SREDNJA ŠKOLA ISIDORA KRŠNJAVOGA

N A Š I C E

**PISANI DIO IZRADBE**

**ZAVRŠNOG RADA**

**Aplikacija za praćenje lokalne nogometne lige**

**šk. god. 2024./2025.**

Ime i prezime učenika: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Zanimanje: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Razred: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Predmet:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Mentor:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ odobrio:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

U Našicama, \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (potpis mentora)

**Opis zadatka:**

Napraviti aplikaciju za praćenje lokalne nogometne lige koristeći programski jezik *Python*. Aplikacija treba sadržavati sljedeće elemente:

* Unos Klubova
* Izbor ručnog i programskog generiranja sezone
* Funkciju Spremi/Učitaj
* Konstanto vidljivu tablicu poretka
* Izvoz rasporeda za sezonu i tablice poretka u PDF dokument

**Sadržaj**

[1. Uvod 1](#_Toc197675795)

[2. Korištene tehnologije 2](#_Toc197675796)

[2.1. Python 2](#_Toc197675797)

[2.2. Tkinter 2](#_Toc197675798)

[2.3. TTKBootstrap 3](#_Toc197675799)

[2.4. ReportLab 3](#_Toc197675800)

[2.5. Visual Studio Code 4](#_Toc197675801)

[3. Princip rada aplikacije 6](#_Toc197675802)

[4. Korisničko sučelje 8](#_Toc197675803)

[4.1. Planiranje korisničkog sučelja 8](#_Toc197675804)

[4.2. Kreiranje korisničkog sučelja 8](#_Toc197675805)

[5. Razvoj aplikacije 14](#_Toc197675806)

[5.1. Opće funkcije 14](#_Toc197675807)

[5.1.1. Funkcije za programsko generiranje sezone 23](#_Toc197675808)

[5.2. Funkcije za ručni unos sezone 27](#_Toc197675809)

[5.3. Funkcije s izvozom i uvozom datoteka 31](#_Toc197675810)

[6. Zaključak 37](#_Toc197675811)

[7. Literatura 38](#_Toc197675812)

# Uvod

Sportska statistika i analiza rezultata ključni su segmenti modernog upravljanja nogometnim ligama. Pristup digitalnoj evidenciji rezultata omogućuje brzu obradu podataka, poboljšava transparentnost natjecanja te potiče daljnji razvoj taktičkih analiza. Informirani korisnici i ljubitelji sporta dobiju alat koji pruža točne, pravovremene i vizualno atraktivne izvještaje.

Izrada aplikacije ostvarena je korištenjem Python programskog jezika unutar *Visual Studio Code* razvojne sredine. *Tkinte*r osigurava temeljno grafičko korisničko sučelje, dok *ttkbootstrap* donosi modernizirani izgled i proširene mogućnosti prilagodbe. Upotreba *reportlab* biblioteke omogućuje generiranje visokokvalitetnih PDF dokumenata, pružajući dodatnu vrijednost krajnjim korisnicima u prezentaciji njihovih podataka.

Sustav koristi suvremene metode programiranja koje osiguravaju modularnost i lako održavanje koda. Primjena objektnog pristupa i jasno definirana arhitektura rezultirala je agilnim i proširivim rješenjem, spremnim za buduće nadogradnje i integraciju dodatnih značajki. Svaki segment aplikacije dizajniran je tako da zadovolji potrebe korisnika u stvarnom vremenu, osiguravajući intuitivnost i pouzdanost pri radu sa sportskim statistikama.

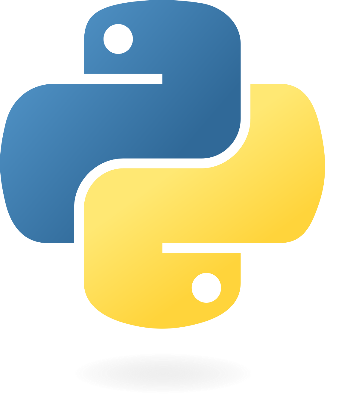
Primijenjene tehnologije izrađene su s naglaskom na interaktivnost i jednostavnost korištenja. Aplikacija ne samo da strukturira unos i obradu rezultata, već i omogućuje detaljnu analizu unesenih podataka kroz pregledne vizualizacije i dokumente. Time se postiže ravnoteža između tehničke sofisticiranosti i praktičnosti, čime se poboljšava cjelokupno korisničko iskustvo.

Ovaj dokument detaljno opisuje funkcionalnosti, arhitekturu i tehničke aspekte aplikacije.

# Korištene tehnologije

## Python

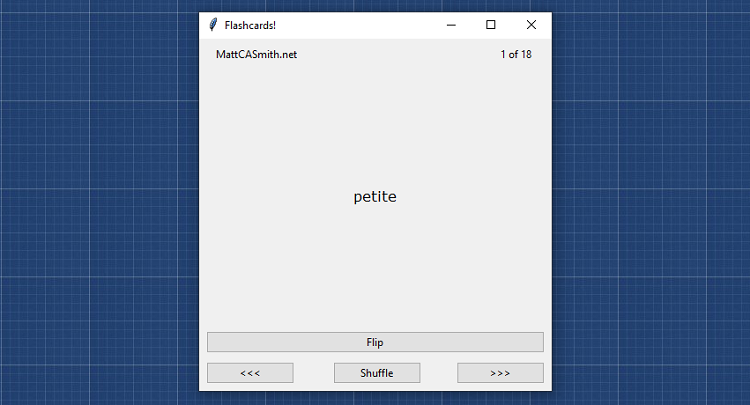
*Python* je interpretirani programski jezik izuzetno cijenjen zbog svoje jednostavne i čitljive sintakse te visoke svestranosti. *Python* omogućuje brzo koncipiranje i razvoj kompleksnih aplikacija zahvaljujući opsežnoj zbirci standardnih biblioteka i vanjskih modula. Globalna zajednica programera kontinuirano doprinosi njegovom razvoju, čime se osigurava stalna evolucija jezika i dostupnost rješenja za brojne tehničke izazove. Njegova sposobnost obrade podataka čini ga idealnim izborom za aplikacije kojima je potrebna brza analiza i prezentacija statističkih informacija. [1]



