Univerzitet Donja Gorica

Fakultet za informacione sisteme i tehnologije

Podgorica

Prikaz rezultata analize podataka na veb-u pomoću Python programskog jezika sa konkretnim primjerom

Diplomski rad

Student: Balša Dogandžić

Broj dosijea: 20/124i

Podgorica, septembar 2023. godine

Univerzitet Donja Gorica

Fakultet za informacione sisteme i tehnologije

Podgorica

Prikaz rezultata analize podataka na veb-u pomoću Python programskog jezika sa konkretnim primjerom

Diplomski rad

Mentor: mr Stevan Čakić

Student: Balša Dogandžić

Broj dosijea: 20/124i

Podgorica, septembar 2023. godine

APSTRAKT

Python je jedan od najpopularnijih programskih jezika zbog svoje jednostavnosti, ali i zbog njegovih ogromnih mogućnosti. U ovom radu je opisan sistem koji je napravljen upravo pomoću ovog programskog jezika. Pomenuti sistem je veb sajt napravljen pomoću *Django* paketa koji prikazuje rezultate analize i grafičke vizualizacije podataka realizovane pomoću paketa *Pandas*, *Matplotlib* i *Seaborn* na interfejsu pretraživača. U poglavlju diskusije, nakon opisa sistema će biti iznesene dobre i loše strane sistema, na koji način se ovaj sistem može popeti na viši nivo, ali i koji je benefit ovakvog sistema.

Ključne riječi: *Python*, Veb, Podaci, Analiza, Vizualizacija, *Django*, *Pandas*.

ABSTRACT

Python is one of the most popular programming language because of it's simplicity, but also because of it's great versatility. A system described in this thesis is made with previously mentioned programming language. System is a web application made with Django framework which displays data analysis results and graphic visualizations of data with Pandas, Matplotlib and Seaborn packages on browser's interface. In the chapter of discussion, after the system description, the good and the bad sides of the system will be presented, how can this system be better, and what is the benefit of such a system.

Key words: Python, Web, Data, Analysis, Visualization, Django, Pandas.

SADRŽAJ

Lista slika

1. UVOD

Podaci su svuda i oni su osnova svih sistema koji olakšavaju svakodnevni život ljudi širom svijeta. Kroz podatke je moguće uočiti pojave, identifikovati potencijalne probleme, ali i donijeti odgovarajuće odluke u biznisu ili drugim sferama života. Podataka je iz godine u godinu sve više, prema istraživačima iz CISCO organizacije: protok podataka kroz internet 2022 godine se procjenjuje na 4.8 zetabajta, što je oko $4.8 * 10^{21}$ bajtova.[1] Rastom protoka podataka raste i potreba da se ovi podaci analiziraju, i da se iz njih stvori neka nova vrijednost. Osim što je tehnologija u velikoj mjeri i "krivac" za generisanje ovolike količine podataka, ona predstavlja i rješenje kako da se ovi podaci predstave na razumljiv način.

Postoji veliki broj softverskih rješenja bilo to komercijalnih ili besplatnih rješenja otvorenog koda za analiziranje podataka, neki od njih su: *MS Excel*, *R* programski jezik, *Matlab*, *Scala*, *Python* i mnogi drugi. *Python* i *MS Excel* su svakako dva najkorišćenija i najpoznatija alata za obradu i manipulaciju nad podacima. Prednost *Python*-a u odnosu na *Excel* je ta što je *Python* programski jezik otvorenog koda i kao programski jezik šire namjene nije ograničen samo na rad sa podacima. Sa *Python* programskim jezikom je moguće kreirati veb aplikacije (*Django*, *Flask*), *desktop* aplikacije (*Tkinter*), ali i skripte različitih namjena pomoću ogromnog broja paketa. Nedostatak *Python*-a u odnosu na *MS Excel* i ostale komercijalne softvere je taj što za korišćenje *Python*-a korisnik mora posjedovati programersko znanje, dok komercijalni alati korisniku pružaju grafički interfejs koji mu omogućava lakše korišćenje softvera i bolje korisničko iskustvo. Ali i pored tih nedostataka *Python* sa svojim paketima za analizu podataka (*NumPy*, *Pandas*, *Matplotlib*...) dobija sve veću popularnost zbog svoje jednostavnosti, brzine i potencijala. Između ostalog je i to razlog zašto je upravo ovaj programski jezik tema ovog rada. U narednim poglavljima ovog rada će biti opisan praktični dio projekta za čiju realizaciju su korišćeni *Python* paketi za analizu

[1] Barnett, T.; Jain, S. (2018). *Cisco visual networking index (vni) complete forecast update, 2017–2022*. Americas/EMEAR Cisco Knowledge Network (CKN) Presentation, strana br. 8

podataka koji su prethodno pomenuti, ali i njegov radni okvir za izradu dinamičnih veb sajtova pod imenom *Django*.

1.1 Ideja rada i cilj rada

Ideja rada je pronalaženje odgovarajućeg skupa podataka nad kojim će se vršiti manipulacija, analiza, vizualizacija podataka i na kraju donošenje zaključaka na osovu rezultata. Sređeni skup podataka bi se zatim koristio kao izvor podataka za kreiranje dinamične veb aplikacije na kojoj bi se prikazivali rezultati analize, statističke vrijednosti i vizualne reprezentacije podataka u vidu grafika/dijagrama. Cilj rada je da se sirovi podaci iz skupa podataka prikažu na interfejsu veb aplikacije. Ova aplikacija bi omogućila korisniku da vidi samo one podatke koji su njemu interesantni i značajni za donošenje zaključaka.

1.2 Očekivanja od rada

Očekivanja su da praktični dio ovog rada predstavlja spoj dvije discipline u IT industriji, i to razvoja veb aplikacija i nauke o podacima. A od ukupnog istraživačkog rada (teorijski i praktični dio) se očekuje da donese novinu u ove dvije oblasti, tj. da pokrene dalji razvoj ideja na ovu temu.

1.3 Tema u okviru mreže međuzavisnosti

Veb aplikacije postaju sve prisutniji oblik aplikacija iz razloga što su najpristupačnije za korisnike. Korisnik ne mora da brine o ažuriranjima i memoriji na računaru kao kod *desktop* aplikacija. Upravo zbog svoje jednostavnosti i pristupačnosti, veb predstavlja idealnu platformu za razvoj ovakvih, i sličnih aplikacija. Statistička analiza ima veliki potencijal za dalji razvoj jer podaci i informacije predstavljaju najvredniji resurs u savremenom svijetu.

2. Metodologija

U ovom poglavlju je naveden materijal i metodologija korišćena za izradu praktičnog dijela projekta. Praktični dio projekta je kao što je ranije navedeno veb aplikacija koja prikazuje rezultate analize podataka i vizualne reprezentacije podataka (grafike). Ovo poglavlje je podijeljeno na dva potpoglavlja, i to prvo potpoglavlje u kojem su opisane biblioteke korišćene u radu, i drugo u kome je opisan skup podataka koji je korišćen.

2.1 Softverski paketi

U ovom poglavlju su detaljno opisani paketi koji su korišćeni za potrebe realizacije praktičnog dijela ovog rada. Paketi koji su korišćeni su:

1. ***NumPy*** – je izuzetno brz i jednostavan paket za manipulaciju nad višedimenzionalnim nizovima, vektorima i matricama. „*NumPy* kombinuje moć programiranja nizova, performanse *C*-a, čitljivost i svestranost *Python*-a u dobro testiranoj, dokumentovanoj i zreloj biblioteci za korišćenje“.[2] Kao što je navedeno *NumPy* ima brzo izvršavanje poput *C* programskog jezika koji je po tome poznat. Samim tim nije ni čudno što je većina biblioteka koje slijede napravljeno upravo sa *NumPy* paketom u osnovi. Ovaj paket nije direktno korišćen u značajnoj mjeri kao ostali paketi, ali jeste indirektno kao njihov sastavni dio.
2. ***Pandas*** – je jednostavan i popularan *Python* softverski paket koji se koristi u analizi i manipulaciji nad podacima. *Pandas* uvodi dvije vrste novih objekata, i to *DataFrame* objekte kao dvodimenzionalne, i *Series* objekte kao jednodimenzionalne strukture. Kao što navodi McKinney: *DataFrame* objekat se sastoji od većeg broja *Series* objekata, pa se može reći da su oni u odnosu tabela i kolona.[3] *Pandas* je u praktičnom dijelu korišćen za

[2] Harris, C. R.; Millman, K. J. (2020). *Array programming with NumPy*. Nature, strana br. 361

[3] McKinney, W. (2010). *Data structures for statistical computing in python*. In Proceedings of the 9th Python in Science Conference, strana br. 60

čišćenje, manipulisanje i analiziranje podataka iz skupa podataka, koji je u vidu *CSV* fajla u dijelu vezanom za analizu podataka. Ali je takođe korišćen i u razvoju funkcionalnosti veb aplikacije.

3. *Matplotlib* – je paket koji se koristi za vizualizaciju podataka. Sa ovim paketom je moguće kreirati veliki broj grafika (pita dijagrami, dijagrami sa stubićima itd.). Grafici se mogu konstruisati iz *Python* listi, *NumPy* nizovova, ali i iz prethodno pomenutih *Pandas* objekata (*DataFrame*, *Series*). Ovaj paket je u radu korišćen za vizualni prikaz podataka i u dijelu analize, a takođe i na veb aplikaciji.
4. *Seaborn* – je takođe paket za vizualizaciju podataka. Razlika između *Matplotlib*-a i *Seaborn*-a je kako navodi Michael L. Waskom u tome što: *Matplotlib* predstavlja paket nižeg nivoa, pa je sa *Seaborn* paketom mnogo jednostavnije predstaviti kompleksne statističke dijagrame nego sa *Matplotlib*-om.[4] *Seaborn* je u projektu korišćen za prikazivanje atraktivnih i kompleksnijih dijagrama kako u analizi, tako i u izradi veb sajta.
5. *Django* – je *Python* radni okvir za kreiranje takozvanih "*fullstack*" veb aplikacija, ili *API* servisa korišćenjem *Django REST Framework*-a. Kada se priča o razvoju veb aplikacija sa *Python*-om obično je *Django* prvi koji se pomene zajedno sa *Flask*-om i *FastAPI*-jem, što dokazuje njegovu popularnost među programerima. Ono što *Django* izdvaja od dva prethodno pomenuta paketa je to što oslobađa programera brige o rutiranju stranica, autentifikaciji korisnika, povezivanju sa bazom podataka, pisanju *SQL* upita i mnogih drugih. To je iz razloga što su sve ove funkcionalnosti već uključene, ili ih je vrlo lako implementirati. Arhitektura aplikacije takođe nije briga programera jer kreiranjem *Django* projekta korisnik dobija jednostavnu *Django* aplikaciju sa definisanom arhitekturom. Kao što kaže William S. Vincent: za razliku od *MVC* (*Model-View-Controller*) arhitekture, *Django* primjenjuje *MVTU* (*Model-View-Template-URL*) arhitekturu, u kojoj je *Model* - reprezentacija podataka, *View* - logika veb stranice, *Template* – struktura veb stranice, *URL* – na kojoj adresi *View* obavlja svoju funkciju.[5] *Django* je u praktičnom dijelu služio kao

[4] Waskom, M. L. (2021). *Seaborn: statistical data visualization*. Journal of Open Source Software, 6(60), 3021. strana br. 1

[5] Vincent, W. S. (2022). *Django for Beginners: Build websites with Python and Django*. WelcomeToCode, strana br. 19

osnova veb aplikacije, u njegovim *view* funkcijama se obavljala analiza sa *Pandas*-om i vizualizacija *Matplotlib*-om.

2.2 Izvor podataka

Procesi analize i istraživanja imaju neke zajedničke korake u procesu, prvi i jako važan korak je pronalaženje i sakupljanje relevantnih podataka. Podaci koji se prikupljaju moraju biti kako je navedeno ranije relelevantni, ali i kvalitetni, sveobuhvatni, tačni i naravno da ih ima što više.

Izvor podataka koji je korišćen za potrebe izrade praktičnog dijela rada je *online dataset* sa *Kaggle* platforme. *Kaggle* je internet platforma koja predstavlja veliki izvor podataka iz različitih oblasti, ovu platformu čak i nazivaju društvenom mrežom za analitičare. Platforma omogućava korisnicima da preuzmu ogroman broj *dataset*-ova, ali i da ih direktno obrađuju i analiziraju kroz *Kaggle notebook*. *Dataset*-ovi i *notebook*-ovi su javno dostupni pa korisnici mogu imati uvid kako su drugi korisnici analizirali *dataset*, kakve su oni rezultate dobili i sl. *Dataset* korišćen u ovom radu se nalazi na sledećoj internet adresi:

https://www.kaggle.com/datasets/azminetoushikwasi/ucl-202122-uefa-champions-league

U pitanju je arhiva *CSV* fajlova koja sadrži podatke o igračima na popularnom fudbalskom takmičenju UEFA Liga šampiona, sezona 2021/2022. Liga šampiona se održava svake godine, i predstavlja jedno od najispraćenijih fudbalskim takmičenjima zajedno sa Svjetskim i Evropskim prvenstvom. Fudbal je pogodan za analizu iz nekoliko razloga. Prvi je taj što je fudbal jedan od najpopularnijih, ako ne i najpopularniji sport na svijetu, i kao takav generiše ogromne profite i gledanost. Osim profita i gledanosti fudbal generiše i ogroman broj podataka. Svakodnevno se odigra veliki broj profesionalnih mečeva, nakon kojih se rezultati klubova i igrača sakupljaju, čuvaju i analiziraju. Analiziranje podataka doprinosi boljim odlukama selektora i trenera timova, tačnijem predviđanju rezultata utakmice, proglašavanjem najboljeg igrača, tima itd. Cilj ovog istraživanja je da se analizom iz ove arhive podataka upravo donesu takvi i slični zaključci.

