Követelményspecifikáció

Sport világeseményekhez jegyzőkönyvszerkesztő

Feladatkiírás

Olyan adatbeviteli-szerkesztő alkalmazás, amivel pl. a wikidata-hoz hasonló megjelenésű jelentések készíthetők sport világeseményekhez.

A fejlesztői csapat

A csapat tagjai:

csapattag neve	Neptun-kód	E-mail cím
Takács Levente	Py82c7	levi.takacs@gmail.com
Szabó András	Mwqci9	szabag@gmail.com

A csapatban dedikált szerepek kiosztását a csapat kis mérete miatt nem tartottuk fontosnak.

Részletes feladatleírás

A projekt során elsődlegesen a liga szerű bajnokságtípust támogatjuk, azaz mindenki mindenkivel játszik kétszer, egyet-egyet a két csapat otthonában.

Az adatokat több formában jelenítjük meg, attól függően, hogy az adott adathalmaznak melyik megjelenés áll a legjobban. A bajnokság aktuális állását, azaz melyik csapat hányadik, hány ponttal stb. egy táblázatban tekinthetjük meg. A lejátszott mérkőzéseket pedig egy mátrixos formában jelenítjük meg. A táblázatos megjelenítés mellett az egyszerű felsorolást is támogatjuk, például egy csapat adatainak megjelenítésére.

Az egyes megjelenítések között hiperhivatkozásokkal navigálhatuk, például a csapat nevére kattintva elérhetünk további adatokat a csapatokról (neve, stadionja, csapattagok, aktuális helyezés stb.) vagy egy mérkőzésre kattintva a mérkőzésről. (kik, hol játszották, ki nyert, góllövők stb.)

Az alkalmazás egy webes alkalmazás lesz, melyben a grafikus elemek egymás alatt helyezkednek el. Lesz egy nyitó oldal, ami a kiválasztott bajnokságról szerepelnek majd a főbb információk. Egy kattintható felületi elemre kattintva, pl csapatnévre kattintva

átnavigálhatunk a csapat oldalára, ahonnan lehetőségünk van visszatérni a főoldalra, vagy tovább böngészni.
Adatokat felvinni illetve módosítani, csak belépés után lehetséges, ezt a megfelelő formon tehetik meg az autentikált felhasználók.

Becsült:

Dokumentáció, belső kommunikáció, konzultációk stb.	20 óra
Hibajavítások és debuggolás	8 óra
Technikai feladatok (pl környezet), kutatás, előkészületek	20 óra
Felhasználói felület	12 óra
Adatelérés, wikidata	20 óra
Üzleti logika	20 óra

Rendszerterv

Sport világeseményekhez jegyzőkönyvszerkesztő

Feladatkiírás

Az szoftver egy webes alkalmazás, melyben liga szerű sportesemények, elsősorban az angol labdarúgó bajnokság (első osztály) adatait tekinthetjük át.
Az adatok a wikidatáról kerülnek lekérdezésre, és általában táblázatban jelennek meg. Hiperhivatkozások segítségével lehetőségünk van lefúrni az adatokban, hogy több részlethez jussunk.
Az alkalmazással lehetőségünk van adatokat rögzíteni a wikidata adatbázisába.

A rendszer főbb funkciói

Vázlatos, rövid felsorolás a rendszer főbb funkcióiról.

- Alapvető adatok lekérdezése a wikidatáról, és ezek megjelenítése táblázatos formában. (Ilyen táblázat például a csapatok alapvető információi, korábbi nyertesek stb.)
- A csapatok nevére kattintva további, részletesebb információt kaphatunk az egyesületről. (Például edző, alapítás, honlap, kép stb.)
- Adatok rögzítése a wikidata adatbázisába, és ezen adatok megjelenítése. (Egy meccs eredményét rögzíthetjük a jelenlegi szezonba, megadva a két csapatot és az eredményt)
- Az adatok komplex táblázatos megjelenítése. (Mátrixos megjelenítése a lejátszott meccseknek csapatokra bontva)
- Jogosultságok kezelése

Megvalósítás

Technológia

Az programot java (1.8) nyelven írtunk, és Spring boot keretrendszert használtuk. Azért választottuk ezt a technológiát, mert könnyen és gyorsan készíthetünk benne webalkalmazásokat, és rengeteg funkciót biztosít, melyek szabadon testre szabhatók. Pár ilyen funkciók, amiket mi használtunk:

- **Spring security** segítségével könnyen implementálható egy egyszerű authentikáció, felhasználók, jogosultságok stb.
- **Dependecy injection**
- **Thymeleaf**, egy szerver oldali template motor, melyet html kódgeneráláshoz használtunk.