**Slika 2.1** – Logotip Python-a

## Tkinter

*Tkinter* je službena biblioteka za razvoj grafičkih korisničkih sučelja (GUI) unutar *Python* okruženja. Omogućuje izgradnju interaktivnih aplikacija kroz širok spektar *widgeta*, uključujući gumbe, okvire, tekstualna polja i kontrole popisa. Integracijom s *Tcl*/*Tk* *toolkitom* postiže se visoka pouzdanost i kompatibilnost, što je ključno za kontinuirani rad aplikacija u stvarnom vremenu. Pojednostavljena integracija unutar *Python-ove* distribucije ubrzava razvoj, dopuštajući developerima da se usmjere na dizajn i funkcionalnost sučelja bez potrebe za dodatnim konfiguracijama. [2]



**Slika 2.2** – Primjer jednostavnog korisničkog sučelja kreiranog uz pomoć Tkintera

## TTKBootstrap

*TTKBootstrap* proširuje funkcionalnosti *Tkintera* nudeći modernizirane stilove i unaprijeđene vizualne teme, inspirirane popularnim web *framework*-ovima. Ova biblioteka transformira klasične *widgete* u elegantne i estetski usklađene elemente, čime se postiže suvremeni izgled korisničkog sučelja. Pored poboljšanja vizualne privlačnosti, *TTKBootstrap* omogućuje detaljnu konfiguraciju svakog elementa, olakšavajući prilagodbu sučelja specifičnim zahtjevima krajnjih korisnika. Rezultat je intuitivan i profesionalan izgled aplikacija koje zahtijevaju uspješno upravljanje velikim količinama podataka. [3],[4]

## ReportLab

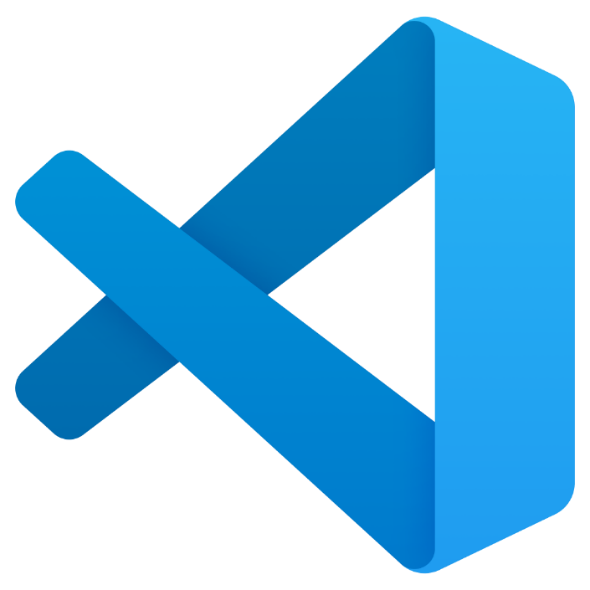
*ReportLab* je napredna biblioteka koja omogućuje dinamičko generiranje PDF dokumenata putem *Pythona*. Omogućuje kreiranje profesionalno oblikovanih izvještaja koji uključuju tekst, grafičke elemente, tablice i prilagođene dizajne. Njegova sposobnost automatiziranog generiranja izvještaja temeljenih na statističkim podacima čini ga ključnim alatom u sustavima za praćenje nogometnih rezultata. Detaljna kontrola nad izgledom i sadržajem dokumenta omogućuje prilagodbu izvještaja specifičnim potrebama korisnika, čime se osigurava visoka kvaliteta prezentacije podataka. [5]