Arhiva sadrži 8 *CSV* fajlova od kojih su neki od njih: fajl sa podacima o golovima, napadačima, golmanima, disciplinom na terenu itd. U zavisnosti od tipa analize se odabira koji od ovih fajlova je prikladan za tu analizu. Npr. ukoliko se analizaju odbrambene sposobnosti igrača onda se bira fajl koji sadrži te podatke. Zajedničko za sve fajlove je to što se nijedan igrač ne pojavljuje više puta unutar jednog fajla. Međutim igrač se može pojaviti u više fajlova pod istim imenom. Ovaj podatak je bitan jer je onda moguće spojiti sve ove fajlove u jedan fajl koji sadrži podatke za sve igrače iz *dataset*-a. Na sledećem linku se nalazi *Kaggle notebook* u kojem se koristi *pandasql* biblioteka da bi se svi fajlovi spojili u jedan:

https://www.kaggle.com/code/rakhaalcander/ucl-2021-2022-player-data-analysis

Pandasql je *Python* biblioteka koja omogućava da pomoću *SQL* upita izvuku podaci na sličan način kao kod baza podataka. Kompletni *dataset* je moguće izvesti u *CSV* format pomoću *to_csv()* metode *DataFrame* objekta.

Sada kada su podaci dostupni moguće je raditi analizu. Analiza se najčešće radi na nekoj od *Jypiter notebook online* platformi kao što je prethodno pomenuti *Kaggle notebook*. Za ovaj rad je izbor pao na *Google colab* platformu. Na početku analize se obično treba upoznati sa *dataset*-om. Prethodno pomenut *pandas* paket ima funkcije koje kao izlaznu vrijednost imaju veličinu *dataset*-a, broj redova i kolona, koje kolone *dataset* ima, koliko ima nepostojećih vrijednosti itd.

Svaki *DataFrame* objekat ima *shape* i *size* atribute. Ovi atributi čuvaju vrijednosti koji ukazuju na veličinu *dataset*-a. Razlika između *shape* i *size* atributa je taj što *shape* predstavlja torku sa dimenzijama *dataset*-a (redovima i kolonama), a *size* atribut ima vrijednost ukupnog broja vrijednosti *dataset*-a. Vrijednost atributa *shape* za dataset koji se koristi za potrebe ovog rada je **(747, 41)**, dok je vrijednost *size* atributa **30627**.

Tokom čišćenja podataka se često desi da je potrebno pregledati promjene koje su se desile nad *dataset*-om. Najčešće za validiranje promjene nije potrebno pregledati sve redove *dataset*-a, već samo par njih. *Pandas DataFrame* objekat ima metode *head()* i *tail()*. Ove metode kao izlaz daju prvih, odnosno poslednjih 5 redova *dataset*-a ukoliko se metodi kao argument ne proslijedi drugačije. Ove metode su, kao što je navedeno ranije jako korisne za validiranje pomjena koje su se desile nad *dataset*-om tokom čišćenja ili preprocesiranja podataka.

Slika 1 Izlaz head() metode

	Unnamed: 0	player_name	club	position	minutes_played	match_played	goals	assists	distance_covered	conceded	...	off_target	on_target_rate	blocked	pass_accuracy	pass_attem
0	0	Courtois	Real Madrid	Goalkeeper	1230	13	0	0	64.2	14.0	...	NaN	NaN	NaN	76.7	
1	1	Vinícius Júnior	Real Madrid	Forward	1199	13	4	6	133.0	NaN	...	10.0	0.296296	9.0	83.1	
2	2	Benzema	Real Madrid	Forward	1106	12	15	1	121.5	NaN	...	13.0	0.511111	9.0	83.1	
3	3	Modrić	Real Madrid	Midfielder	1077	13	0	4	124.5	NaN	...	3.0	0.357143	6.0	89.8	
4	4	Éder Militão	Real Madrid	Defender	1076	12	0	0	110.4	NaN	...	5.0	0.444444	0.0	87.5	

5 rows × 41 columns

Izvor: Autor (2023) Link

CSV fajlovi su tabelarno organizovne strukture, tj. imaju redove i kolone. *DataFrame* objekat kao učitani *CSV* fajl takođe ima takvu strukturu. Znati kolone *dataset*-a je jako korisno jer se na taj način podaci u redovima stavljaju u kontekst, dobijaju značenje. Kolone *dataset*-a je moguće naći na više načina, najjednostavniji način je korišćenjem atributa *DataFrame* objekta koji se naziva *columns*. Atribut *columns* čuva nazive kolona *dataset*-a kao listu, na slici broj 2 je demonstracija ove metode:

Slika 2 Atribut columns

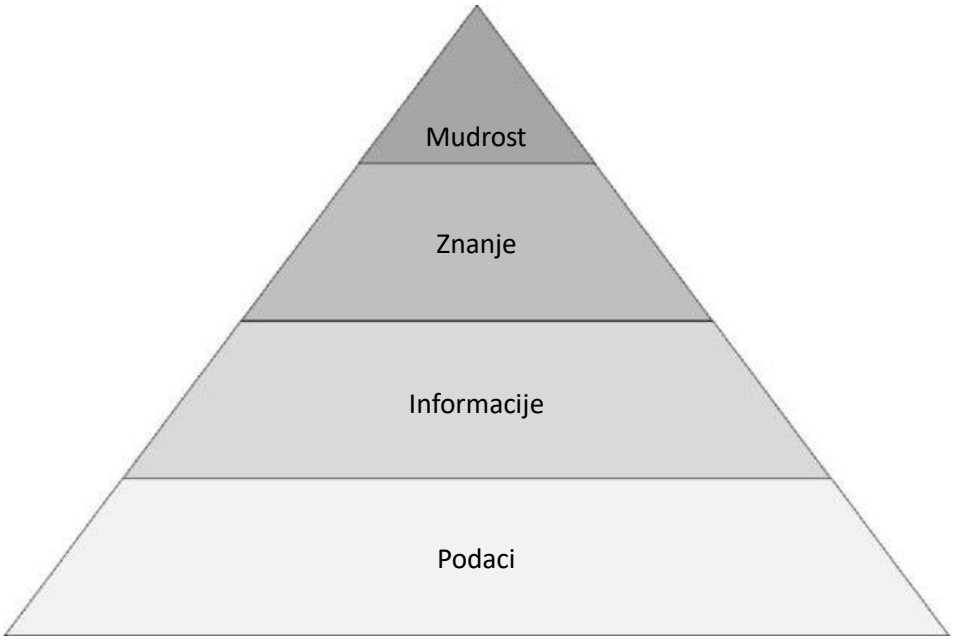
```
Index(['Unnamed: 0', 'player_name', 'club', 'position', 'minutes_played',
       'match_played', 'goals', 'assists', 'distance_covered', 'conceded',
       'balls_recoverd', 'tackles', 't_won', 't_lost', 'clearance_attempted',
       'corner_taken', 'offsides', 'dribbles', 'fouls_committed',
       'fouls_suffered', 'yellow', 'red', 'right_foot', 'left_foot', 'headers',
       'others', 'inside_area', 'outside_areas', 'penalties', 'total_attempts',
       'on_target', 'off_target', 'on_target_rate', 'blocked', 'pass_accuracy',
       'pass_attempted', 'pass_completed', 'cross_accuracy', 'cross_attempted',
       'cross_complted', 'freekicks_taken'],
      dtype='object')
```

Izvor: Autor (2023) Link

3. Analiza podataka

Analiza podataka je kao što je ranije navođeno, proces obrade i predstavljanja podataka kako bi se došlo do zaključaka i kako bi se donijele prave odluke. Podaci su prikaz neke pojave ili objekta koji se mogu sakupljati i organizovati na razne načine. Kada se podacima da kontekst oni postaju informacije. Kao što navodi Martin Braschler: podaci su na dnu piramide znanja, davanjem konteksta podacima nastaju informacije, analizom i interpretacijom informacija nastaje znanje.[6] Piramida znanja iz prethodno referenciranog rada je prikazana na slici 3:

Slika 3 Piramida znanja



Izvor: Braschler, Martin (2019). Applied Data Science. Springer, Cham. Strana br. 24.

Znanje je dakle osnovni cilj analize podataka i njoj sličnih disciplina. Kako bi se došlo do znanja iz podataka potrebno je ispratiti neke jasno definisane korake koji su uglavnom isti kod svakog analiziranja podataka. Koraci u analizi podataka su sledeći:[7]

1. Pronalaženje/sakupljanje podataka
2. Preprocesiranje (čišćenje) podataka

[6] Braschler, Martin (2019). *Applied Data Science*. Springer, Cham. Strana br. 24.
[7] Isto. Strana br. 25.

3. Analiza podataka
4. Vizualizacija i/ili interpretacija podataka
5. Donošenje odluka

Ovo poglavlje u radu će da prati ove korake u objašnjavanju kako je prethodno opisani *dataset* analiziran. Zanemarujući naravno prvi korak jer je on već objašnjen i detaljno opisan u poglavlju 2.2 Izvor podataka.

3.1 Čišćenje podataka

Kao što je navedeno ranije preprocesiranje podataka je drugi korak u analizi podataka. Podaci najčešće nisu u idealnom formatu (nedostajuće vrijednosti, nesortirani podaci, loš format datuma itd.) posebno ako se ti podaci mogu naći na internetu. Čišćenje skupa podataka je jako bitan korak jer olakšava dalju analizu podataka. U ovom poglavlju će se detaljno opisati na koji način je *dataset* od početnog stanja doveden do stanja u kojem se može koristiti u analizi.

3.1.1 Kolone i vrijednosti dataset-a

U poglavlju broj 2.2 o izvoru podataka je predstavljen *columns* atribut koji za *dataset* koji se koristi predstavlja listu od čak 41 kolone. Većina ovih kolona je nepotrebna za analizu koja je planirana za ovaj *dataset*, pa je višak kolona potrebno odstraniti. Kolone je moguće ukloniti tako što se od originalnog *DataFrame* objekta odaberu samo kolone koje treba da ostanu u *dataset*-u. Najinteresantnije kolone za ovu analizu jesu golovi, asistencije i slične kolone.

Od 41 kolone koliko se sadržao originalni *dataset* je ostalo samo 17 najbitnih kolona za ovu analizu. Kao što je moguće primijetiti na slici 7, kolone *dataset*-a su na Engleskom jeziku. Nazive kolona je moguće promijeniti pomoću *Pandas* paketa i metode *rename()*. Ova metoda se razlikuje po tome što kao izlaz ne vraće novi *DataFrame* objekat, već mijenja originalni objekat. Najlakši način da se više kolona preimenuju odjednom je da se napravi *dictionary* (neprimitivni

tip podatka u kojem su podaci raspoređeni po ključevima) u kojem su ključevi prethodna imena kolona, a vrijednosti nova imena kolona.

Dataset-ovi koji su nastali takozvanim "skrejpovanjem" podataka sa interneta, ili u ovom slučaju spajanjem više *dataset*-ova u jedan imaju veliki broj takozvanih *NaN* ili *Null* vrijednosti (nepostojeće vrijednosti). Problem nepostojećih vrijednosti se rešava na dva načina:

1. Uklanjanjem redova sa nepostojećim vrijednostima
2. Njihovom zamjenom sa nekom vrijednošću (najčešće aritmetička sredina kolone)

Broj nepostojećih vrijednosti je pomoću *Pandas* paketa jako jednostavno pronaći, i za to postoji više načina. Jedan način je metodom *info()* koja kao izlaz prikazuje broj vrijednosti po kolonama koje nisu nepostojeće, ali i tip podatka u svakoj od kolona. Drugi način je kombinacijom *isna()* i *sum()* metode, koje kao izlaz vraćaju broj nepostojećih vrijednosti po kolonama. U ovom slučaju je korišćena *info()* metoda zato što prikazuje i tipove podataka po kolonama koji će se mijenjati u nastavku poglavlja.