Ezen felül a spring segít az MVC architektúra kialakításában, hibakezelésben (Pl: error page), beágyazott futtatókörnyezettel egyszerűvé teszi a futtatást, és jó fejlesztői támogatást biztosít, pl: reload, vagy live reload.

A függőségek kezelését mavennel oldottuk meg. Főbb függőségek:

- Spring boot (minden szükséges függősége)
- Logger (slf4j + logback)
- Wikidata toolkit
- JQuery + bootstrap

Architektúra

Az architektúra tipikus MVC architektúra, melyben a kliens oldal html és javascript, a szerver oldalt pedig java nyelven írtuk.

Megjelenítési réteg

A megjelenítés az egyik kulcs fontosságú feladat volt az alkalmazás készítésekor, mert nagyon fontos, hogy az adatokat átlátható, olvasható formában jelenítsük meg.
Az adatokat leginkább táblázatos formában jelenítjük meg. Léteznek egyszerű, és komplex, összetett táblázatok is, ahol az adatokon valamilyen transzformációt is végzünk, és úgy jelenítjük meg. Bizonyos oszlopok értékei hiperhivatkozások, melyeket további részleteket kaphatunk az adott cella értékéről.

A részletek egy form formájában jelennek meg, mert az áttekintésük így sokkal kényelmesebb. Itt láthatunk képeket is, hivatkozásokat stb.
Léteznek még további formok is, például a bejelentkező form, vagy az új meccs rögzítő form.

Az egyes oldalak html kódja mindig a szerver oldali adatok függvényében készül el. Ehhez a thymeleaf nevű könyvtárat használtuk, ami remekül integrálható a springgel. Így generáljuk az összes táblázatot, a formokat, vagy például a selectek lehetséges értékeit.

A szép megjelenítéshez a bootstrap css osztályait használtuk. (Pl: stripped table, buttons stb.)
A megjelenítési réteg készítésekor különösen ügyeltünk a responsiv felületre, hogy az alkalmazás telefonon is jól használható legyen.

Controllers

A konrollerek kötik össze a service-ből kapott adatokat a megjelenítési réteggel. Az alkalmazás 3 kontrollerel dolgozik:

- Authentication controller

A be és a kijelentkezést kezeli.

- Error kontroller

Hibakezelés, saját error page, bejelentkezési kísérlet meghiúsult.

- Default kontroller

Ez a konroller végzi a legtöbb feladatok: minden oldalra való HTTP GET kérést ez szolgál ki, összeállítja az adott oldalon szereplő táblázatok adatait egy strukturált formába, és átirányit az oldalra.
Az adatokat a kontrollerbe beinjektált wikidata service szolgáltatja.

Model

Az alkalmazásban több fajta model osztály is van, melyek azt a célt szolgálják, hogy a wikidatáról lekérdezett adatokat beparsolják, és így struktúrált formában kezelhetjük a lekérdezések eredményeit. A model osztályok több csoportba sorolhatók, funkciójuk szerint:

- **Wikidata objektum**

Ilyen osztály az *WikidataObject,* mely egy wikidata objektumot reprezentál. (Ez úgy képzelhető el hogy egy táblázat egy cellája.) Ennek mindig van egy értéke, és egy típusa. (Itt a típus a megjelenítés módját határozza meg, így akár cella pontosan beállítható, hogy az érték képként, linkként stb. jelenjen meg.)