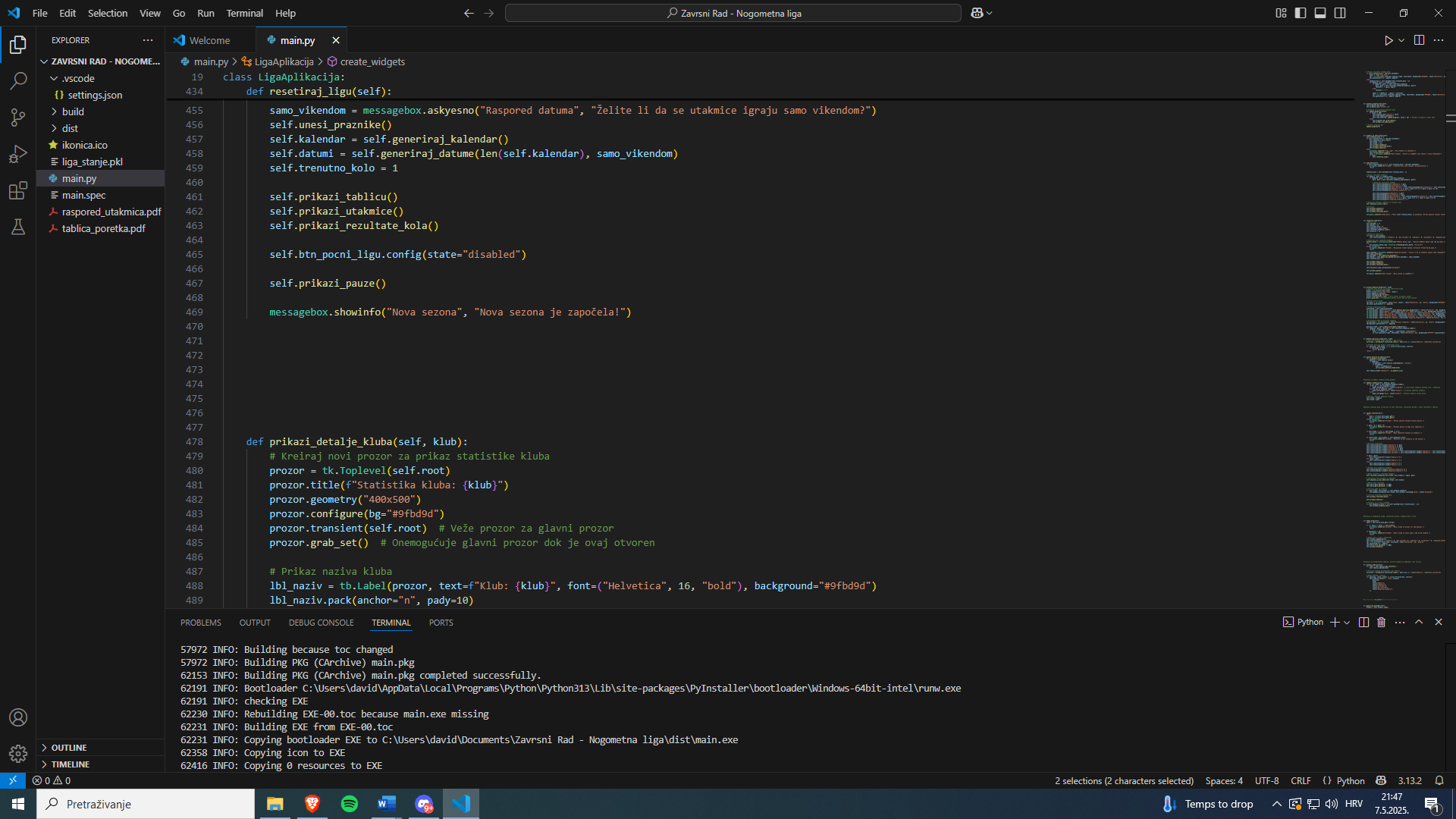
**Slika 2.3** – Logotip ReportLab-a

## Visual Studio Code

*Visual Studio Code*, predstavlja moderan editor koda i integrirano razvojno okruženje (IDE) koje potiče produktivnost kroz bogat ekosustav ekstenzija*. Visual Studio Code* je razvio *Microsoft*. Ovaj alat nudi napredne značajke poput automatskog dovršavanja koda, alata za *debugging* i integracije sustava za upravljanje verzijama, čime se optimizira razvojni tijek projekta. Posebna podrška za *Python* putem raznih dodataka omogućuje bržu i učinkovitiju izradu, testiranje i održavanje aplikacija, osiguravajući time visoku razinu kvalitete i pouzdanosti koda. Personalizacija radnog okruženja dodatno doprinosi efikasnosti i komfornosti tijekom razvoja kompleksnih softverskih rješenja. [6]



**Slika 2.4** – Logotip Visual Studio Code-a



**Slika 2.5** – Korisničko sučelje Visual Studio Code-a pri uređivanju koda

# Princip rada aplikacije

Aplikacija funkcionira po principu da korisnik prvo unese klubove koje želi imati u ligi. Nakon što korisnik unese sve klubove, započinje ligu. Pri početku lige, korisnik odabere želi li ručno odabrati utakmice i termine utakmica, ili da program generira sezonu uz parametre koje korisnik zadaje. Ukoliko korisnik izabere ručno biranje utakmice, on bira domaćina i gosta za svaku utakmicu te datum na koji se odigrava utakmica. Program samostalno određuje kraj tj. početak kola i ne dozvoljava korisniku da odabere parove za utakmicu koji nisu ostvarivi (npr. ukoliko jedan klub već igra utakmicu u tom kolu korisnik ga ne može izabrati za drugu utakmicu u tom kolu, ili je klub već ugostio sve ostale klubove te ne može više biti domaćin). Nakon što korisnik unese sve moguće utakmice po željenom rasporedu započinje unos rezultata kolo po kolo. Ukoliko je korisnik odabrao programsko generiranje utakmica, aplikacija zadaje korisniku upite kako bi se odredili parametri za ligu. Upiti uključuju početni datum sezone, odigravanje tokom tjedna ili vikenda (obzirom da su većinom niže lokalne lige igrane vikendom), broj razdoblja u kojima se liga pauzira i na kraju unos trajanje samih razdoblja (ukoliko je korisnik unio da ih ima). Nakon toga upotreba aplikacije je identična u oba slučaja (ručno ili programsko generiranje). U okviru *utakmica\_frame* se za trenutno kolo prikazuju utakmice koje su ili ručno ili programski stvorene, također prikazuju se s desne strane sučelja sve utakmice u tekućem kolu i automatski se generira PDF dokument s rasporedom svih utakmica za sezonu (koji se sprema na istoj lokaciji gdje je i aplikacija). Korisnik pritišče na gumb koje god utakmice želi te unosi rezultat utakmice u entry\_gol1 i entry\_gol2 polja te pritišče gumb „Spremi rezultat“. Aplikacija pohranjuje korisnikov unos rezultata utakmice te ažurira tablicu poretka i okvir s utakmicama (tj. rezultatima) trenutnog kola s desne strane sučelja nakon svake unesene utakmice. Nakon što se unese rezultat svake utakmice iz prvog kola, prelazi se na drugo kolo i proces se ponavlja sve dok se ne unese posljednja utakmica posljednjeg kola i završi sezona. U međuvremenu, korisnik ima mogućnost spremanja trenutnog stanja lige u bilo kojem trenutku kako bi mogao u nekom drugom trenutku učitati ligu sa svime što je on unio. Također korisnik može bilo kada pritisnuti gumb „Undo Kolo“ kako bi sve unose i upisanu statistiku iz tekućeg kola poništio ukoliko želi prepraviti svoje unose. Korisniku je omogućeno u bilom kojem trenutku izvesti raspored u PDF dokument (svakoj utakmici kojoj je unesen rezultat prije generiranja dokumenta je pripisan rezultat) i omogućeno mu je u bilo kojem trenutku generirati PDF dokument s tablicom poretka. Kada je svakoj utakmici u sezoni upisan rezultat, time završava sezona i automatski se generira PDF dokument tablice poretka. Zatim aplikacija pita korisnika želi li započeti novu sezonu s istim klubovima koji su uneseni. Ukoliko korisnik pritisne „ne“, ništa se ne događa, ukoliko pritisne „da“ proces počinje iznova od ispitivanja korisnika želi li unositi utakmice ručno ili da budu programski generirane (sačuvaju se jedino klubovi).