Slika 4 Izlaz info() metode

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 747 entries, 0 to 746
Data columns (total 17 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Igrač             747 non-null    object
 1   Klub              747 non-null    object
 2   Pozicija          747 non-null    object
 3   Broj_minuta       747 non-null    int64
 4   Broj_mečeva       747 non-null    int64
 5   Golovi            747 non-null    int64
 6   Asistencije       747 non-null    int64
 7   Broj_faulova      582 non-null    float64
 8   Golovi_desnom     183 non-null    float64
 9   Golovi_lijevom    183 non-null    float64
 10  Golovi_glavom     183 non-null    float64
 11  Drugačiji_golovi  183 non-null    float64
 12  Iz_šesnaesterca   183 non-null    float64
 13  Van_šesnaesterca  183 non-null    float64
 14  Penali            183 non-null    float64
 15  Šutevi            543 non-null    float64
 16  Unutar_okvira     543 non-null    float64
dtypes: float64(10), int64(4), object(3)
memory usage: 99.3+ KB
```

Izvor: Autor (2023) Link

Kao što se može vidjeti na slici broj 4, postoje kolone *dataset*-a koji imaju samo 183 postojeće vrijednosti od 747 redova koliko ukupno ima *dataset*. Kao što je rečeno ranije, problem nepostojećih vrijednosti se rešava na dva načina. Prvi je da se obrišu svi redovi sa nepostojećim vrijednostima, a drugi da se nepostojeće vrijednosti zamijene sa nekom vrijednošću. Izbor se pravi u odnosu na to da li je bitan kvantitet podataka (treniranje modela vještačke inteligencije), ukoliko jeste vrijednosti se mijenjaju. A ukoliko kvantitet nije toliko bitan onda se ide na radikalniji način, na brisanje podataka kao i u slučaju ovog rada. Nepostojeće vrijednosti se brišu na veoma jednostavan način, pomoću metode *DataFrame* objekta koja se naziva *dropna()*. Ova metoda vraće novi *DataFrame* objekat koji nema redove sa nepostojećim vrijednostima. Posle primjene ove metode *dataset* je sa 747 redova spao na samo 176 redova.

Na slici broj 4 se takođe mogu vidjeti tipovi podataka po kolonama. Tipovi podataka koji su napisani su zapravo tipovi podataka koje definiše *NumPy* softverski paket. Zašto *Pandas* ne koristi standardne *Python* tipove podataka, već koristi *NumPy* tipove podataka? Razlog je taj što *NumPy* tipovi podataka bili oni osnovni ili kompleksni zauzimaju mnogo manje *RAM*-a od standardnih *Python* tipova podataka. *Python* je objektno-orijentisan programski jezik, pa je i najobičniji cijeli broj u ovom programskom jeziku objekat. Objekti imaju svoje atribute i metode koji zauzimaju određeni prostor u memoriji. Kao što navodi Itamar Turner-Trauring u svom članku: cijeli broj u *Python* programskom jeziku koji može biti predstavljen sa 64 bita, zauzima 28 bajtova, pa lista od milion cijelih brojeva zauzima 35mb (28mb brojevi u listi, i oko 7mb za reference u memoriji).[8] Tako da nije čudno zašto *Python* ima reputaciju kao jako spor programski jezik. Ovaj problem performansi se međutim može riješiti korišćenjem *NumPy* paketa i njegovih tipova podataka. Na slici broj četiri se vidi da je kolona „Golovi“ tipa podatka *int64*. Riječ *int* u ovom tipu podatka označava da se radi o cijelom broju, dok broj koji stoji uz ovu riječ označava koliko bita zauzma jedan takav podatak. U slučaju da kolona golovi ima milion redova, cijela lista bi zauzimala 8mb radne memorije, za razliku od standardne *Python* liste koja zauzima 35mb. Može se dakle zaključiti zašto su i *Pandas* i *Matplotlib*, a i ostale slične bibliokete napravljene upravo na bazi *NumPy*-a.

[8] Turner-Trauring, Itamar (2020). *Massive memory overhead: Numbers in Python and how NumPy helps*.

Pandas paket omogućava da se kolonama u *dataset*-u mijenjaju tipovi podataka. Na slici broj 4, tj. izlazu *info()* metode se može vidjeti da je jako puno kolona za koje bi bilo logičnije da su predstavljene sa tipom podatka *int* predstavljene tipom podatka *float* (decimalni broj). Oba ova tipa podatka zauzimaju istu količinu memorije pa promjena tipa podatka ne bi uticala na performanse, ali je svakako bolje da svaka kolona bude predstavljena odgovarajućim tipom podataka. Metoda koja se koristi za mijenjanje tipova podataka u *Pandas DataFrame*-u se naziva *astype()*. Ovoj metodi se prosleđuje pojedinačna kolona ili više njih kao *dictionary*, i ona kao izlaz vraće novi *DataFrame* objekat sa izmijenjenim tipovima podataka. Kao i sa metodom *rename()* ključevi *dictionary*-a koji se presleđuje kao argument metodi jesu nazivi kolona, a vrijednosti su tip podatka u koji je potrebno promijeniti tu kolonu. Kao tip podatka se mogu proslijediti ključne riječi već ugrađene u *Python* (*int*, *float*, *bool*...), ili se mogu proslijediti i tipovi podataka koji su definisani u *NumPy* paketu (*int8*, *float32*...).

Kolone koje sadrže podatke o klubovima i igračima imaju tip podatka *object*, taj tip podatka se koristi za predstavljanje tekstualnih podataka. Klub i pozicija su takozvane „kategoričke“ vrijednosti, vrijednosti ovih kolona se ponavljaju kroz redove *dataset*-a (u klubu igra više igrača, jednu poziciju igra više igrača). Vrijednosti kolone pozicija su na Engleskom jeziku, a vrijednosti u koloni klub su netačno ili nepotpuno navođeni nazivi klubova. Osim promjene imena i tipa podatka u kolonama, *Pandas* omogućava i promjenu vrijednosti u kolonama korišćenjem metode *replace()*. Ova metoda mijenja originalni *DataFrame* objekat. I ovoj metodi se takođe prosleđuje *dictionary*, ali malo drugačije strukture. Ključevi *dictionary*-a su nazivi kolona, a vrijednosti su *dictionary*-ji u kojima su vrijednosti koje treba promijeniti ključevi, a vrijednosti koje treba da zauzmu njihovo mjesto vrijednosti.

3.1.2 Indeksi i sortiranje

Redovi u *Pandas DataFrame* objektu imaju dodijeljen indeks. Indeks je u suštini jedinstveni identifikator tog reda. Obično je to neka od kolona *dataset*-a, ili redni broj ukoliko indeks kolona nije navedena tokom konverzije fajla u *DataFrame* objekat. U slučaju *dataset*-a koji

je korišćen u ovom radu indeks predstavlja redni broj. Međutim redosled indeks kolone je izgubljen tokom brisanja nepostojećih vrijednosti *dataset*-a, pa ga je potrebno ponovo postaviti u normalu sa metodom *reset_index()*. Prije ponovnog indeksiranja redova je korisno sortirati vrijednosti po golovima i asistencijama, kako bi se kasnije tokom izrade veb aplikacije lakše dobili potrebni podaci. Sortiranje *dataset*-a se radi pomoću metode *sort_values()*, i redove je moguće sortirati na osnovu više kolone u rastućem ili opadajućem poretku. Sortiranje i ponovno indeksiranje je moguće uraditi u jednoj liniji *Python* koda kombinacijom metoda *sort_values()* i *reset_index()*, tako što se nad objektom prvo pozove metoda za sortiranje, a zatim metoda *reset_index()*. Sa ovim je završeno čišćenje i manipulacija nad podacima, sledeći koraci su analiziranje podataka i vizuelna prezentacija istih, što će biti pokriveno u sledećem poglavlju.

3.2 Analiziranje i vizualizacija podataka

U ovom poglavlju će biti pokriven proces analize podataka i vizualnog predstavljanja istih. Analiza u slučaju ovog rada počinje pregledom osnovnih statističkih parametara u podacima, npr. aritmetička sredina, medijana, maksimalna i minimalna vrijednost itd. *Pandas* je ovaj dio analize olakšao svojom *describe()* metodom. Ova metoda zao izlaz vraće tabelarni prikaz aritmetičke sredine, standardne devijacije, medijane itd. za svaku od numeričkih kolona u *dataset*-u. Demonstracija metode *describe()* se nalazi na sledećoj slici:

Slika 5 Izlaz describe() metode

	Broj_minuta	Broj_mečeva	Golovi	Asistencije	Broj_faulova	Golovi_desnom	Golovi_lijevom	Golovi_glavom	Drugačiji_golovi
count	176.000000	176.000000	176.000000	176.000000	176.000000	176.000000	176.000000	176.000000	176.000000
mean	458.085227	6.914773	2.017045	0.835227	5.863636	0.948864	0.704545	0.335227	0.022727
std	232.963660	2.504535	2.026889	1.186162	3.644744	1.403139	1.157920	0.619777	0.149458
min	38.000000	2.000000	1.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000
25%	293.000000	5.000000	1.000000	0.000000	3.000000	0.000000	0.000000	0.000000	0.000000
50%	447.000000	6.000000	1.000000	0.000000	5.000000	1.000000	0.000000	0.000000	0.000000
75%	577.250000	8.000000	2.000000	1.000000	8.000000	1.000000	1.000000	1.000000	0.000000
max	1199.000000	13.000000	15.000000	6.000000	19.000000	11.000000	8.000000	3.000000	1.000000

Izvor: Autor (2023) Link

Metoda *describe()* je korisna iz razloga što je u njenom izlazu moguće analizirati u kojem opsegu se nalaze podaci. Na slici 5 se na primjer može vidjeti da je maksimalna vrijednost u koloni sa golovima broj 15, a aritmetička sredina te kolone približno jednaka broju 2. U opsegu od 2 gola se i nalazi 75% igrača, što znači da je ¾ igrača u ovom *dataset*-u dalo između 1 i 2 gola. Pa se može zaključiti da je igrač koji je dao 15 golova daleko premašio prosječnog igrača, ali i 75% svih igrača u *dataset*-u.

3.2.1 Korelacija u podacima

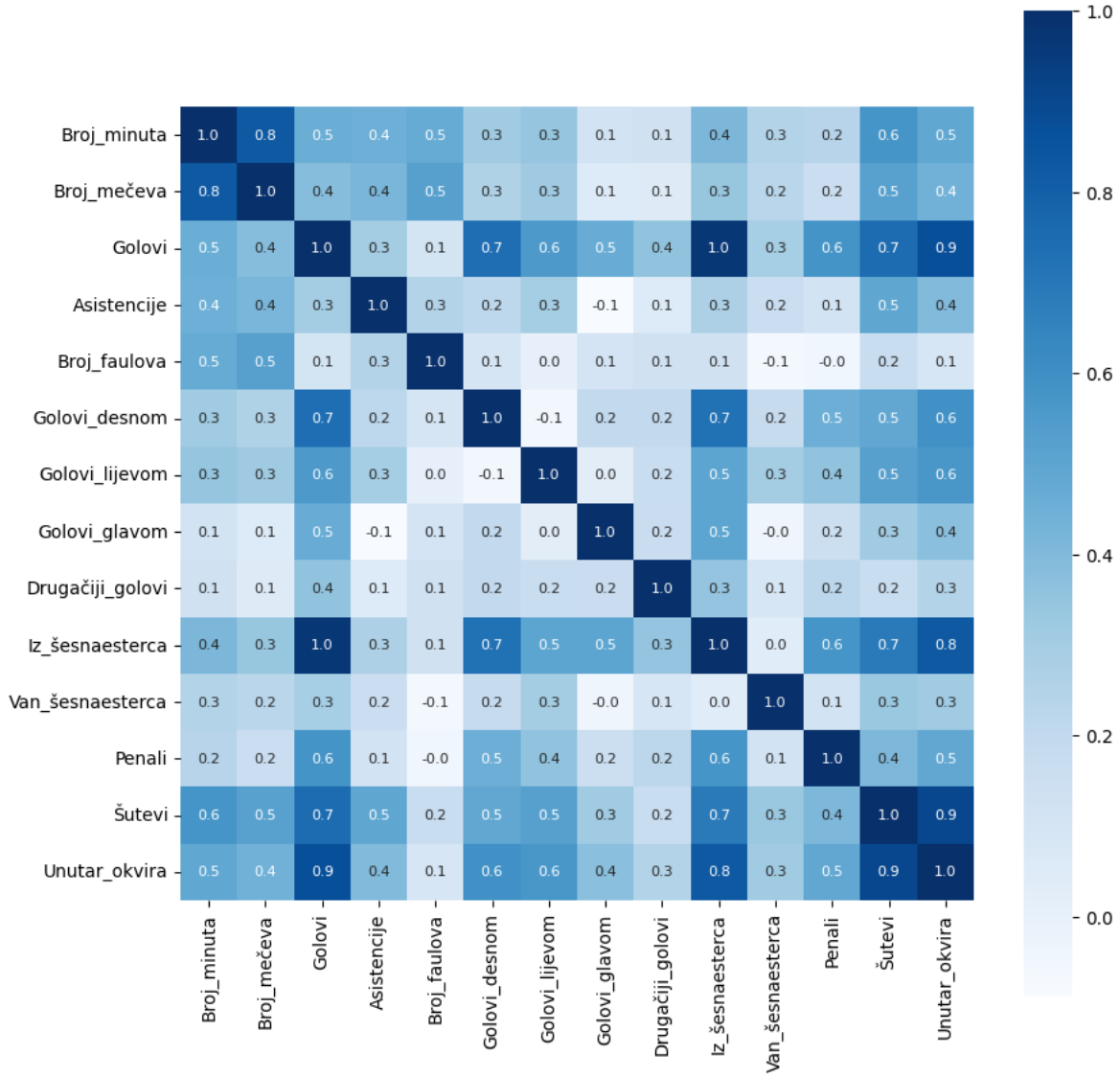
Podaci mogu da budu u međusobnim vezama. Veza između podataka se može primijetiti kada promjena jednog podatka prouzrokuje promjenu drugog podatka. Na primjer broj sati učenja je u vezi sa rezultatom na ispitu, što je broj sati učenja veći to će i rezultat na ispitu biti bolji. Ili suprotno tome količina fizičke aktivnosti i vjerovatnoća obolijevanja od neke bolesti su takođe u međusobnoj vezi, što je više fizičke aktivnosti to je vjerovatnoća obolijevanja manja. Ova veza između podataka se naziva korelacija, a brojčana vrijednost kojom se ona izražava se naziva koeficijent korelacije. Prema Brusu Ranteru koeficijent korelacije se interpretira po sledećim pravilima:[9]

1. Koeficijent 0 označava da veza između varijabli ne postoji
2. Koeficijent 1 označava savršenu pozitivnu korelaciju (kada jedna varijabla raste, druga takođe raste)
3. Koeficijent -1 označava savršenu negativnu korelaciju (kada jedna varijabla raste, druga opada)
4. Koeficijent između 0 i 0.3 (0 i -0.3) označava slabu pozitivnu (negativnu) korelaciju
5. Koeficijent između 0.3 i 0.7 (-0.3 i -0.7) označava osrednju pozitivnu (negativnu) korelaciju
6. Koeficijent između 0.7 i 1 (-0.7 i -1) označava jaku pozitivnu (negativnu) korelaciju

[9] Ratner, B. (2009). *The correlation coefficient: Its values range between+ 1/− 1, or do they?* Journal of targeting, measurement and analysis for marketing, 17(2). Strane br. 139 i 140.

Koeficijent korelacije je jako bitan podatak u analiziranju nekog skupa podataka, jer se pomoću njega utvrđuje koje kolone u *dataset*-u zavise jedna od druge. *Pandas* omogućava jednostavan način da za prikaz koeficijenta korelacije između kolona korišćenjem metode *DataFrame* objekta pod nazivom *corr()*. Ova metoda vraće novi *DataFrame* objekat nalik na matricu u kojem su indeksi redova i kolone zapravo nazivi kolona iz dataset-a, a ćelije u samom objektu vrijednosti koeficijenta korelacije. Vizualni prikaz ovog objekta se može atraktivnije predstaviti pomoću *Seaborn* paketa kao na sledećoj slici:

Slika 6 Mapa koeficijenata korelacije



Izvor: Autor korišćenjem Seaborn paketa (2023) Link

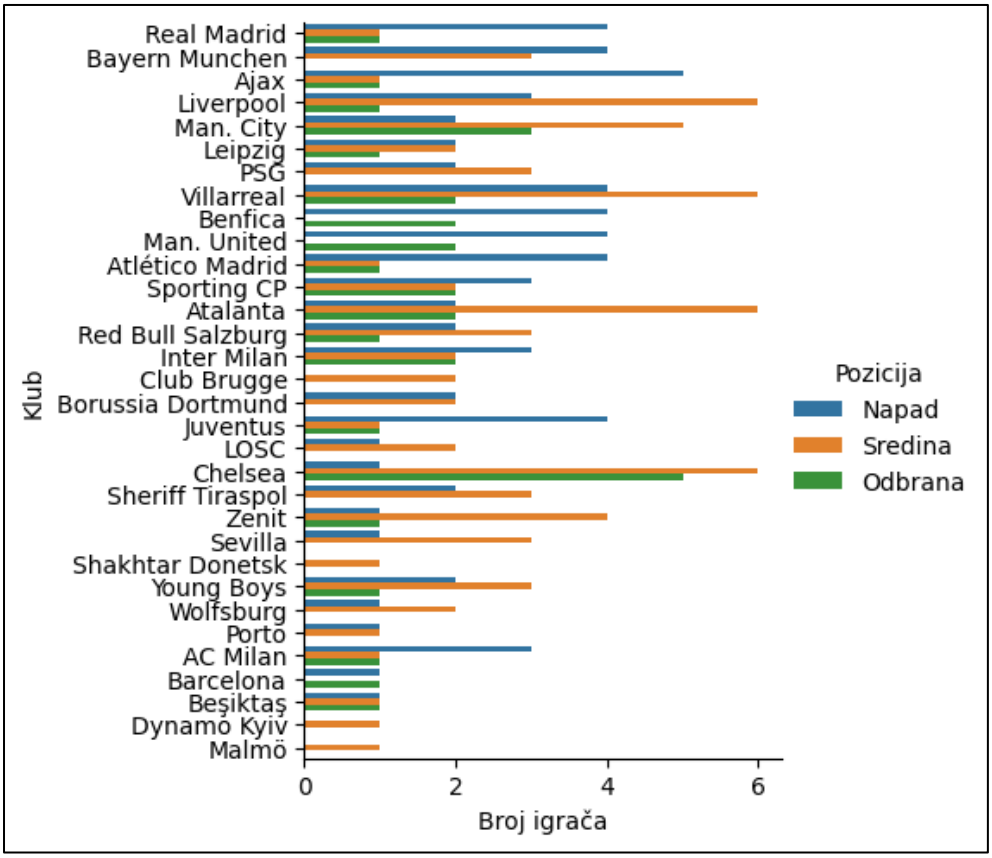
Na slici broj 6 su prikazani koeficijenti korelacije između svake kolone u *dataset*-u. Sa slike se može primijetiti da su na primjer golovi u perfektnoj jakoj pozitivnoj korelaciji sa šutevima unutar okvira gola, a u slaboj pozitivnoj korelaciji sa brojem prekršaja. Takođe su golovi u perfektnoj korelaciji sa golovima iz šestaesterca, međutim golovi unutar šesnaesterca su uključeni u ukupan broj golova. Tako da iako je moguće izračunati koeficijent korelacije u ovom slučaju, kada se ubaci u kontekst on ne predstavlja značajan statistički podatak (jer broj golova unutar šesnaesterca mora biti manji ili jednak ukupnom broju golova). Veliki broj jakih pozitivnih korelacija između kolona je upravo posledica toga što je veliki broj kolona uključen u nekoj drugoj koloni (golovi desnom/lijevom nogom, golovi glavom, šutevi unutar okvira gola i dr.).

3.2.2 Vizualizacija podataka

Vizualno predstavljanje podataka je jako važan korak u analiziranju nekog izvora podataka. Rezultate koji su ostvareni analizom podataka treba predstaviti članovima tima, kolegama sa posla, publici na javnom obraćanju itd. Kako bi svi podjednako razumjeli suštinu, ono što se prezentuje treba biti predstavljeno na jednostavan i atraktivan način. Vizualnu prezentaciju podataka iz *DataFrame* objekata omogućavaju *Python* paketi kao ranije pomenuti *Matplotlib* (za jednostavne grafike) i *Seaborn* (za kompleksne grafike).

U ovom poglavlju će biti predstavljeni grafici napravljeni kroz *Google Colab* platformu, ostali grafici će biti predstavljeni u poglavlju vezanom za veb aplikaciju. U nastavku grafik koji prikazuje broj igrača po pozicijama za svaki od klubova u *dataset*-u:

Slika 7 Pozicije po klubovima

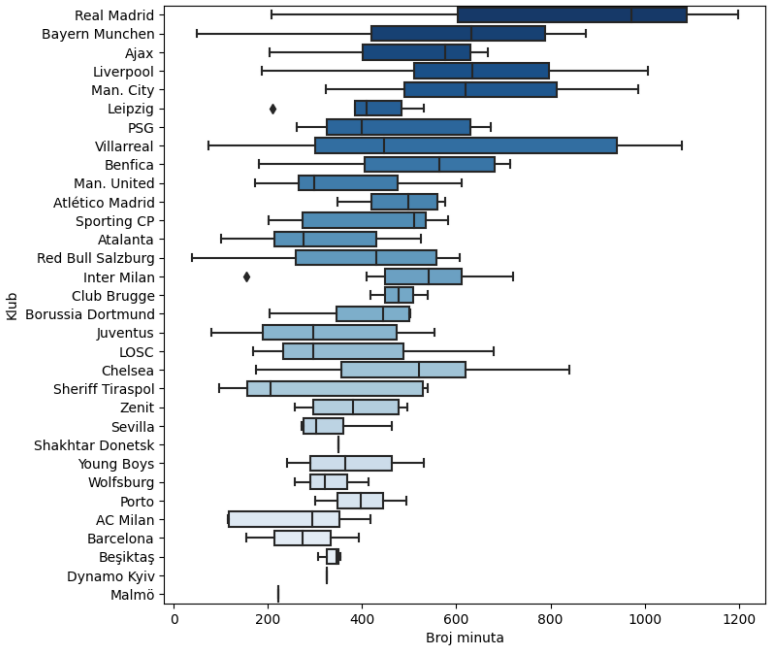


Izvor: Autor korišćenjem Seaborn paketa (2023) Link

Grafik predstavljen na slici 7 je takozvani „*catplot*“ koji je realizovan preko *Seaborn* paketa. Izgled grafika je jako pregledan s obzirom na to da u *dataset*-u ima ukupno 32 kluba, i po tri pozicije za svaki klub. Ostali dijagrami bi imali problem preglednosti sa tolikim brojem kategoričkih vrijednosti, pa je zato ovaj dijagram bio idealan za predstavljanje distribucije igrača po pozicijama za svaki od klubova.

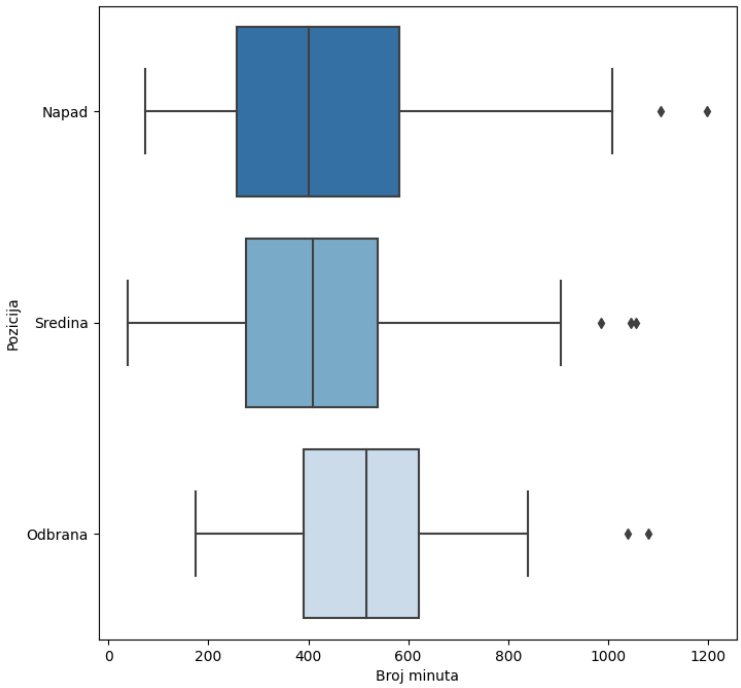
Naredni dijagrami predstavljaju u kojem rasponu minuta meča su igrači odigrali prema klubovima, ali i prema pozicijama.

Slika 8 Raspon minuta koji su igrači odigrali po klubovima



Izvor: Autor korišćenjem Seaborn paketa (2023) Link

Slika 9 Raspon minuta koji su igrači odigrali po pozicijama



Izvor: Autor korišćenjem Seaborn paketa (2023) Link

Slike 8 i 9 su demonstracije takozvanog „*boxplot*-a“ realizovanog pomoću *Seaborn* paketa. Prema Kristin Poter *boxplot* se tumači na sledeći način:[10]

- Krajevi horizontalnih linija su obično minimalna i maksimalna vrijednost opsega
- „Kutija“ predstavlja opseg od prvog do trećeg kvartila (vrijednosti 25% i 75% sa slike 5)
- Vertikalna linija unutar kutije predstavlja medijanu
- Dijamanti van opsega linije su ekstremno male ili velike vrijednosti

Ostale mnogobrojne vizualne reprezentacije podataka, ali i kompletan proces analize podataka je moguće vidjeti u *Google Colab notebook*-u koji je korišćen u praktičnom dijelu ovog rada na ovom link-u.

4. Razvoj veb aplikacije