- **Kliens oldali megjelenítésre formalizált objektumok, melyek a Wikidata objektumokat fogják össze**

Ilyen osztályok a *WikidataTableObject*, és a *WikidataFormObject*. Ezek az objektumok képezik a megfeleltetést a wikidata gráfadatbázisából kapott lekérdezések eredménye, és a mi alkalmazásunk kliens oldali megjelenítési formája között. Alapvetően egy title-ből (mely az adott tábla, vagy form címe) és egy mátrixból állnak. (Más értékeik is lehetnek pl: header) Ez a mátrix az előző pontban ismertetett Wikidata objektumokat tartalmazza, és ezek alapján renderelődnek a megfelelő html objektumok.
Ezek az objektumok képesek létrejönni egy wikidata lekérdezés eredményéből, és lehetővé teszik, hogy egységesen, struktúráltan kezeljük a lekérdezés eredményét. (Pl: beállítható egy oszlop típusa - pl link - vagy a fejlécek)

- **Adatrögzítésre formalizált objektumok**

Ilyen objektum például a *model,* melyek a másik irányú megfeleltetést teszik lehetővé, azaz a programban hozzuk létre őket, és a wikidata adatbázisába insertáljuk őket.

Service

A szolgáltatás réteg szolgáltatja az adatokat a controller rétegnek. A fő service a *wikidataService,* mely az éles wikidata gráfadatbázisából kérdezi le az adatokat, illetve szúrja be. A lekérdezések eredményét becsomagolja egy, a model-ben definiált objektumba, és így adja vissza.

Wikidata

A wikidata egy mindenki számára elérhető hatalmas gráfadatbázis (több mint 52 millió csomóponttal), melyből bárki kérdezhet le, vagy szúrhat be adatot. Fő célja hogy a wikipédia adatait hozza egy stuktúrált formába.
A wikidatához való kapcsolódásnak több módja létezik, rengeteg nyelvhez létezik jól dokumentált API-juk, rengeteg példával (Javához a wikidata-toolkit nevű könyvtárat használtuk), de sima lekérdezéseket is futtathatunk. A programban mi mind a két fajta technikát használtuk, ugyanis a lekérdezés sokkal egyszerűben megfogalmazhatók SPARQL nyelven, míg a beszúráshoz meg a wikidata-toolkit szolgáltatott egy egyszerű megoldást.

Lekérdezések

A wikidatából SPARQL nyelven kérdezhetünk le adatokat. A SPARQL egy RDF lekérdező nyelv, mellyel SQL-hez hasonló lekérdezéseket fogalmazhatunk meg a wikidata adatbázisából.
A lekérdezéseket a wikidata által ajánlott, Bordercloud SPARQL-JAVA nevű könyvtárral futtattuk. (https://github.com/BorderCloud/SPARQL-JAVA) Ennek használata nagyon egyszerű, azonban sajnos a maven nem ismeri, így ’kényszírteni’ kellett, hogy húzza be

lokálisan. További előnye hogy szintaktikai ellenőrzést is végez a SPARQL lekérdezéseken, hibakezelést is megoldja stb.
A lekérdezések eredményét mappekbe csomagolva kapjuk vissza, melyeket a megfelelő model osztályba csomagolunk. Továbbá a service további beállításokat is végez a becsomagolt objektumokon, például beállítja a fejléceket (ha táblázatról van szó), az egyes oszlopok típusát (Pl.: linkek) stb.
Az összes SPARQL lekérdezés elszeparálva a kódtól, a *SparqlQueries.java* fájlban található.
A hosszabb lekérdezések eredményei a service cachelni (mivel úgyse változik annyira), így az alkalmazás indulása után csak az első megnyitás tart sokáig.

Insertálás

A beszúrás SPARQL nyelven nem, vagy csak nehezen lehetséges (ennek több oka van, például a beszúráshoz autentikáció is szükséges, illetve nagyon kompex SPARQL kifejezések születnének) ezért ezt a Wikidata által szolgáltatott API-val oldjuk meg. Ennek segítségével létrehozhatunk propertiket, ezeket a propertiket a megfelelő értékükkel hozzáadhatjuk item-ekez. (Így például hozzá tudjuk kapcsolni az új item-ünket meglevőkhöz.) Az API biztosítja továbbá az autentikációt, és új itemek beszúrását a megfelelő label és description beállítása után. (Ennek a kombinációnak egyedinek kell lennie). Sokkal többet tud valóban, lehet vele módosítani, lekérdezni stb.