# Korisničko sučelje

Korisničko sučelje (UI) aplikacije razvijeno je korištenjem *Python* biblioteka *Tkinter* i *ttkbootstrap*. *Tkinter* je standardna *Python* biblioteka za izradu grafičkih sučelja, dok *ttkbootstrap* omogućuje modernizaciju izgleda i dodatne stilove za *widgete*. Sučelje je dizajnirano tako da bude intuitivno i funkcionalno, omogućujući korisnicima jednostavno upravljanje ligom, unos podataka i pregled rezultata.

## Planiranje korisničkog sučelja

Korisničko sučelje podijeljeno je u tri glavna dijela kako bi se olakšalo projektiranje sučelja, a i samo iskustvo korisnika zbog organiziranosti elemenata. Prvi dio (gornji) koristi za unos kluba i upravljačke gumbove te predstavlja „zaglavlje“ GUI-a. Drugi dio (srednji) prikazuje klubove, utakmice i kratku povijest utakmica, nalazi se u sredini obzirom da je to najvažniji dio u aplikaciji za korisnika. Treći dio (donji) iskorišten je za unos rezultata, tablicu poretka klubova te prikazivanje praznika u sezoni. UI je osmišljen tako d proces koji se odvija je vizualno prikazan od vrha prema dnu, počevši od unosa kluba (gornji dio), nastavlja se na odabir utakmice (srednji dio), zatim se unosi rezultat (donji dio) te su odmah vidljive promjene poretka koje je ta utakmica uzrokovala na samom dnu.

## Kreiranje korisničkog sučelja

Funkcija *create\_widgets* u kodu odgovorna je za inicijalizaciju i postavljanje svih elemenata korisničkog sučelja. Ova funkcija koristi različite *widgete* iz *ttkbootstrap* biblioteke kako bi se postigao moderan, jednostavna, uredan i responzivan dizajn.

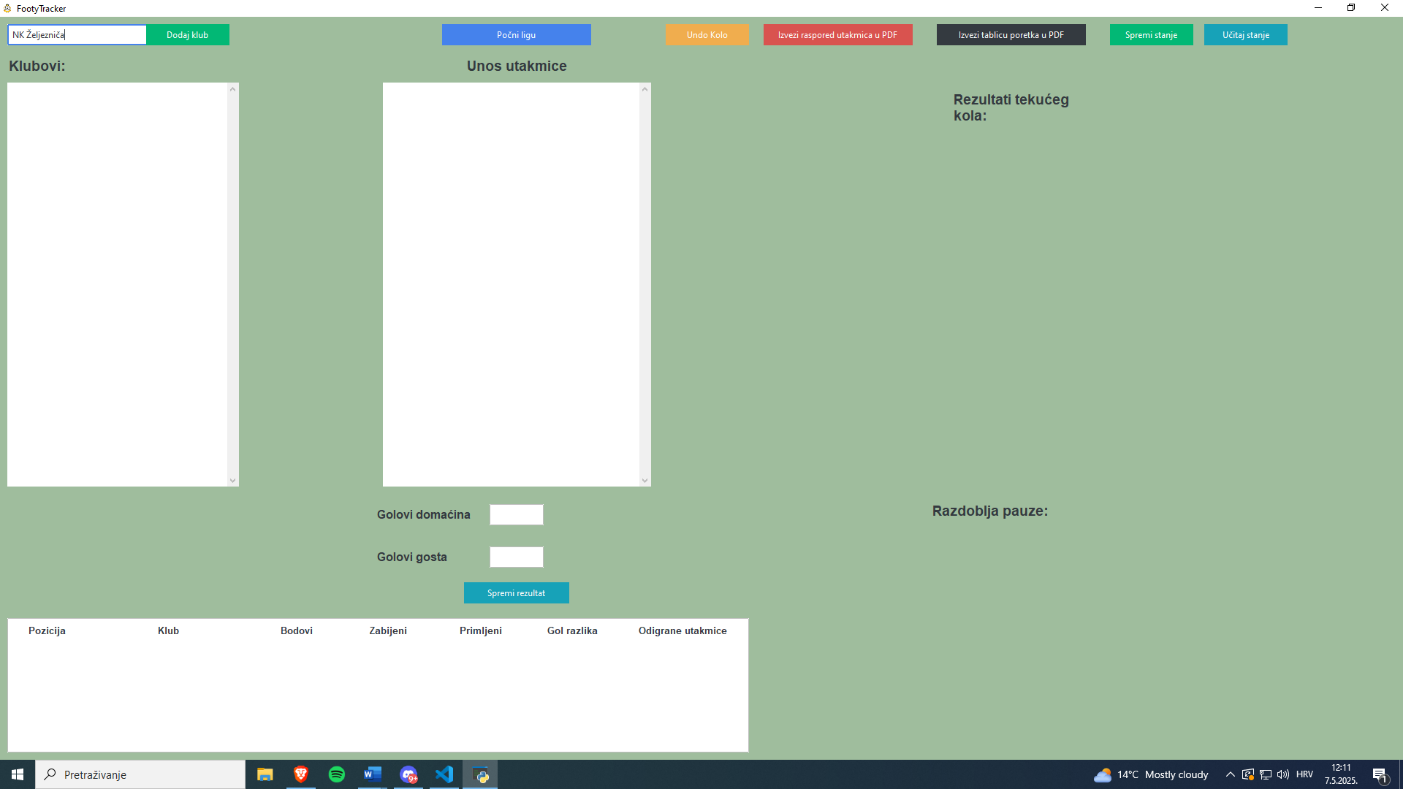
1. def create\_widgets(self):
2. style = tb.Style()
3. style.configure("Custom.TFrame", background="#9fbd9d")