Veb aplikacije postaju sve popularnija vrsta aplikacija, čak i u poslovnom sektoru. Razlog za to je što veb aplikacije ne zahtijevaju puno resursa računara i ne moraju se ažurirati i instalirati na svakom uređaju kao *desktop* i mobilne aplikacije. Još jedan od razloga je sve veća popularnost radnih okvira koji značajno olakšavaju i ubrzavaju proces razvoja veb aplikacija, npr. *React*, *Vue.js* na *frontend*-u i *Laravel*, *Express.js*, *Ruby on Rails* i *Django* na *backend*-u. *Django* je konkretno kao radni okvir tema ovog poglavlja, ali i kompletnog rada.

Django je *Python*-ov radni okvir koji je u kratkim crtama već opisan u poglavlju 2.1 o softverskim paketima koji su korišćeni. Pomoću *Django*-a se mogu razvijati *fullstack* aplikacije, ali i *REST API* servisi pomoću njegovog dodatka zvanog *Django REST Framework*. Kompatibilan je sa velikim brojem *DBMS*-a (sistem za upravljanje bazom podataka) i drugih radnih okvira (kako *frontend* tako i *backend*). Ovaj paket omogućava programeru da vrlo lako implementira neke komponente koje su uglavnom dio svake veb aplikacije, kao na primjer:

- Autentifikacija korisnika

[10] Potter, Kristin (2006). *Methods for presenting statistical information: The box plot*. In VLUDS. Strana br. 98.

- Povezivanje sa bazom podataka
- Administracija
- Rutiranje itd.

Programer na taj način ne gubi vrijeme da bi implementirao sistem autentifikacije ili *admin panel*, već se više koncentriše na glavne funkcionalnosti sistema. Ogromna zajednica uz sve prethodno navedeno *Django* čini izborom tehnologije nekih od najvećih veb aplikacija i servisa.

4.1 Softverska arhitektura

Django-va arhitektura podsjeća na *MVC* arhitekturu. *MVC* je anagram za *Model-View-Controller*, svaka od ovih riječi predstavlja komponentu ove arhitekture. Uloge svake od komponenti u ovoj arhitekturi su sledeće[11]:

- *Model* je esencija aplikacije, to je set klasa koji služe za rješavanje nekog problema. Ova komponenta sistema se obično ne mijenja dokle god postoji problem.
- *View* je jedan ili više interfejsa koje prikazuju podatke iz modela, interfejsi mogu biti grafički, na komandnoj liniji ili *API* (interfejsi koji druge aplikacije koriste)
- *Controller* manipuliše sa *view* komponentom, u najkraćim crtama *controller* upravlja ulaznim podacima, a *view* prikazuje izlazne podatake

Kao što je navedeno u 2.1 poglavlju, *Django* koristi arhitekturu koja se može nazvati *MVTU* (*Model-View-Template-URL*) arhitektura, u kojoj svaka od komponenti predstavlja sledeće:

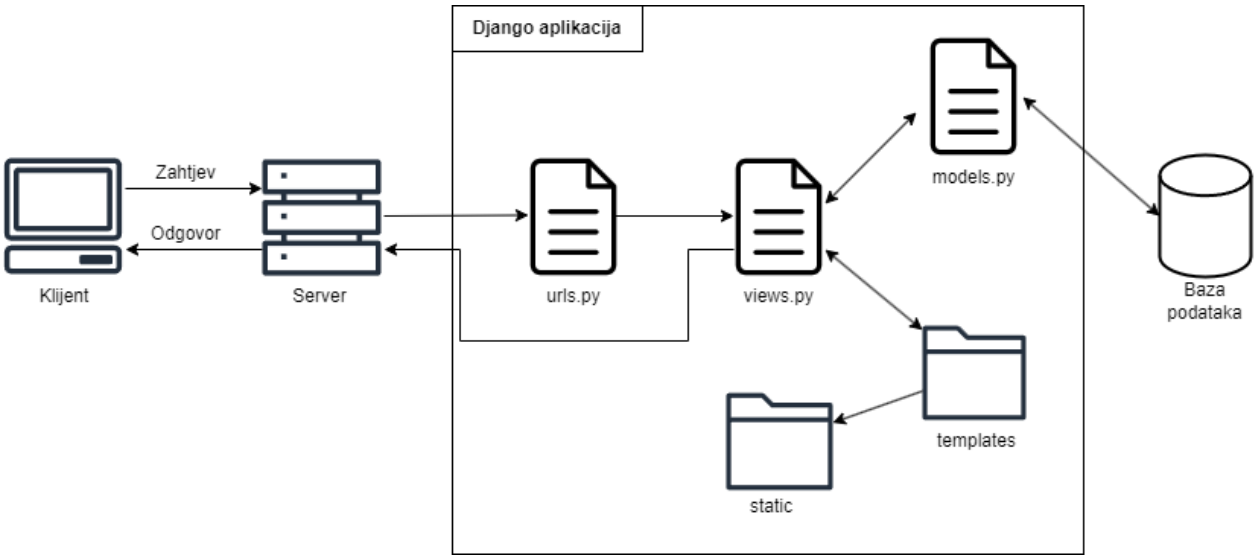
- *Model* komponenta predstavlja skup klasa iz *models.py* fajla koji se generiše pravljenjem *Django* projekta. Klase u ovom fajlu su zapravo tabele u bazi podataka
- *View* komponenta predstavlja skup funkcija ili klasa u *views.py* fajlu koje određuju logiku kojom će se podaci prikazivati na *template* stranicama
- *Template* komponenta je skup *HTML* stranica koje se dinamički mijenjaju u odnosu na podatke koje im se proslijede kroz *view* funkcije.

[11] Deacon, John (2009). *Model-view-controller (mvc) architecture*. Strana br. 2.

- *URL* komponenta definiše rute unutar aplikacije, ali i koje *view* funkcije će biti odgovorne za koje adrese sajta. *URL* adrese se definišu u *urls.py* fajlu

Django arhitektura, tj. na koji način komponente arhitekture komuniciraju jedne sa drugima je prikazana na sledećoj ilustraciji:

Slika 10 Arhitektura Django aplikacije

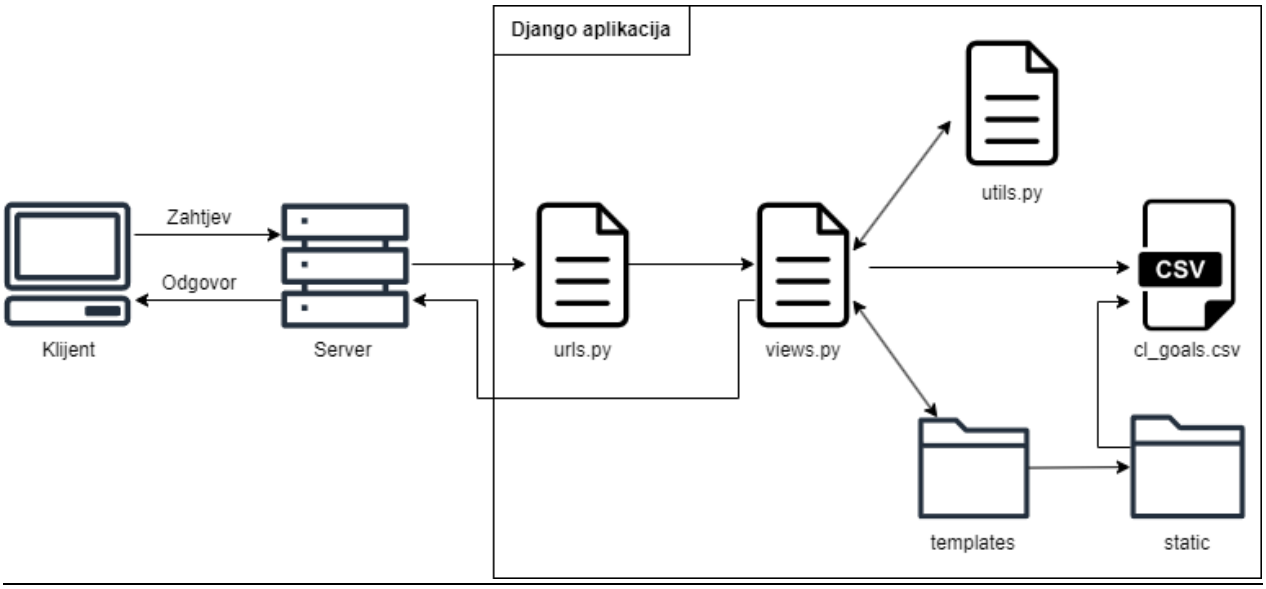


Izvor: Autor pomoću draw.io platforme po uzoru na rad iz literature.[12]

S obzirom na to da veb aplikacija koja se razvija u svrhe pisanja ovog rada koristi *CSV* fajl kao izvor podataka kao izvor podataka, a ne bazu podataka kao na slici 10, arhitektura sistema će funkcionisati na sledeći način:

[12] Bioco, Joao i Rocha, Álvaro. (2019). *Web Application for Management of Scientific Conferences*. In New Knowledge in Information Systems and Technologies. Springer International Publishing. Strana br. 770.

Slika 11 Arhitektura aplikacije iz rada



Izvor: Autor pomoću draw.io platforme po uzoru na rad iz literature[13].

Arhitektura ovog sistema je većinom slična standardnoj *Django* arhitekturi sa slike 10, razlika je ta što se baza podataka ne koristi u ovom sistemu. Samim tim što se ne koristi baza podataka, iz upotrebe se isključuje i *models* fajl. Fajl *utils* koji zauzima njegovo mjesto predstavlja skup funkcija koje služe za vizualizaciju podataka. U folderu *static* se pored uobičajenih statičkih fajlova (*CSS*, *JS*, slike itd.) nalazi i *CSV* fajl koji je rezultat analize podataka. Komponenta *View* pristupa tom fajlu preko *URL*-a i obrađuje ga pomoću paketa *Pandas*. Podaci i grafici se na kraju ubacuju u *template HTML* fajlove i prikazuju korisniku kao veb stranice.

4.2 Stranice

Veb sajtovi su struktuirani po stranicama, kako bi svaka od stranica ima svoju posebnu funkciju. Odvajanje funkcionalnosti po stranicama omogućava bolje korisničko iskustvo jer je

[13] Bioco, Joao i Rocha, Álvaro. (2019). *Web Application for Management of Scientific Conferences*. In New Knowledge in Information Systems and Technologies. Springer International Publishing. Strana br. 770.

korisnicima lakše da pronađu ono zbog čega su posjetili neki veb sajt. Veb aplikacija koja je predmet istraživanja ovog rada ima 5 stranica i to:

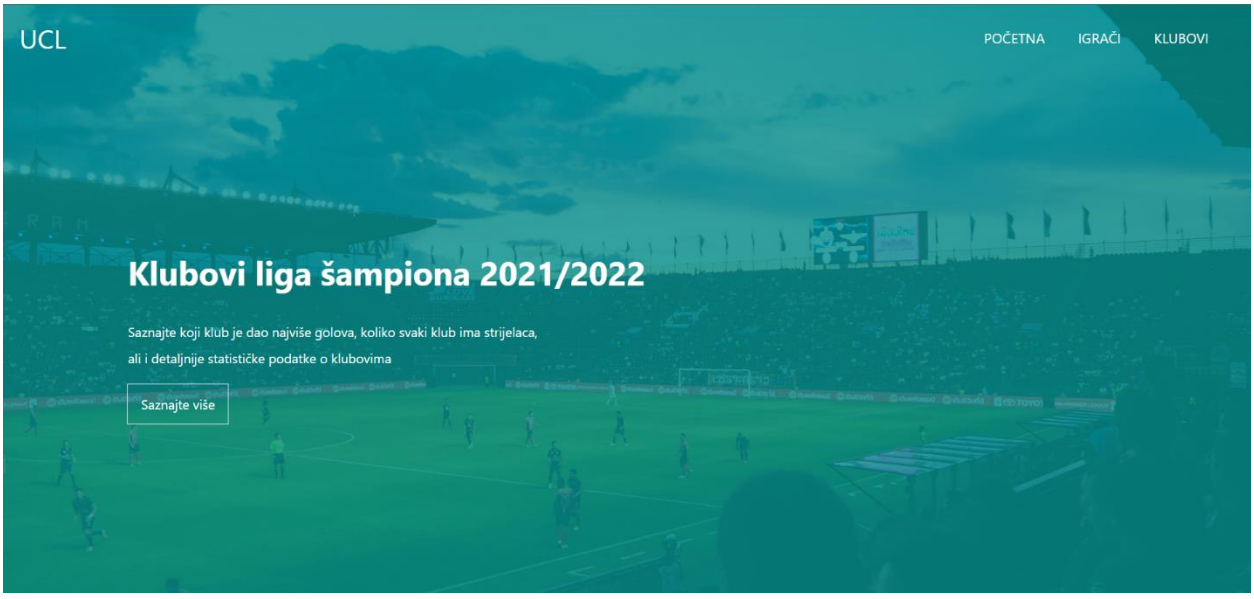
- Početna stranica – prva stranica sa kojom se korisnik susreće pri posjeti sajta
- Stranica klubovi - stranica na kojoj su predstavljeni statistički podaci o svim klubovima
- Stranica igrači - stranica na kojoj su predstavljeni statistički podaci o igračima
- Stranica klub - stranica posvećena pojedinačnom klubu, sa njegovim rezultatima
- Stranica igrač – stranica na kojoj su predstavljeni rezultati pojedinačnog igrača

Svaka od ovih stranica će biti opisana u svom zasebnom poglavlju.

4.2.1 Početna stranica

Kao što je ranije navedeno, to je prva stranica sa kojom se korisnik susreće kada posjeti veb sajt. Ova stranica je obično najatraktivnijeg izgleda jer se upravo na početnoj stranici korisnik odlučuje hoće li ostati na sajtu ili ne. Na početku sajta korisnika očekuje početna sekcija:

Slika 12 Hero sekcija

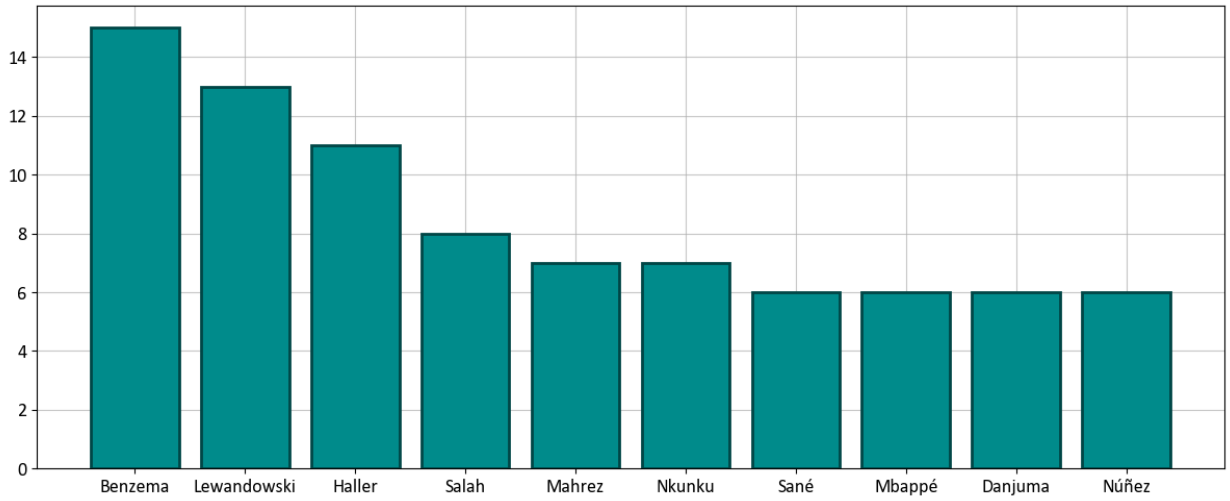


Izvor: Autor (2023) Snimak ekrana veb aplikacije

Početna sekcija, ili kako se još i naziva "*hero*" sekcija je tu da objasni korisniku na kakav je veb sajt došao i kakav sadržaj može da očekuje. Na vrhu sekcije, ali i na vrhu svake stranice se nalazi sekcija za navigaciju koje služe za bolju navigaciju kroz veb sajt.

Sledeća sekcija na početnoj stranici sajta je malo konkretnija od prethodne, u pitanju je grafik najboljih strijelaca Lige šampiona, tj. *dataset*-a. Grafik na stranici izgleda ovako:

Slika 13 Top 10 strijelaca prvenstva



Izvor: Autor (2023) fotografija sa veb aplikacije

Svaki od grafika je generisan sa *Matplotlib* paketom ili u *Seaborn*-om. Generisani grafik je zatim potrebno predstaviti na stranici, a za to postoje dva načina. Prvi je da se grafik čuva kao slika u *Django media* folderu što zauzima mnogo memorijskog prostora, ili da se grafik šifruje pomoću algoritma kao tekstualni podatak, i da se zatim taj tekst proslijedi *HTML* elementu na stranici. Za predstavljanje grafika na ovoj stranici je korišćen *Base64* algoritam za šifrovanje slika kako bi se izbjeglo trošenje memorijskog prostora.

Sledeća sekcija na početnoj stranici je lista klubova koji su se takmičili u Ligi šampiona u sezoni 2021/2022 godine. Sekcija sa klubovima predstavljena na sledećoj fotografiji:

Slika 14 Lista klubova