Rétegfüggetlen szolgáltatások

Biztonság

A program készítésekor különös hangsúlyt fektettünk a biztonságra, ezért bizonyos funkciókat bejelentkezéshez kötöttünk. Többféle jogosultság csoport létezik:

- Bejelentkezés nélkül lehetősége van a felhasználónak megtekinteni a lekérdezések eredményeit, azaz láthatja a home menüpontban az összes táblát.
- Egyszerű user jogosultsággal a felhasználó további részleteket tudhat meg egy csapatról a nevére kattintva, vagy a teams menüpontban. (Illetve természetesen ő is látja azt amit bejelentkezés nélkül)
- Admin joggal az előző kettő mellet, a felhasználónak lehetősége van új meccsek eredményét rögzíteni az éles wikidata adatbázisba. Ez egy kritikus pont, ezért kötöttük admin joghoz.

Hibakezelés

A hibák kezelését egy külön kontroller végzi, így egy hiba esetén egy saját, testreszabott html oldal jelenik meg, ahol további részleteket kaphatunk a hibáról.
Ezzel a módszerrel lehetőség van kezelni a sikertelen bejelentkezéseket is, és így olvasható visszajelzést adhatunk a felhasználónak.

Megjelenített adatok

Az alkalmazásban Premier League bajnoksághoz kapcsolódó adatokat jelenítünk meg, több módon.

Egyszerű táblázatos megjelenítés

Egyszerű táblázatok a wikidatából lekérdezett adatokon parsolásokat, formázásokat végeznek. Ezek jól olvasható, áttekinthető megjelenítést biztosítanak. Ezek a táblázatok a kezdőoldalon egymás alatt jelennek meg, és bejelentkezés nélkül megtekinthetők. Ilyen táblázatok:

Premier league teams

A premier league-ban játszó csapatokat tartalmazza. A megjelenített adatok: név, teljes név, székhely, stadion.

Táblázat

Premier league teams

Team	Full name	Headquarter	Home venue
Watford F.C.	Watford Football Club	Watford	Vicarage Road
Everton F.C.	Everton Football Club	Liverpool	Goodison Park
Chelsea F.C.	Chelsea Football Club	London	Stamford Bridge stadium
Arsenal F.C.	Arsenal Football Club	London	Emirates Stadium
Manchester United F.C.	Manchester United Football Club	Manchester	Old Trafford
Middlesbrough F.C.	Middlesbrough Football Club	Middlesbrough	Riverside Stadium
Cardiff City F.C.	Cardiff City Football Club	Cardiff	Cardiff City Stadium
Newcastle United F.C.	Newcastle United Football Club	Newcastle upon Tyne	St James' Park

SPARQL:

```
SELECT DISTINCT ?teamLabel ?hivatalos_n_v ?sz_khelyLabel ?hazai_stadion__sz_khely_Label WHERE {
  ?team (wdt:P118/wdt:P279*) wd:Q9448.
  ?team (wdt:P31/wdt:P279*) wd:Q476028.
  ?team wdt:P17 wd:Q145.
  OPTIONAL { ?team wdt:P571 ?inception. }
  OPTIONAL { ?team wdt:P1448 ?hivatalos_n_v. }
  OPTIONAL { ?team wdt:P159 ?sz_khely. }
  OPTIONAL { ?team wdt:P115 ?hazai_stadion__sz_khely_. }
  SERVICE wikibase:label { bd:serviceParam wikibase:language "[AUTO_LANGUAGE],en". }
}
ORDER BY ?Name
```

Champions

A korábbi bajnokságokról tudhatunk meg információt: ki nyerte, illetve hány meccset játszottak az adott szezonban. Akár 1992-93-as bajnokságról is kaphatunk információt.