**Programski kod 3.1** – create\_widgets

Na početku funkcije *create\_widgets* koristi se *ttkbootstrap.Style* za definiranje osnovnog stila aplikacije. Stilovi omogućuju prilagodbu izgleda *widgeta*. Ovdje je definiran stil za okvire *TFrame* s pozadinskom bojom #9fbd9d (heksadekadski kod za maslinastu zelenu), koja se koristi za stvaranje vizualne konzistentnosti.

1. # Gornji dio: Unos kluba i gumbovi za save itd
2. self.entry\_klub = tb.Entry(self.root, width=40)
3. self.entry\_klub.grid(row=0, column=0, padx=(10, 5), pady=10, sticky="w")
4. self.entry\_klub.bind("<**Return**>", lambda event: self.dodaj\_klub())
5. self.entry\_klub.grid\_propagate(False)
7. self.btn\_dodaj\_klub = tb.Button(self.root, text="Dodaj klub", command=self.dodaj\_klub, bootstyle="success", width=15)
8. self.btn\_dodaj\_klub.grid(row=0, column=0, padx=200, pady=10, sticky="w")
9. self.btn\_dodaj\_klub.grid\_propagate(False)
11. self.btn\_pocni\_ligu = tb.Button(self.root, text="Počni ligu", command=self.pocni\_ligu, bootstyle="primary", width=30)
12. self.btn\_pocni\_ligu.grid(row=0, column=1, padx=10, pady=10)
13. self.btn\_pocni\_ligu.grid\_propagate(False)
15. self.btn\_undo\_kolo = tb.Button(self.root, text="Undo Kolo", command=self.undo\_kolo, bootstyle="warning", width=15)
16. self.btn\_undo\_kolo.grid(row=0, column=2, padx=10, pady=10)
17. self.btn\_undo\_kolo.grid\_propagate(False)
19. self.btn\_spremi\_stanje = tb.Button(self.root, text="Spremi stanje", command=self.spremi\_stanje, bootstyle="success", width=15)
20. self.btn\_spremi\_stanje.grid(row=0, column=5, padx=10, pady=10)
21. self.btn\_spremi\_stanje.grid\_propagate(False)
23. self.btn\_ucitaj\_stanje = tb.Button(self.root, text="Učitaj stanje", command=self.ucitaj\_stanje, bootstyle="info", width=15)
24. self.btn\_ucitaj\_stanje.grid(row=0, column=6, padx=5, pady=10)
25. self.btn\_ucitaj\_stanje.grid\_propagate(False)
27. self.btn\_izvezi\_pdf = tb.Button(self.root, text="Izvezi raspored utakmica u PDF", command=self.izvezi\_raspored\_u\_pdf, bootstyle="danger", width=30)
28. self.btn\_izvezi\_pdf.grid(row=0, column=3, padx=10, pady=10)
29. self.btn\_izvezi\_pdf.grid\_propagate(False)
31. self.btn\_izvezi\_tablicu\_pdf = tb.Button(self.root, text="Izvezi tablicu poretka u PDF", command=self.izvezi\_tablicu\_u\_pdf, bootstyle="dark", width=30)
32. self.btn\_izvezi\_tablicu\_pdf.grid(row=0, column=4, padx=10, pady=10)
33. self.btn\_izvezi\_tablicu\_pdf.grid\_propagate(False)