Izvor: Autor (2023) snimak ekrana veb aplikacije

Svaki od članova liste je link koji vodi do posebne stranice za taj klub. Na toj stranici korisnik može da vidi kakve je rezultate taj klub postigao, koji igrači su postigli pogotke itd.

4.2.2 Stranica klubovi

Na stranici klubovi se nalaze podaci o svim klubovima koji su učestvovali na takmičenju. Za razliku od igrača, klubovi nisu odvojeni po redovima pa se njihovi parametri ne mogu dobiti samo isčitavanjem skupa podataka. Ovaj problem je riješen korišćenjem *Pandas* paketa na način što su klubovi grupisani, a podaci o igračima koji igraju u tim klubovima sabrani kao rezultat tog kluba.

Prva sekcija na ovoj stranici je posvećena najboljem klubu na prvenstu. Najbolji klub se ne odnosi na klub koji je osvojio prvenstvo, iz razloga što ti podaci nisu dostupni u skupu podataka koji se koristi. Najbolji klub je međutim onaj čiji su igrači zajedno postigli najviše pogodaka i asistencija. Prethodno pomenuta sekcija je predstavljena na sledećoj fotografiji:

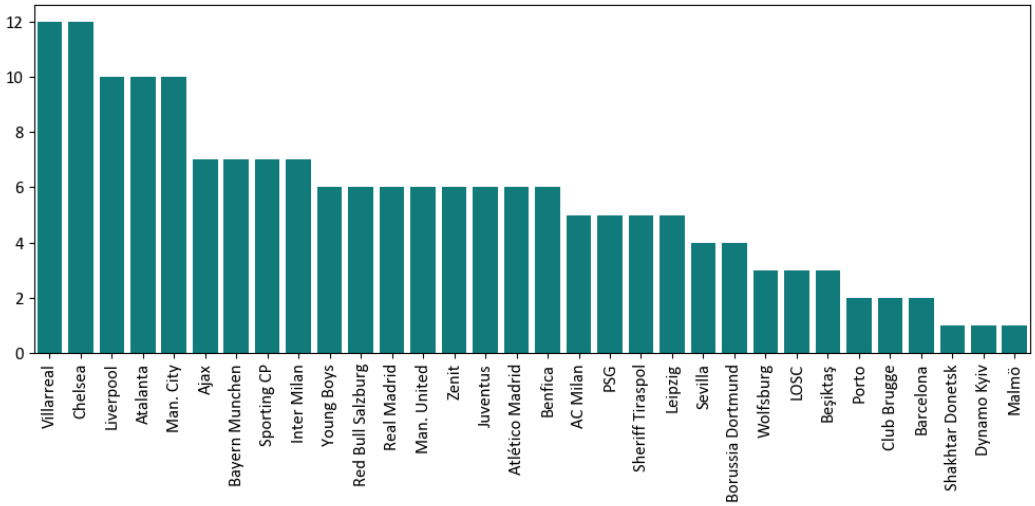
Slika 15 Najbolji klub

Izvor: Autor (2023) snimak ekrana veb aplikacije

Bajern Minhen je na ovom prvenstvu ostvario naviše pogodaka i asistencija. Osim podataka o golovima i asistencijama igrača ovog kluba, napisani su i ostali podaci koji sumirani imaju neku statističku vrijednost. Npr. suma broja mečeva ili minuta koji su igrači odigrali nema statističku vrijednost kada se ti podaci ubace u kontekst, pa se po njima klubovi ne mogu rangirati.

Sledeća sekcija je ove stranice je još jedan grafik. Na ovom grafiku se prikazuje broj igrača po klubovima u *dataset*-u koji su postigli barem jedan pogodak. Pa se može reći da se na grafiku može vidjeti broj strijelaca po klubovima. Grafik je napravljen sa *Seaborn* paketom i izgleda kao na sledećoj fotografiji:

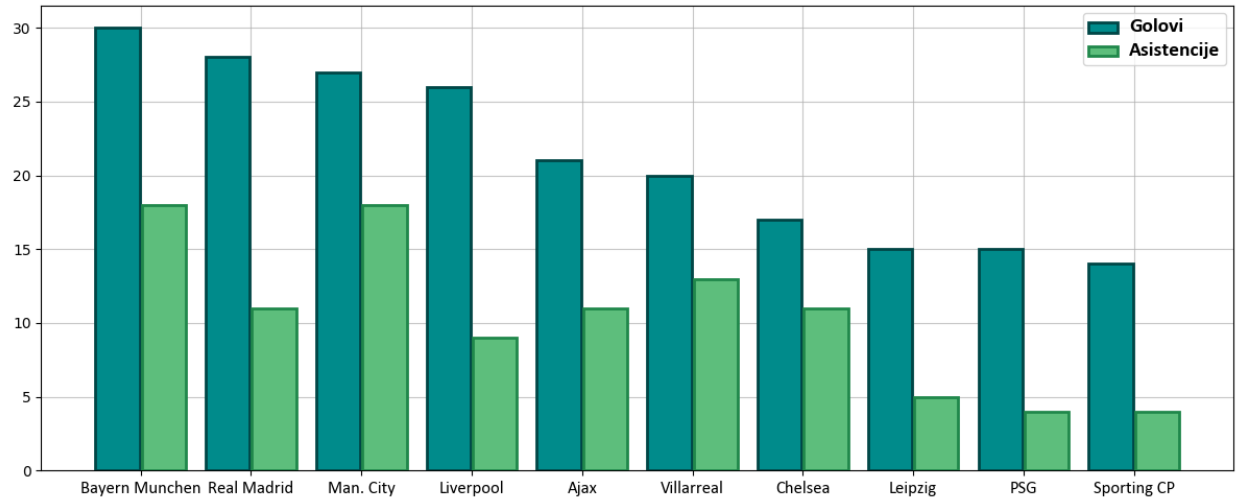
Slika 16 Grafik strijelaca po klubovima



Izvor: Autor (2023) fotografija sa veb aplikacije

Treća sekcija je lista klubova kao i na početnoj stranici, pa je ne treba ponovo objašnjavati. Sekcija posle te je takođe grafik sa stubićima na kojem su predstavljeni 10 najboljih klubova u Ligi šampiona po golovima i asistencijama. Svaki klub na x-osi ima dva stubića od kojih jedan predstavlja broj golova, a drugi broj asistencija. Grafik je napravljen pomoću *Matplotlib* paketa i izgleda kao na sledećoj fotografiji:

Slika 17 Top 10 klubova po golovima i asistencijama



Izvor: Autor (2023) fotografija sa veb aplikacije

4.2.3 Stranica igrači

Na stranici igrači se nalaze sekcije i grafici koji sadrže statističke podatke vezane generalno za igrače koji su učestvovali u Ligi šampiona 2021/2022 godine.

Prva sekcija sa kojom se korisnik susreće kada posjeti ovu stanicu je slična kao kod stranice klubovi, a to je sekcija sa podacima o najboljem strijelcu prvenstva. Na ovoj sekciji je prikazano međutim znatno više parametara za igrača. nego za klub na prošloj stranici. Razlog je taj što neke sumirane parametre kod kluba nije imalo smisla stavljati (odigrani mečevi, minuti...). Ova sekcija je prikazana na sledećoj fotografiji:

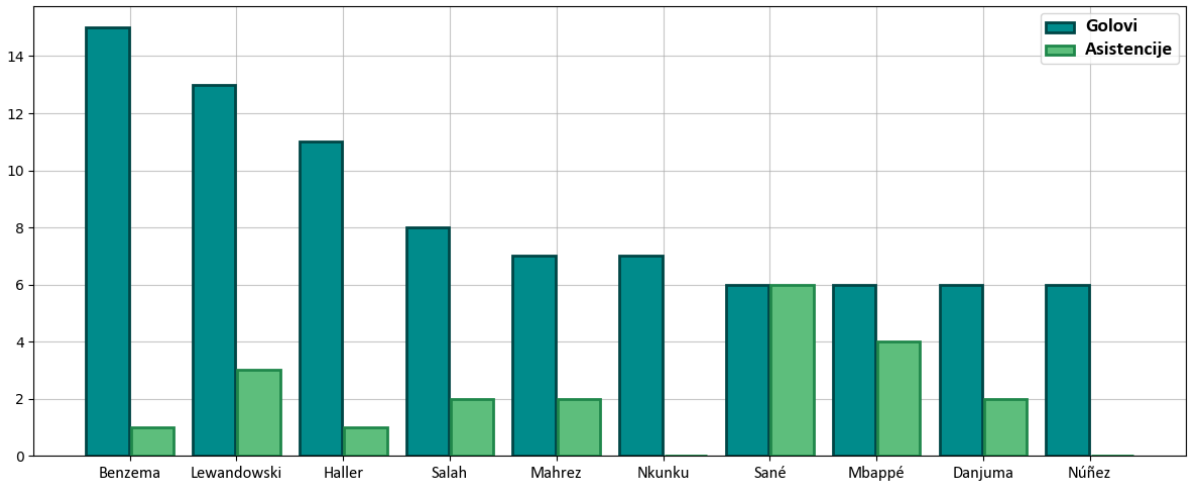
Slika 18 Najbolji strijelac prvenstva

Izvor: Autor (2023) snimak ekrana veb aplikacije

Najbolji strijelac na prvenstvu je napadač iz Real Madrida Karim Benzema koji je ostvario 15 pogodaka na takmičenju. Interesantan podatak je taj što najbolji strijelac prvenstva ne igra u klubu koji je postigao najviše pogodaka, Bajern Minhenu.

Na sledećoj sekciji ove stranice su predstavljeni igrači rangirani po golovima i asistencijama, grafik je prikazan na sledećoj fotografiji:

Slika 19 Najbolji igrači po golovima/asistencijama

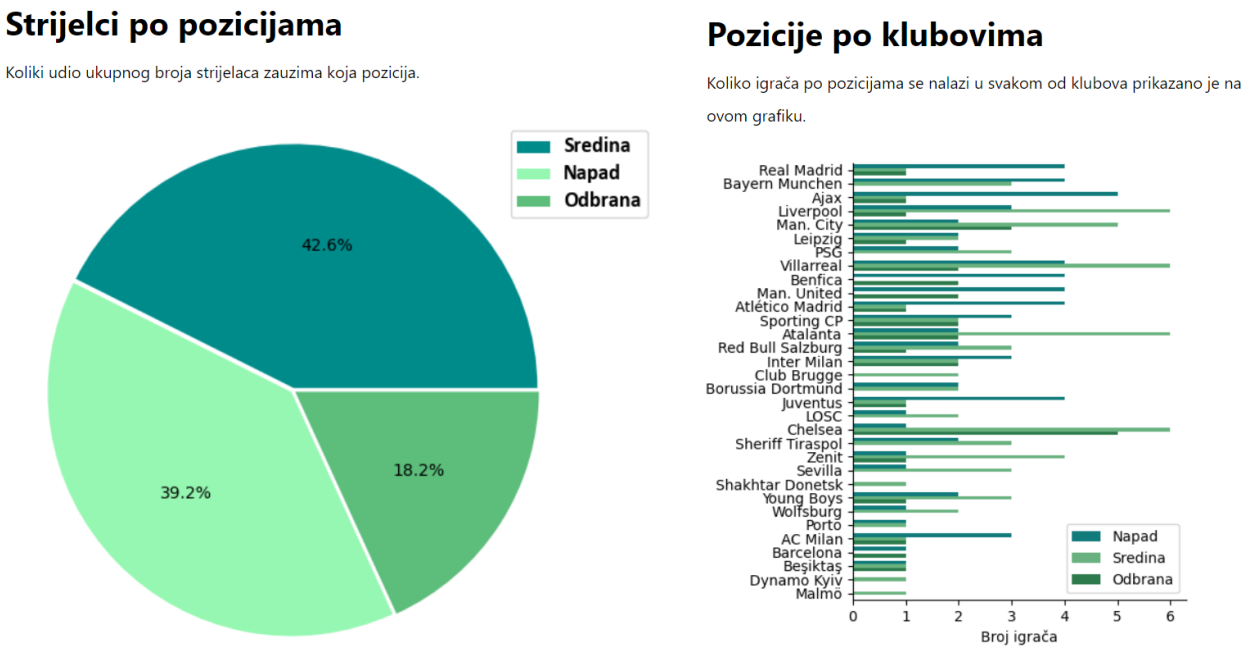


Izvor: Autor (2023) fotografija sa veb aplikacije

Grafik je veoma sličan onome sa slike br. 13, razlika je u tome što grafik sa stranice igrači ima još jedan stubić za svakog igrača koji pokazuje broj asistencija.

Na sledećoj sekciji se nalaze dva grafika. Jedan grafik predstavlja pita dijagram koji pokazuje koji procenat igrača igra koju poziciju, dok je drugi grafik sličan kao grafik sa slike 7 sa razlikama u bojama koje se koriste. Ova sekcija je prikazana na sledećoj fotografiji:

Slika 20 Pita dijagram i catplot na stranici igrači



Izvor: Autor (2023) snimak ekrana veb aplikacije

4.2.4 Stranica klub

Na stranici klub su prikazani podaci o jednom od 32 kluba koliko ih ukupno ima u skupu podataka. Kombinovanjem *Django*-a, *Pandas*-a i *Matplotlib*-a na ovoj, i na stranici klub je omogućeno da se kod za grafike i izvlačenje podataka napiše samo jednom a ne 32 puta, ili 176 u slučaju igrača. Pa se može zaključiti da je ova aplikacija automatizovala proces vizualizacije i analize za pojedinačne klubove i igrače.

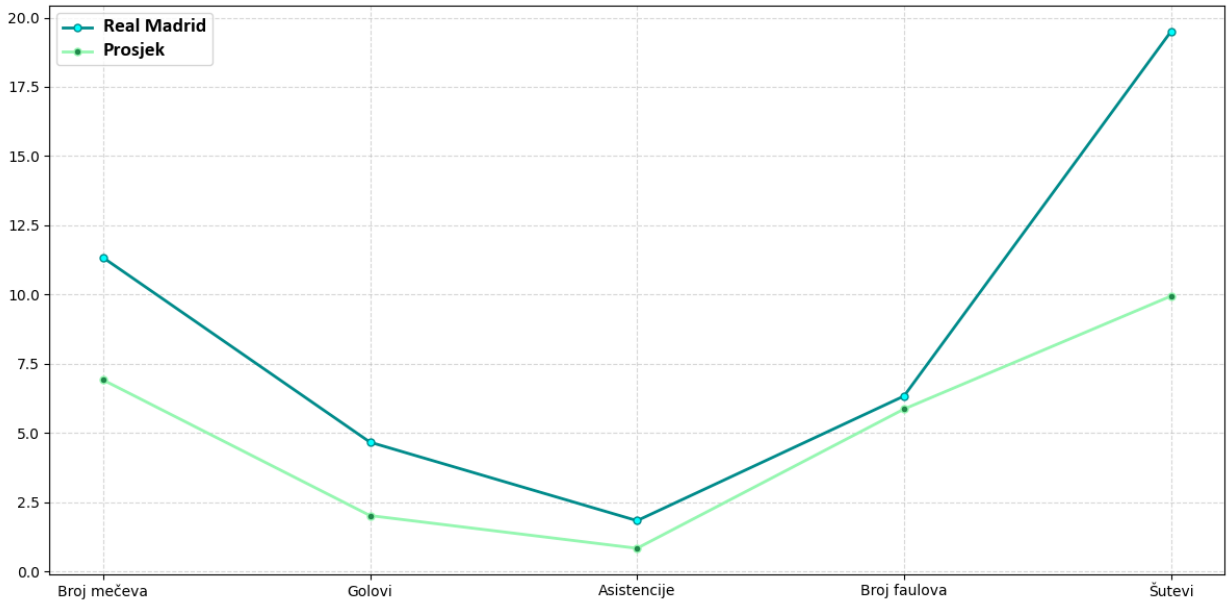
Prva sekcija na stranici klub jeste sekcija na kojoj su prikazani osnovni statistički podaci o klubu, na sličan način kao na lici br. 15. Izgled ove sekcije je prikazan na sledećoj fotografiji:

Slika 21 Prva sekcija na stranici Real Madrida

Izvor: Autor (2023) snimak ekrana veb aplikacije

Sledeći grafik na stranici je grafik koji se nalazi samo na ovoj i na stranici igrač. Radi se o grafiku koji prikazuje koliko je taj klub ili igrač bolji od prosječnog kluba/igrača. Konkretno grafik koji se nalazi na stranici klub je napravljen upoređivanjem prosječnih vrijednosti igrača kluba, i prosjekom svih igrača. Vizualno upoređivanje ovih vrijednosti se predstavlja pomoću dvije linije na koordinatnom sistemu. Korisnik pomoću te dvije linije može da upoređuje po kojim parametrima je neki klub bio bolji od prosječnog kluba, a po kojim je isti bio gori od prosjeka. Izgled ovog grafika je prikazan na sledećoj slici:

Slika 22 Upoređivanje rezultata Real Madrida i prosječnog tima



Izvor: Autor (2023) fotografija sa veb aplikacije

Sledeća sekcija je tabela u kojoj se nalaze svi igrači kluba sa njihovim postignutim golovima, asistencijama itd. Osim tih podataka u tabeli se nalazi link koji vodi na stranicu svakog od igrača. Tabela na sajtu izgleda kao na sledećoj fotografiji:

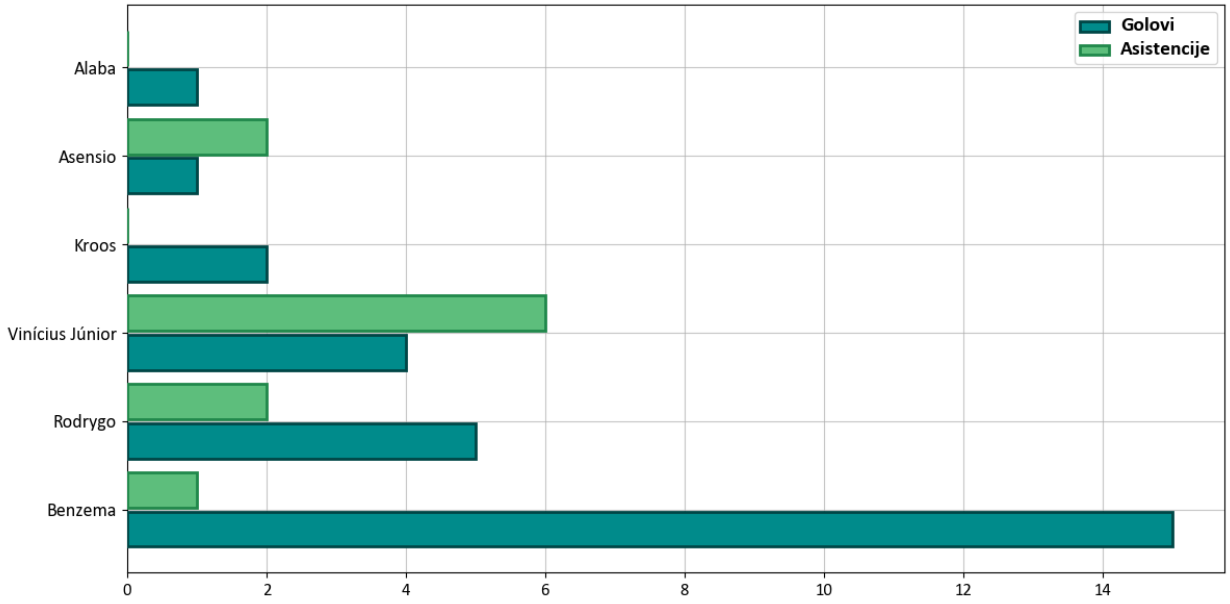
Slika 23 Fudbaleri koji igraju za Real Madrid

Igrač	Pozicija	Broj minuta	Broj mečeva	Golovi	Asistencije	Broj faulova	Broj šuteva	
Benzema	Napad	1106	12	15	1	2	45	Saznaj više
Rodrygo	Napad	505	11	5	2	5	23	Saznaj više
Vinícius Júnior	Napad	1199	13	4	6	13	27	Saznaj više
Kroos	Sredina	902	12	2	0	7	12	Saznaj više
Asensio	Napad	207	8	1	2	2	5	Saznaj više
Alaba	Odbrana	1040	12	1	0	9	5	Saznaj više

Izvor: Autor (2023) snimak ekrana veb aplikacije

Poslednja sekcija na ovoj stranici je grafik na kojem se nalaze golovi i asistencije svakog igrača kluba. Grafik prikazuje podatke kao horizontalne stubiće, i izgleda kao na narednoj fotografiji:

Slika 24 Golovi i asistencije fudbalera Real Madrida



Izvor: Autor (2023) fotografija sa veb aplikacije

4.2.5 Stranica igrač

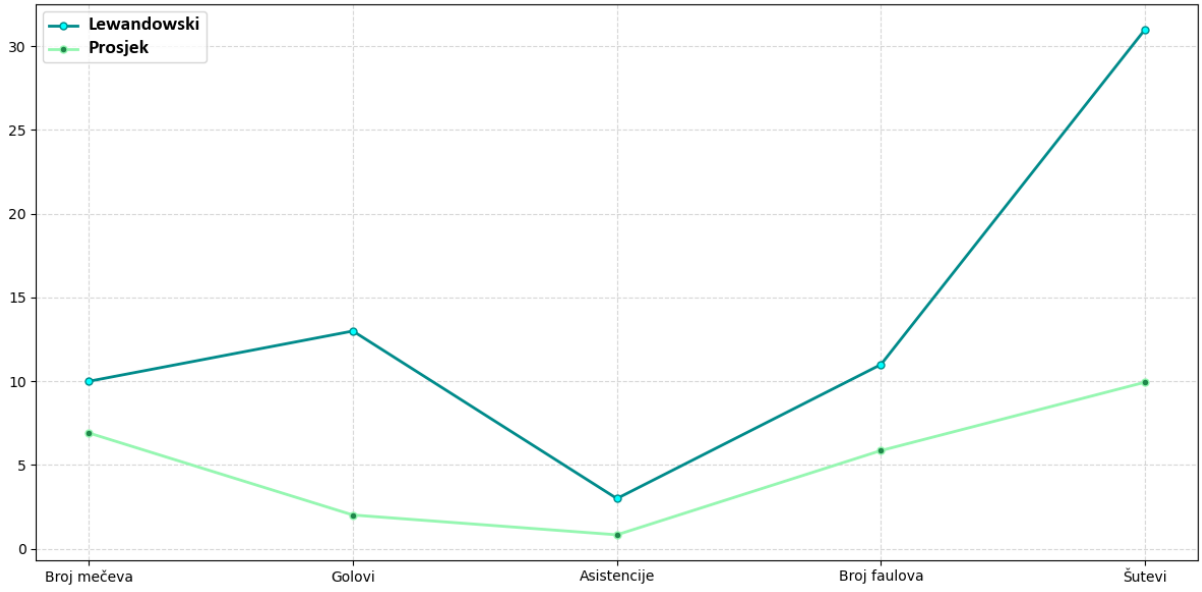
Stranica igrač je stranica koja prikazuje podatke i grafike za jednog konkretnog igrača. Na prvoj sekciji ove stranice se nalaze podaci o tom igraču (golovi, asistencije, šutevi, odigrani mečevi itd.), slična sekcija je prikazana na slici br. 18. Sekcija na stranici za igrača Bajerna pod imenom Lewandowski je prikazana na sledećoj fotografiji:

Slika 25 Robert Lewandowski i njegovi rezultati