Táblázat

Champions

Championship	Winner	Games played
1999–2000 FA Premier League	Manchester United F.C.	380
1997–98 FA Premier League	Arsenal F.C.	380
2008–09 Premier League	Manchester United F.C.	380
2010–11 Premier League	Manchester United F.C.	380
2007–08 Premier League	Manchester United F.C.	380
2009–10 Premier League	Chelsea F.C.	380
2006–07 FA Premier League	Manchester United F.C.	380
1994–95 FA Premier League	Blackburn Rovers F.C.	462
2005–06 FA Premier League	Chelsea F.C.	380

SPARQL

```
SELECT DISTINCT ?leagueLabel ?nyertesLabel ?merkozesek_szama WHERE {
 ?league wdt:P3450 wd:Q9448.
 ?league wdt:P1346 ?nyertes.
 SERVICE wikibase:label { bd:serviceParam wikibase:language "[AUTO_LANGUAGE],en". }
 OPTIONAL { ?league wdt:P1350 ?merkozesek_szama. }
}
ORDER BY ?korszakLabel
```

Match results

Mérkőzések eredménye az 2018-19es szezonban. Ezek az adatok nem álltak rendelkezésre a wikidata adatbázisában, úgyhogy ezeket mi rögzítettük a hivatalos eredményeknek megfelelően. (https://www.premierleague.com/results) További meccsek rögzítésére is van lehetőség a *Create* menüpontban.

Table

Match results

Match	Result	Winner	Participating teams
Everton F.C. vs Liverpool F.C.	1-1	draw	Everton F.C.,Liverpool F.C.
Arsenal F.C. vs Watford F.C.	2-0	Arsenal F.C.	Watford F.C.,Arsenal F.C.
Everton F.C. vs Crystal Palace F.C.	1-3	Crystal Palace F.C.	Everton F.C.,Crystal Palace F.C.
Chelsea F.C. vs Arsenal F.C.	3-2	Chelsea F.C.	Chelsea F.C.,Arsenal F.C.
Chelsea F.C. vs Tottenham Hotspur F.C.	0-0	draw	Chelsea F.C.,Tottenham Hotspur F.C.
Arsenal F.C. vs Liverpool F.C.	1-1	draw	Arsenal F.C.,Liverpool F.C.
Arsenal F.C. vs Leicester City F.C.	3-1	Arsenal F.C.	Arsenal F.C.,Leicester City F.C.
Newcastle United F.C. vs Arsenal F.C.	1-2	Arsenal F.C.	Arsenal F.C.,Newcastle United F.C.

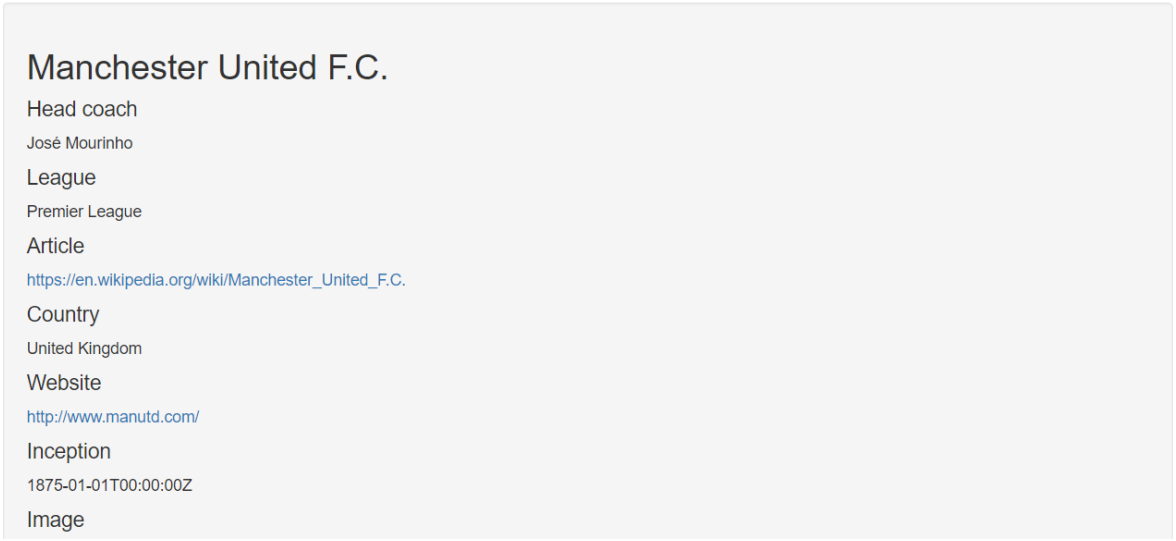
SPARQL

```
SELECT ?matchLabel ?matchDescription ?nyertesLabel (GROUP_CONCAT(DISTINCT ?teamName;separator=",") AS ?teams) WHERE {
    ?match wdt:P31 wd:Q16466010.
    ?match wdt:P361 wd:Q52394608.
    ?match wdt:P710 ?participating.
    OPTIONAL { ?participating rdfs:label ?teamName.
               FILTER(LANG(?teamName) = "en").}
    OPTIONAL { ?match wdt:P1346 ?nyertes. }
    SERVICE wikibase:label { bd:serviceParam wikibase:language "[AUTO_LANGUAGE],en". }
  }
  GROUP BY ?matchLabel ?matchDescription ?nyertesLabel
  ORDER BY ?match
```