**Programski kod 3.2** – Kreiranje gornjeg dijela sučelja

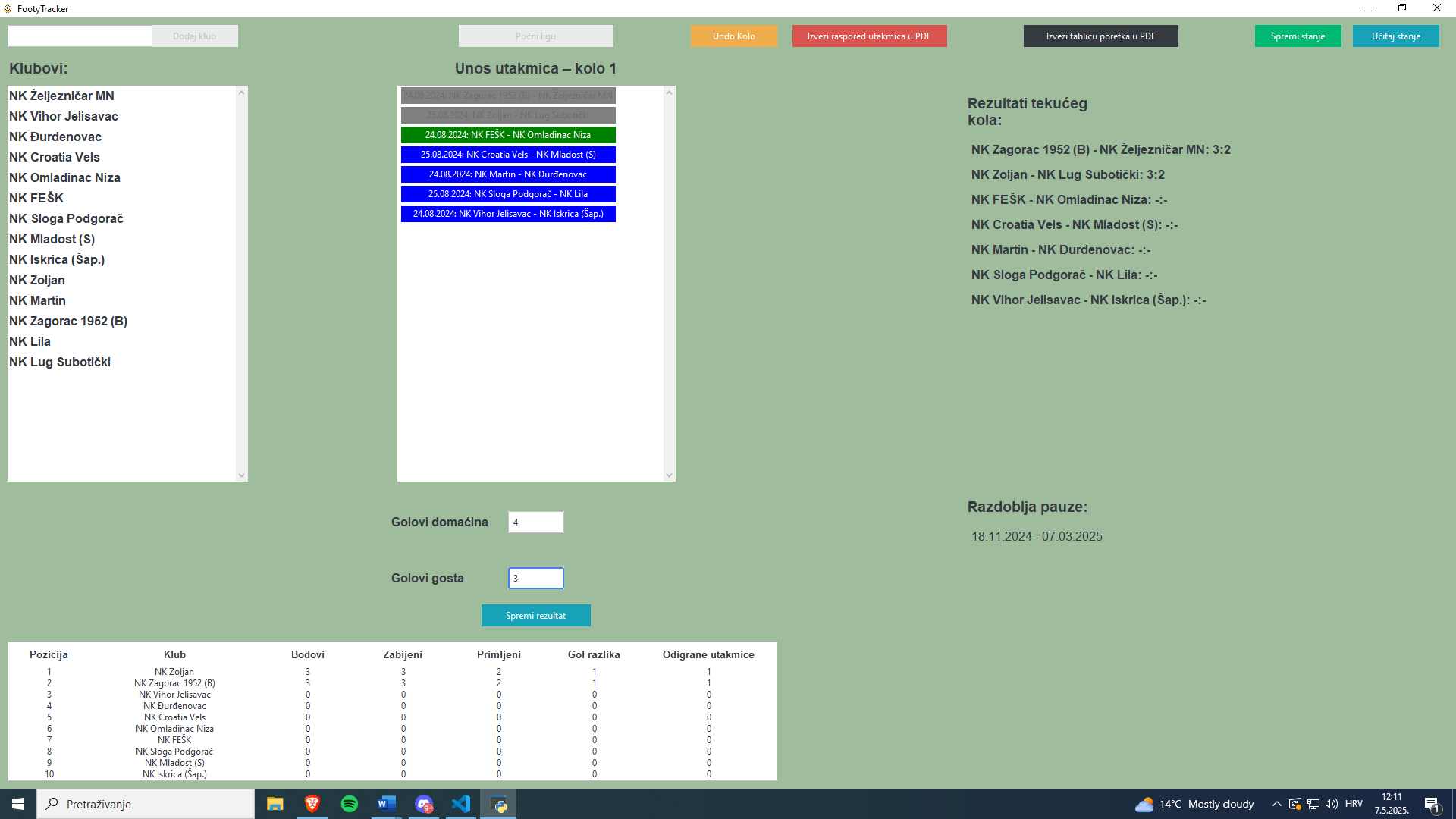
U ovom dijelu koda prikazano je kreiranje gornjeg dijela korisničkog sučelja. Element *entry\_klub* odnosi se na područje gdje se unosi naziv kluba. Na *entry\_klub* je također primijenjena funkcija „*bind*“ kako bi se olakšao unos naziva kluba. Gumb *btn\_dodaj\_klub* je gumb koji se koristi kako bi se potvrdio unos naziva kluba, njegova funkcija je ista kao tipka „*Return*“ prilikom unosa. Gumb *btn*\_*pocni*\_*ligu* koristi se za započinjanje lige te pokreće funkciju *pocni*\_*ligu*. Gumb *undo\_kolo* koristi se kako bi se pokrenula funkcija *undo*\_*kolo* koja postavi sve unesene podatke o trenutnom kolu na zadane podatke. Služi se ukoliko su uneseni krivi podaci da korisnik može s lakoćom ispraviti svoj unos. Gumbovi *btn*\_*spremi\_stanje* i *btn\_ucitaj\_stanje* koriste se za pohranu i učitavanje podataka. Ukoliko korisnik želi spremiti ligu koju trenutno koristi, može pritisnuti gumb *btn\_spremi\_stanje* kako bi pohranio sve trenutne informacije o ligi te u daljnjoj upotrebi ili prilikom gašenja aplikacije imao pohranjenu verziju svoje lige. Posljednja verzija koja se pohranila može se učitati uz pomoć gumba btn\_ucitaj\_stanje. Gumbovi *btn\_izvezi\_pdf* i *btn\_izvezi\_tablicu\_pdf* služe kao metode za izvoz rasporeda za sezonu ili za izvoz trenutnog stanja u tablici poretka. Izvoz u grafički uređen PDF format omogućuje biblioteka *reportlab*.

