, GraphDB e SPARQL.

O trabalho permitiu ao grupo explorar em profundidade cada uma das tecnologias utilizadas, obtendo assim um conhecimento prático e teórico maior de cada uma delas. Consideramos que a escolha do tema do dataset foi bom no sentido que permitiu explorar a maioria das funcionalidades possíveis de implementar em SPARQL e consideramos que essas mesmas funcionalidades foram implementadas com sucesso.

Referências

- [1] “EA Sports FC 24 player ratings database - Electronic Arts.” EA, <https://www.ea.com/games/ea-sports-fc/ratings>. Accessed 16 April 2024.
- [2] *Drop API Service*, <https://drop-api.ea.com/api>. Accessed 16 April 2024.
- [3] “FUTBIN.” *FUTBIN: EA FC 24 Ultimate Team Prices, Squad Builder, Draft and Players Database*, <https://www.futbin.com>. Accessed 16 April 2024.