Részletek

A „részletek” nézetbe vagy egy csapat nevére, vagy pedig a navbar-ban a teams menüpontra kattintva juthatunk. Utóbbi esetben az összes csapathoz tartozó részletes információt megkapjuk. (Ez elsőre kicsit lassú lehet, mivel sok adatról van szó, de az eredményt elcacheljük, így másodjára pillanatok alatt betöltődik.) Ez a funkció bejelentkezés után érhető el, egyszerű user, vagy admin jogosultsággal.
A részletek nézetben egy csapat részletes adatait láthatjuk (ami rendelkezésre áll a wikidata adatbázisában) egy formon. (Például alapítás éve, ország, link a wikipedia oldlhoz, kép, becenév, edző stb.)
Itt is ügyeltünk az átlátható megjelenítésre. (Ezért választottuk a form-os megjelenítést)

Form



SPARQL

```
SELECT DISTINCT ?article ?countryLabel ?imageLabel ?inceptionLabel ?nicknameLabel ?headcoachLabel ?leagueLabel ?homevenueLabel ?officalwebsiteLabel
WHERE
  { ?article  schema:about ?item ;
              schema:inLanguage "en" ;
              schema:isPartOf    <https://en.wikipedia.org/> .
    OPTIONAL { ?item wdt:P17 ?country. }
    OPTIONAL { ?item wdt:P18 ?image. }
    OPTIONAL { ?item wdt:P571 ?inception. }
    OPTIONAL { ?item wdt:P1449 ?nickname. }
    OPTIONAL { ?item wdt:P286 ?headcoach. }
    OPTIONAL { ?item wdt:P118 ?league. }
    OPTIONAL { ?item wdt:P115 ?homevenue. }
    OPTIONAL { ?item wdt:P856 ?officalwebsite. }
    FILTER ( ?item = <%s> )
    SERVICE wikibase:label
      { bd:serviceParam
                  wikibase:language  "en"
      }
  }
```

Komplex táblázatok

Minden sporttípusnak és minden sporteseményhez köthető lebonyolítási/rendezési formának megvannak a saját speciális nézetei, amelyekben az adatok és eredmények a legszemléletesebben, legintuitívabban megjeleníthetők.
Az általunk kiválasztott bajnokság a Premier League volt; ennek tipikus jellemzői, hogy a csapatok oda-vissza vágókat játszanak egymással, és minden csapat játszik mindenkivel (ellentétben mondjuk az NFL-lel). Az állások továbbá egy egyszerű táblázaton követhetőek nyomon, a legtöbb pontot szerzett csapat van az élen, pontegyenlőség esetén további kritériumok vannak, de ezekkel nem foglalkoztunk részletesebben.