**Slika 3.1 –** Gornji dio korisničkog sučelja

1. self.lbl\_klubovi = tb.Label(self.root, text="Klubovi:", background="#9fbd9d", font=("Helvetica", 14, "bold"))
2. self.lbl\_klubovi.grid(row=1, column=0, padx=10, pady=5, sticky="w")
4. self.klub\_frame\_container = tb.Frame(self.root, height=500, width=300)
5. self.klub\_frame\_container.grid(row=2, column=0, padx=10, pady=5, sticky="nsw")
6. self.klub\_frame\_container.grid\_propagate(False)
7. self.klub\_frame = self.create\_scrollable\_frame(self.klub\_frame\_container)
9. self.lbl\_utakmica = tb.Label(self.root, text="Unos utakmice", background="#9fbd9d", font=("Helvetica", 14, "bold"))
10. self.lbl\_utakmica.grid(row=1, column=1, columnspan=1, pady=5)
12. self.utakmica\_frame\_container = tb.Frame(self.root, height=300, width=350)
13. self.utakmica\_frame\_container.grid(row=2, column=1, columnspan=1, padx=10, pady=5, sticky="ns")
14. self.utakmica\_frame\_container.grid\_propagate(False)
15. self.utakmica\_frame = self.create\_scrollable\_frame(self.utakmica\_frame\_container)
17. # Okvir za prikaz rezultata tekućeg kola
18. self.rezultati\_frame = tb.Frame(self.root, style="Custom.TFrame", width=200, height=200)
19. self.rezultati\_frame.grid(row=2, column=4, padx=10, pady=10, sticky="ns")
20. self.rezultati\_frame.grid\_propagate(False)
22. self.lbl\_rezultati = tb.Label(self.rezultati\_frame, text="Rezultati tekućeg kola:", background="#9fbd9d", font=("Helvetica", 14, "bold"), wraplength=200)
23. self.lbl\_rezultati.pack(anchor="nw", padx=5, pady=5)
24. self.lbl\_rezultati\_lista = []
26. # Pod-okvir za dinamički sadržaj rezultata
27. self.rezultati\_sadrzaj\_frame = tb.Frame(self.rezultati\_frame, style="Custom.TFrame")
28. self.rezultati\_sadrzaj\_frame.pack(fill="both", expand=True, padx=5, pady=5)

**Programski kod 3.3 –** Kreiranje srednjeg dijela sučelja

*Label lbl\_klubovi* odnosi se na fiksirani naslovni tekst koji je iznad okvira za klubove. *Okvir klub\_frame* koristi kao okvir koji sadrži popis unesenih klubova. *Label lbl\_utakmica* je također kao i *lbl\_klubovi* naslovni tekst, nalazi se iznad okvira koji sadrži popis utakmica za trenutno kolo. Okvir *utakmica\_frame* sadržava popis utakmica u trenutnom kolu u formi gumbova. *Rezultati\_frame* je okvir koji sadržava sve utakmice iz tekućeg kola te njihove rezultate. Ima pripadajući pod-okvir *rezultati\_sadrzaj\_frame* koji služi za dinamički sadržaj kako bi se kvalitetnije prikazali svi rezultati, neovisno o duljini teksta koji je promjenjiv kroz uporabu.



**Slika 3.2** – Središnji dio sučelja

1. self.entry\_gol1 = tb.Entry(self.root, width=10)
2. self.entry\_gol1.grid(row=3, column=1, padx=20, pady=10)
4. self.lbl\_goldomacina = tb.Label(self.root, text="Golovi domaćina", background="#9fbd9d", font=("Helvetica", 12, "bold"))
5. self.lbl\_goldomacina.grid(row=3, column=1, columnspan=2, padx=0, pady=5, sticky="w")



10. self.entry\_gol2 = tb.Entry(self.root, width=10)
11. self.entry\_gol2.grid(row=4, column=1, padx=0, pady=10)
13. self.lbl\_golgosta = tb.Label(self.root, text="Golovi gosta", background="#9fbd9d", font=("Helvetica", 12, "bold"))
14. self.lbl\_golgosta.grid(row=4, column=1, columnspan=2, padx=0, pady=5, sticky="w")


18. self.btn\_spremi = tb.Button(self.root, text="Spremi rezultat", command=self.spremi\_rezultat, bootstyle="info", width=20)
19. self.btn\_spremi.grid(row=5, column=1, columnspan=1, padx=0, pady=10)
21. # Okvir za prikaz razdoblja pauze
22. self.pauze\_frame = tb.Frame(self.root, style="Custom.TFrame", width=200, height=150)
23. self.pauze\_frame.grid(row=3, column=4, padx=10, pady=10, sticky="nsew")
24. self.pauze\_frame.grid\_propagate(False)
26. self.lbl\_pauze = tb.Label(self.pauze\_frame, text="Razdoblja pauze:", background="#9fbd9d", font=("Helvetica", 14, "bold"))
27. self.lbl\_pauze.pack(anchor="nw", padx=5, pady=5)
29. self.lbl\_pauze\_lista = []
31. # Pod-okvir za dinamički sadržaj pauza
32. self.pauze\_sadrzaj\_frame = tb.Frame(self.pauze\_frame, style="Custom.TFrame")
33. self.pauze\_sadrzaj\_frame.pack(fill="both", expand=True, padx=5, pady=5)

36. # Progress bar za prelazak na iduće kolo
37. self.progress\_bar = ttk.Progressbar(self.root, orient="horizontal", length=250, mode="determinate")
38. self.progress\_bar.grid(row=5, column=4, padx=10, pady=10, sticky="w")
39. self.progress\_bar.grid\_remove()  # Sakrij progress bar dok nije potreban