Izvor: Autor (2023) snimak ekrana veb aplikacije

Naredna sekcija je grafik kojim se vizualno predstavlja upoređivanje rezultata igrača sa stranice i prosječnog igrača. Ovaj grafik izgleda ovako:

Slika 26 Robert Lewandowski protiv prosječnog igrača

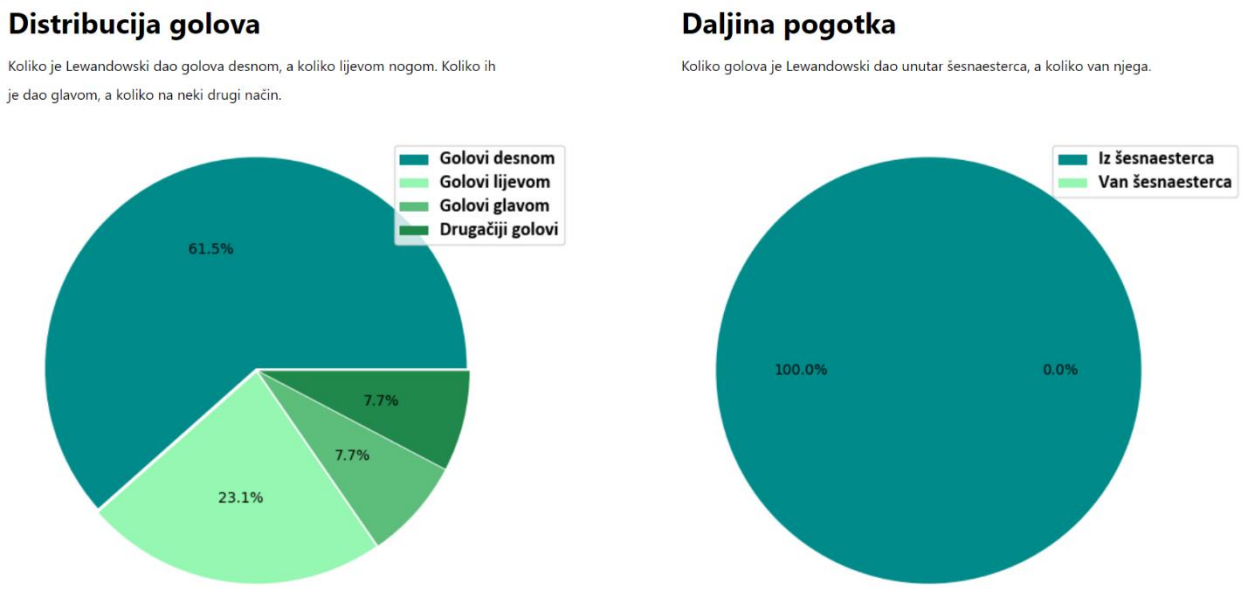


Izvor: Autor (2023) fotografija sa veb aplikacije

Poslednja sekcija na ovoj stranici je sekcija sa dva pita dijagrama. Jedan pokazuje distribuciju golova po načinu pogotka (desnom nogom, lijevom nogom, glavom ili drugačiji način).

Dok drugi dijagram pokazuje distribuciju golova po distanci pogotka (unutar ili van šesnaesterca). Sekcija sa graficima je prikazana na sledećoj fotografiji:

Slika 27 Golovi koje je postigao Lewandowski



Izvor: Autor (2023) snimak ekrana veb aplikacije

5. Diskusija

Radovi koji na sličan način obrađuju ovu temu su rijetkost, posebno ukoliko se govori o praktičnom rješenju. Neki od primjera u literaturi su sledeći:

- Prvi sistem se bavi vizualizacijom podataka iz realnog biznis sektora na vebu, urađen takođe u *Django*-u. Sistem iz baze podataka izvlači podatke koji se vizualno predstavljaju kroz grafike. [14]
- Drugi sistem je intranet platforma za praćenje prisustva studenata. Platforma ima par tipova korisnika sa različitim permisijama unutar sistema. Studentima i zaposlenima koji

[14] Vamsi, K. (2021). *Visualization of Real World Enterprise Data using Python Django Framework*. IOP Conference Series: Materials Science and Engineering.

koriste ovaj sistem su predstavljeni dijagrami prisustva studenta ili studenata u zavisnosti od tipa korisnika.[15]

Pored dosta sličnosti koje ovi sistemi dijele sa sistemom opisanim u ovom radu, postoje i značajne razlike. Tehnologije koje se koriste su uglavnom iste sa razlikama u izboru arhitekture. Nijedan od dva prethodno navedena sistema ne koristi *Pandas* paket kako bi pristupio podacima iz fajla. Naprotiv oba sistema koriste bazu podataka za čuvanje podataka. U kojim situacijama onda treba koristiti *Pandas*, a u kojim treba koristiti bazu podataka?

Prednost *Pandas*-a u odnosu na *SQL* je taj što *Pandas* može da parsira različite tipove fajlova (*CSV*, *Excel*, *HTML*, *XML* itd.), a može da parsira čak i *SQL* upit. Druga prednost *Pandas*-a u odnosu na *SQL* je ta kompatibilnost sa drugim paketima koji se koriste u analizi podataka, iako se ne može se reći da je *Pandas* kompatibilan sa *Django* paketom. Tu se sve prednosti *Pandas* paketa završavaju. Za razliku od *Pandas*-a, *SQL* može da obavlja *CRUD* operacije (kreiranje, čitanje, mijenjanje i brisanje podataka). Teoretski *CRUD* operacije može da obavlja i *Pandas*, ali se onda praktičnost svega toga dovodi u pitanje. Brzina odgovora predstavlja veliku prednost *SQL*-a u odnosu na *Pandas*, iz razloga što je brzina značajan faktor kvaliteta veb aplikacije. Prema pisanju Šulea: *Umbra* sistem za upravljanje bazama podataka ima značajno bolje performanse u odnosu na *Pandas*, dok *PostgreSQL* pokazuje bolje performanse od *Pandas*-a nad većim skupovima podataka. [16]

Na koji način bi ovaj sistem mogao da bude bolji? Sistem je jako ograničem analiziranjem jedne sezone jednog takmičenja. Uvođenjem baze podataka u arhitekturu ovog sistema bi se riješio taj problem ograničenosti pomoću *CRUD* operacija. Naredno poboljšanje koje bi popelo ovaj sistem na veći nivo jeste odvajanje *frontend*-a i *backend*-a. *Frontend* aplikacije bi bio napravljen u nekom od *JavaScript* radnih okvira kao što su *React* ili *Vue.js*. Dok bi *backend* bio napravljen pomoću *Django REST framework*-a u obliku *REST API*-ja. Ovaj *API* bi funkcionisao tako što bi podatke iz baze podataka slao u *JSON* formatu ka *frontend*-u. Grafici koji se prikazuju na stranicama aplikacije bi mogli da se šalju preko interneta *frontend*-u kao tekstualni podatak

[15] Tej, Dharan (2023). *An INTRANET-Based Web Application for College Management System Using Python with Django Web Framework*. International Journal for Research in Applied Science and Engineering Technology.
[16] Schüle, M. E. (2023). *Blue Elephants Inspecting Pandas*. Strana br. 9.

(*Base64*) ili da se čuvaju na serveru i da se šalju kao *URL*. Svakako je prva solucija bolja kada se performanse i štednja memorijskog prostora uzmu u obzir.

Zaključak

U ovom radu je opisana veb aplikacija koristeći dominantno *Python* programski jezik, zajedno sa osnovnim veb tehnologijama (*HTML*, *CSS*, *JS*). Aplikacija ima svoje dobre i loše strane. Loše strane i predlozi kako ih popraviti su navedene u prethodnom poglavlju diskusije. Dobra strana aplikacije je ta što je automatizovala proces analiziranja konkretno skupa podataka o igračima Lige šampiona. Korišćenje *Python*-a za ovaj zadatak je bilo idealno rješenje iz razloga što nijedan drugi alat za analiziranje podataka ne može da uradi slično. Takođe treba navesti da ova aplikacija, ali i ovaj rad predstavljaju dobar početak u daljem istraživanju da ovu temu. Ova tema može biti od velikog značaja u daljem razvoju veb aplikacija ali i daljem razvoju analize i nauke o podacima kao discipline.

LITERATURA

1. Barnett, T.; Jain, S. (2018). *Cisco visual networking index (vni) complete forecast update, 2017–2022*. Americas/EMEAR Cisco Knowledge Network (CKN) Presentation.
2. Bioco, Joao i Rocha, Álvaro. (2019). *Web Application for Management of Scientific Conferences*. In New Knowledge in Information Systems and Technologies. Springer International Publishing.
3. Braschler, Martin (2019). *Applied Data Science*. Springer, Cham.
4. Deacon, John (2009). *Model-view-controller (mvc) architecture*.
5. Duisebekova, Kulanda (2021). *Django as secure web-framework in practice*. Вестник КазАТК.
6. Forcier, J. (2008). *Python web development with Django*. Addison-Wesley Professional.
7. Harris, C. R. (2020). *Array programming with NumPy*. Nature.
8. Hunter, J. D. (2007). *Matplotlib: A 2D graphics environment*. Computing in science & engineering.
9. McKinney, W. (2010). *Data structures for statistical computing in python*. In Proceedings of the 9th Python in Science Conference.
10. McKinney, W. (2012). *Python for data analysis: Data wrangling with Pandas, NumPy, and IPython*. O'Reilly Media, Inc.
11. Potter, Kristin (2006). *Methods for presenting statistical information: The box plot*. In VLUDS.
12. Ratner, B. (2009). *The correlation coefficient: Its values range between+ 1/− 1, or do they?* Journal of targeting, measurement and analysis for marketing.
13. Schüle, M. E. (2023). *Blue Elephants Inspecting Pandas*.
14. Sial, A. (2021). *Comparative analysis of data visualization libraries Matplotlib and Seaborn in Python*. International Journal.

15. Tej, Dharan (2023). *An INTRANET-Based Web Application for College Management System Using Python with Django Web Framework*. International Journal for Research in Applied Science and Engineering Technology.
16. Temiz, Hakan. (2022). *Recording Performances of Some File Types for Pandas Data*. European Journal of Science and Technology.
17. Tosi, S. (2009). *Matplotlib for Python developers*. Packt Publishing Ltd.
18. Turner-Trauring, Itamar (2020). *Massive memory overhead: Numbers in Python and how NumPy helps*. https://pythonspeed.com/articles/python-integers-memory/. Poslednji pristup: 02.09.2023.
19. Vamsi, K. (2021). *Visualization of Real World Enterprise Data using Python Django Framework*. IOP Conference Series: Materials Science and Engineering.