2018-19 Premier league matches

Az egyik leggyakrabban használt táblázat a mátrix szerű megjelenítési forma. Az oszlopok a hazai meccseket jelölik, a sorok az idegenbeli meccseket. A sorok és az oszlopok is ugyanolyan sorrendben ugyanazokat a csapatokat tartalmazzák. Ha tudni akarjuk egy csapat eredményeit csak egyszerűen leolvassuk az adott csapat meccseit sorfolytonosan. A párharcokat is jól lehet követni, hiszen egyik mérkőzés az egyik csapat otthonában volt, a másik pedig a másik csapat otthonában, és a metszéspontokból leolvashatóak ezek az eredmények.
Az végeredmény formátuma az Európában használatos konvenciókat követi, elől a hazai csapat által szerzett gólok, majd a kötőjel után a vendégcsapat góljai.

Table

2018-19 Premier league matches

Home \ Away	BOU	ARS	BRI	BUR	CAR	CHE	CRY	EVE	LEI	LIV	MCI	MUN	NEW	TOT	WAT	WHU	WOL
AFC Bournemouth	-																
Arsenal F.C.		-				2-3			3-1	1-2		0-2	2-1		2-0		
Brighton & Hove Albion F.C.			-										1-2				
Burnley F.C.				-					0-0								
Cardiff City F.C.					-						0-5			0-1			
Chelsea F.C.		3-2				-								0-0			
Crystal Palace F.C.							-	3-1									
Everton F.C.							1-3	-		1-1							
Leicester City F.C.		1-3		0-0					-								
Liverpool F.C.		2-1						1-1		-							
Manchester City F.C.					5-0						-						
Manchester United F.C.		2-0										-		1-1			
Newcastle United F.C.		1-2	2-1										-				
Tottenham Hotspur F.C.					1-0	0-0						1-1		-			
Watford F.C.		0-2													-		
West Ham United F.C.																-	
Wolverhampton Wanderers F.C.																	-

2018-19 Premier league standing

A 2018-19-es szezon jelenlegi rangsora. A táblázat tartalmazza, a csapatok neveit, lejátszott mérkőzéseinek a számát, győzelmek, vereségek, döntetlenek számát, pontjaikat (e szerint rendezi sorba a csapatokat) és egy leírást. Ha felveszünk egy meccsek akkor amint a meccs bekerül a korábban említett „Match results” táblába, azzal egy időben ebben a táblában is frissülnek az értékek, és a helyezések. A frissítés java oldalon történik meg a meccsek feldolgozásával.
A csapatok természetesen ezen az oldalon is kattinthatók, a „részletek” nézetre tudunk navigálni az interakcióval.

Table

2018-19 Premier league standing

Place	Team	Match	W	D	L	Points	Description
1	Manchester United F.C.	4	3	1	0	10	Bajnokok Ligája-csoportkör
2	Arsenal F.C.	8	3	1	4	10	Bajnokok Ligája-csoportkör
3	Tottenham Hotspur F.C.	4	1	2	1	5	Bajnokok Ligája-csoportkör
4	Liverpool F.C.	3	1	2	0	5	Bajnokok Ligája-csoportkör
5	Chelsea F.C.	2	1	1	0	4	Európa-liga-csoportkör
6	Newcastle United F.C.	2	1	0	1	3	
7	Manchester City F.C.	1	1	0	0	3	
8	Crystal Palace F.C.	1	1	0	0	3	
9	Leicester City F.C.	2	0	1	1	1	

SPARQL

Ezek a táblázatok több lekérdezést is használnak, és ezeknek az adatait kombinálják, és transzformálják.

- 2018-19 es csapatok lekérdezés:

```
SELECT DISTINCT ?teamName WHERE {
  ?team wdt:P3450 wd:Q9448.
  ?team wdt:P31 wd:Q27020041.
  ?team wdt:P17 wd:Q145.
  ?team wdt:P2348 wd:Q52394608.
  ?team wdt:P1923 ?participating.
  OPTIONAL { ?participating rdfs:label ?teamName.
             FILTER(LANG(?teamName) = "en").}
  SERVICE wikibase:label { bd:serviceParam wikibase:language "[AUTO_LANGUAGE],en". }
}
ORDER BY ?teamName
```

- Bajnokság állása

```
SELECT DISTINCT (1 as ?place) ?teamName (0 as ?matchPlayed) (0 as ?win) (0 as ?draw) (0 as ?lose) (0 as ?points) ('' as ?description) WHERE {
  ?team wdt:P3450 wd:Q9448.
  ?team wdt:P31 wd:Q27020041.
  ?team wdt:P17 wd:Q145.
  ?team wdt:P2348 wd:Q52394608.
  ?team wdt:P1923 ?participating.
  OPTIONAL { ?participating rdfs:label ?teamName.
             FILTER(LANG(?teamName) = "en").}
  SERVICE wikibase:label { bd:serviceParam wikibase:language "[AUTO_LANGUAGE],en". }
}
ORDER BY ?teamName
```

Adat rögzítés

Lehetőségünk van új adatokat rögzíteni az éles wikidata adatbázisba. Mivel ez a funkció a legkritikusabb, ezért admin joggal kell rendelkezni a funkció használatához, egyszerű user-ek nem használhatják.
Az alkalmazással a jelenlegi szezonhoz (2018-19) es bajnoksághoz rögzíthetünk új mecseket. Ezt a create menüpontban tehetjük meg bejelentkezés után, kitöltve az alábbi formot.

Megjegyzés: Az adat rögzítése egyből megtörténik, azonban amíg ténylegesen megjelenik egy lekérdezésben az eltarthat 5-6 percig (ez a wikidata sajátossága, nyilván valamilyen indexelés miatt), ezért csak ez után fog megjelenni a táblázatokban.

Telepítési leírás

A készítéskor nagy hangsúlyt fektettünk arra, hogy az alkalmazásnak nem csak a használata, hanem a telepítése is egyszerű legyen. (E miatt is választottuk a Spring Boot keretrendszert, illetve a maven-t)

1. A program letöltése githubról: https://github.com/banda13/WikiSport
2. A root mappában található egy build.bat windwos file, ezt kell futtatni. (Ennek futtatása elsőre eltarthat egy darabig, mert az alkalmazás le fog tölteni minden szükséges függőséget, azonban másodjára már gyorsan indul)

A futtatáshoz 1.8-as java, illetve 3.5-ös maven szükséges.

Belépési adatok:

	felhasználónév	jelszó
USER	test	asd123
ADMIN	admin	admin

Összefoglalás

A feladat során sikerül elkészíteni a szoftver, külön hangsúlyt fektettünk arra, hogy a specifikációban meghatározott összes funkciót elkészítsük.
Az alkalmazással egy átfogó képet kaphatunk a Permier league jelenlegi állásáról, a csapatokról, korábbi eredményekről. Az adatok (ahol rendelkezésre állnak) mindig naprakészek.
A biztonság kellő módon megoldott, a meghatározott jogosultság csoportoknak jól definiált a scope-uk.
A megjelenítés egyértelmű, és áttekinthető, az alkalmazás használata egyszerű, és mivel webes alkalmazás, könnyen hozzáférhető bárki számára, mind deskop gépen, mind mobilon.
A specifikációban definiált követelményeket hiánytalanul sikerült teljesíteni, és a becsült időtartamokat is sikerült tartani körülbelül.

Továbbfejlesztési lehetőséget

Ugyan a kíván funkcionalitást sikerült elérni, a program még rengeteg irányba fejleszthető tovább.

- További adatokat megjeleníthetne a Premier league-hoz kapcsolódóan. (Habár ebbe az irányba korlátosan, ugyanis a wikidatán tárolt adatok többségét lefedtük)
- Több bajnokság támogatása. (Ne csak az angol első osztályról, hanem minél több bajnokságról kapjuk információt)
- Új bajnokság létrehozása, és nyomon követése

A programot felkészítettük a továbbfejlesztésre, úgy alakítottuk ki az architektúrát, az objektumokat, hogy nyitottak legyenek a bővítésre. (Open closed principle)
Összefoglalva érdekes volt egy ilyen alkalmazás készítése, mivel új volt számunkra a spring framework (Legalább is nem volt tapasztalatunk benne), a wikidata által megismerkedhettünk a gráfadatbázisok sajátosságaival, a SPARQL lekérdező nyelvvel, és egy architektúra készítését is gyakorolhattuk.

Tartalom jegyzék