20. Vincent, W. S. (2022). *Django for Beginners: Build websites with Python and Django*. WelcomeToCode.
21. Waskom, M. L. (2021). *Seaborn: statistical data visualization*. Journal of Open Source Software.

INTERNET IZVORI:

https://docs.djangoproject.com/en/4.2/

https://pandas.pydata.org/docs/

https://matplotlib.org/stable/index.html

RIJEČI

1. **Sloboda** – nepostojanje ograničenja, moć čovjeka da radi ono što on želi
2. **Pravda** – ljudsko načelo u kome svako snosi odgovornost za svoje postupke
3. **Pravo** – nepotpuna realizacija pravde
4. **Pobunjenik** – pokretač promjene
5. **Istina** – najjače oružje
6. **Laž** – u nedostatku istine pobjednik je onaj koji se najvještije služi lažima
7. **Ograničenje** – sprečava čovjeka da uradi nešto
8. **Vrijeme** – mjesto gdje se dešava interakcija između materije
9. **Prostor** – mjesto gdje se nalazi materija
10. **Nauka** – ljudska djelatnost konstantnog eksperimentisanja kojom se dolazi do otkrića
11. **Obrazovanje** – proces intelektualnog i društvenog razvoja pojedinca
12. **Posao** – skup obaveza koje čovjek obavlja, a koje bi trebalo da ga ispunjavaju
13. **Odmor** – neophodna pauza od posla
14. **Bogatstvo** – često se odnosi na količinu materijalne svojine, mada se odnosi i na količinu duhovne i intelektualne svojine
15. **Hrabrost** – odlika ljudi koji su spremni da prevaziđu svoje strahove
16. **Motivacija** – razlog da se nešto uradi
17. **Ambicija** – eufemizam za pohlepu
18. **Ličnost** – odnosi se na sve urođene i stečene osobine čovjeka (dobre i loše)
19. **Učenje** – proces spoznavanja nečeg već otkrivenog
20. **Razmišljanje** – proces stvaranja misli i ideja, spoznavanje neotkrivenog
21. **Misao** – proizvod ljudskog mozga tokom procesa razmišljanja
22. **Ideja** – misao koja se može sprovesti u djelo
23. **Vizija** – slikoviti prikaz misli i ideja
24. **Savjest** – mehanizam koji sprečava čovjeka da čini loša djela
25. **Jezik** – najvažniji dio kulturno-istorijskog identiteta neke zajednice
26. **Govor** – artikulacija jezika

27. **Pismo** – materijalizacija govora
28. **Logika** – nauka o rješavanju problema
29. **Činjenica** – tvrdnja koju je nemoguće demantovati
30. **Zajednica** – skup ljudi koji razmišljaju na sličan način i imaju isti cilj
31. **Novac** – papir koji ima zamišljenu vrijednost
32. **Istorija** – nauka o prošlim događajima, pomaže shvatanju sadašnjih događaja
33. **Politika** – vještina vladanja
34. **Ugovor** – pismeno obećanje dvije strane da će poštovati određena pravila
35. **Dogovor** – usmeno ostvareni ugovor
36. **Tijelo** – materijalni dio čovjeka
37. **Duh** – nematerijalni dio čovjeka
38. **Računar** – ljudska kreacija koja je promijenila svijet
39. **Programer** – osoba koja daje piše instrukcije računaru, često uz šoljicu kafe
40. **Hardver** – svaki opipljivi dio računara, analogija tijela kod čovjeka
41. **Softver** – neopipljivi dio računara, analogija ljudskog duha
42. **Interpreter** – prevodilac koda koji programer napiše
43. **Kompajler** – konvertor koda u izvršni program
44. **Internet** – globalna mreža računara, i izvor ogromne količine podataka
45. **Sekvenca** – niz instrukcija koje se redom izvršavaju
46. **Selekcija** – blok instrukcija koji se izvršava ukoliko je ispunjen uslov
47. **Petlja** – blok instrukcija koje se iznova izvršavaju dok je ispunjen uslov
48. **Biblioteka** – tuđi kod koji se uvozi i upotrebljava kroz program
49. **Projekat** – proces sa definisanim početkom, krajem i ciljem
50. **Filozofija** – ljubav prema znanju
51. **Tehnologija** – faktor koji čini procese efikasnijim
52. **Moral** – nepisana pravila ponašanja koja regulišu ljudsko ponašanje
53. **Disciplina** – moć čovjeka da ne skreće sa puta kojim je krenuo
54. **Avantura** – neuobičajeno iskustvo
55. **Individualnost** – biti sam, a ne i usamljen

56. **Umjetnost** – stvaralaštvo u kome čovjek na razne načine prikazuje svoje viđenje ljudi, prirode, događaja itd.
57. **Muzika** – umjetnost koja u stvaralaštvu koristi zvuk
58. **Slikarstvo** – umjetnost koja u stvaralaštvu koristi boje
59. **Književnost** – umjetnost koja u stvaralaštvu koristi riječi
60. **Kič** – loš pokušaj stvaralaštva koji sebe naziva umjetnošću
61. **Šund** – književno djelo bez umjetničke vrijednosti, kič u oblasti književnosti
62. **Emocije** – raspon brojnih ljudskih osjećanja
63. **Sreća** – emotivno blagostanje
64. **Tuga** – dugoročno ili kratkoročno nezadovoljstvo
65. **Kritika** – negativan, ali konstruktivan stav o nečemu ili nekome
66. **Kafa** – najprihvaćenija psihostimulativna supstanca, izvor lažne energije
67. **Požrtvovanost** – najplemenitija ljudska osobina
68. **Proizvod** – materijalno dobro koje mijenjamo u zamjenu za novac
69. **Usluga** – nematerijalo dobro koje mijenjamo u zamjenu za novac
70. **Odluka** – izbor između više opcija, vrlo često ne postoji prava odluka
71. **Bilješka** – napisana misao koju razumije samo autor, a u nekim slučajevima ni autor
72. **Inteligencija** – sposobost brzog učenja
73. **Uspomena** – stvar koja budi sjećanja na neki događaj
74. **Rat** – loš i besmislen način rješavanja problema
75. **Razgovor** – prenos mišljenja i ideja između dvije ili više strana u cilju rješavanja nekog problema
76. **Kultura** – duhovna svojina jednog naroda
77. **Statistika** – ozbiljna nauka kojom se manipuliše javnim mjenjem
78. **Analiza** – proces izvlačenja zaključaka iz podataka
79. **Priroda** – prostor ne narušen čovjekovim djelovanjem
80. **Matematika** – nauka o brojevima, primijenjena logika, osnova prirodnih nauka
81. **Porodica** – zajednica ljudi u krvnom srodstvu, najčešće ima najveći uticaj u razvoju ličnosti

82. **Prijateljstvo** – veza između ljudi koji dijele isti sistem vrijednosti
83. **Kooperacija** – zajednički napor u cilju rješavanja problema
84. **Snaga** – fizička/psihička sposobnost čovjeka
85. **Usamljenost** – nedostatak društvenosti
86. **Sport** – aktivnost u kojoj se pojedinac ili tim takmiče, iskvarena pohlepom za novcem i rezultatima
87. **San** – iluzija koja se stvara tokom spavanja
88. **Pobjeda** – dokaz vrijednog rada
89. **Gubitnik** – prelazna faza između pobjednika i onog koji nikad ne pokušava
90. **Putovanje** – kratkotrajna promjena okruženja
91. **Haos** – stanje svijeta bez pravila
92. **Harmonija** – stanje apsolutnog blagostanja
93. **Utopija** – idealan svijet, svako ga zamišlja drugačije
94. **Stoicizam** – racionalnost, samokontrola i prevazilaženje prevelikog uticaja emocija
95. **Nihilizam** – obezvređivanje svega oko sebe
96. **Optimizam** – princip sagledavanja stvari u pozitivnom svjetlu
97. **Pesimizam** – princip sagledavanja samo najgoreg iz neke situacije
98. **Realizam** – sagledavanje stvari onakvim kakve jesu, niko nije 100% realan
99. **Investicija** – ulaganje u nešto što ima potencijal za uspjeh
100. **Ciklus** – proces koji se iznova ponavlja

SLIKE

Genetika. Jako zanimljiva nauka i omiljeni predmet u srednjoj školi. Izvor: link

Botanika je na drugom mjestu. Izvor: link

Jako lijep sport. Izvor: link

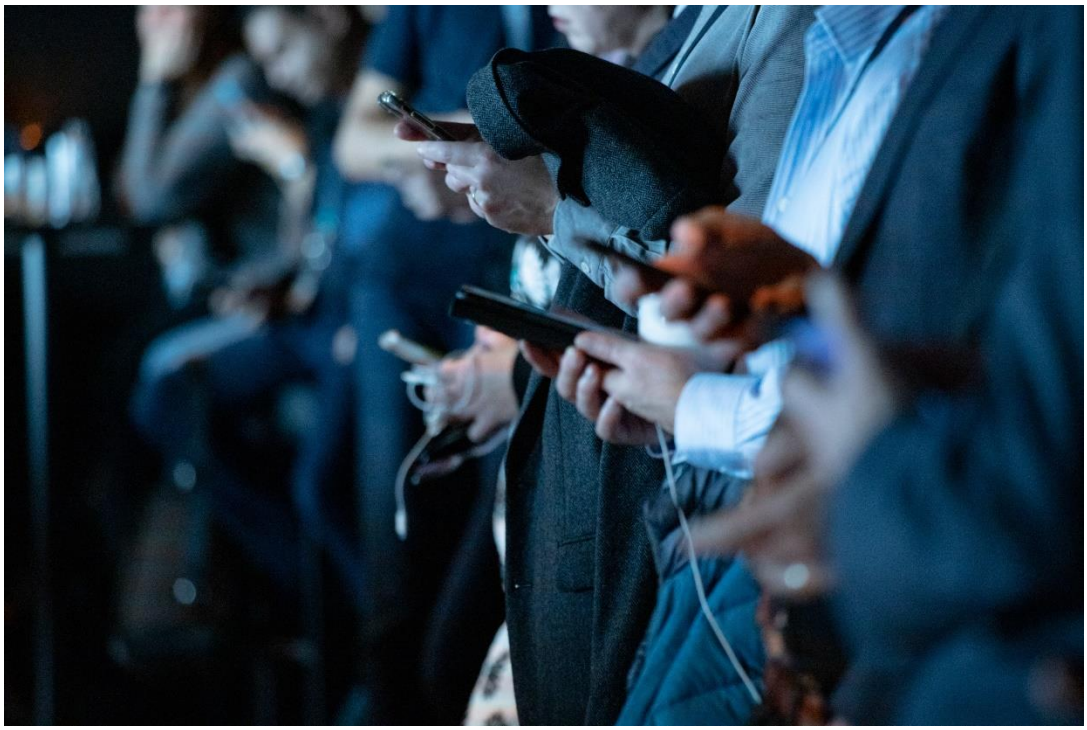
Dojč, domaća proizvodnja

Livade na Zlatiboru. Izvor: link

Spuž, Danilovgrad. Izvor: link

Azurna obala

Surogat društvenosti, društvene mreže. Izvor: link



Gvozdeni tron

Nije mačak, nego Lav

Kako nastaje softver, i kako je nastao ovaj rad. Izvor: link