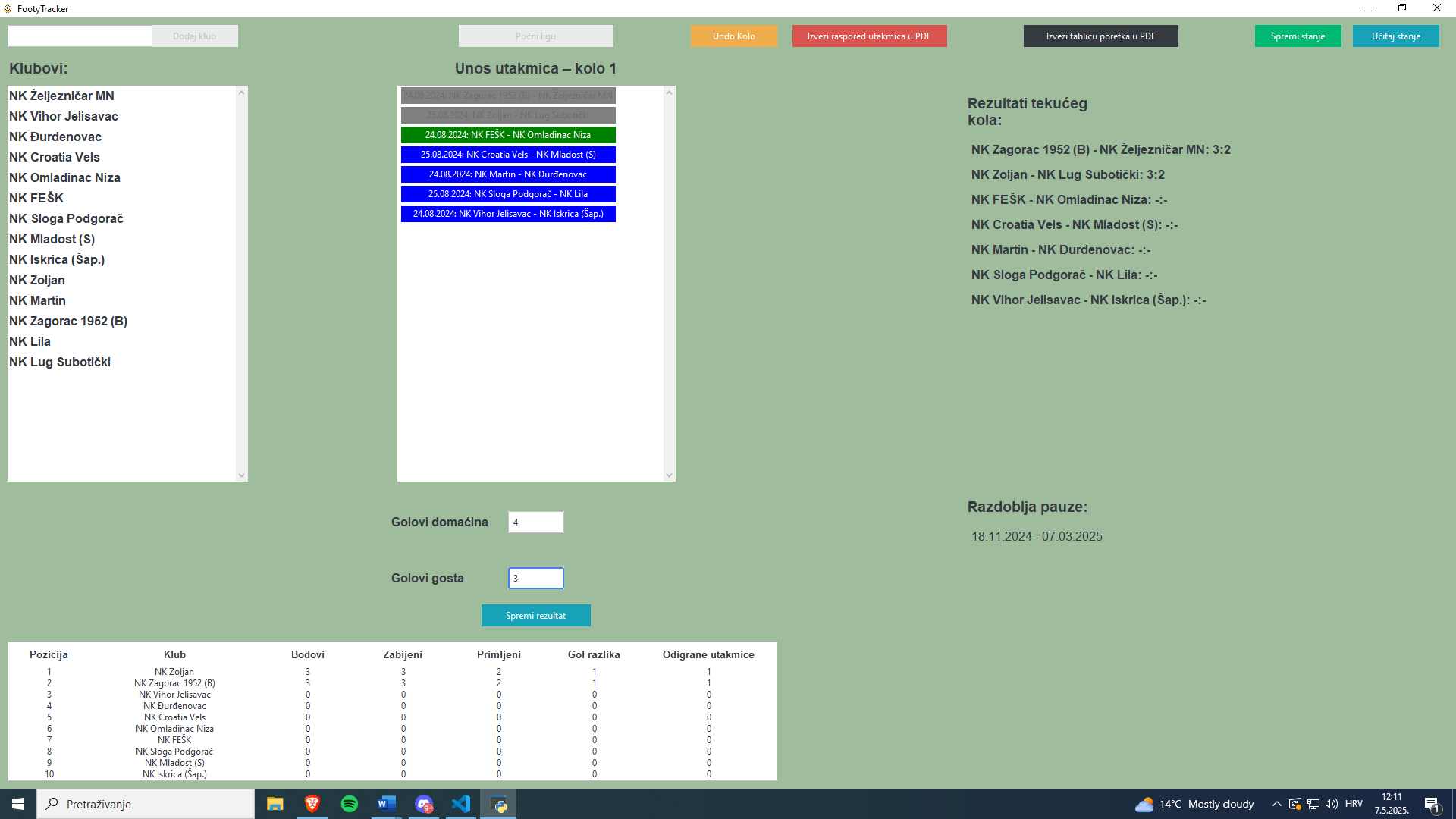


44. # Scrollable frame - tablica za poredak
45. self.tablica\_frame\_container = tb.Frame(self.root, height=200, width=500)
46. self.tablica\_frame\_container.grid(row=6, column=0, columnspan=3, padx=10, pady=10, sticky="nsew")
47. self.tablica\_frame\_container.grid\_propagate(False)
49. # Treeview za tablicu lige preko ttk
50. styletablica = ttk.Style()
51. styletablica.configure("Treeview.Heading", font=("Helvetica", 10, "bold"))

54. self.tablica = ttk.Treeview(self.tablica\_frame\_container, columns=("Pozicija", "Klub", "Bodovi", "Zabijeni", "Primljeni", "Gol razlika", "Odigrane Utakmice"), show="headings", height=10)
55. self.tablica.heading("Pozicija", text="Pozicija")
56. self.tablica.heading("Klub", text="Klub")
57. self.tablica.heading("Bodovi", text="Bodovi")
58. self.tablica.heading("Zabijeni", text="Zabijeni")
59. self.tablica.heading("Primljeni", text="Primljeni")
60. self.tablica.heading("Gol razlika", text="Gol razlika")
61. self.tablica.heading("Odigrane Utakmice", text="Odigrane utakmice")

**Programski kod 3.4** – Donji dio sučelja i tablica

Polja za unos teksta *entry\_gol1* i *entry\_gol2* primaju unos broja od korisnika koji se odnosi na broj golova*. Labeli lbl\_goldomacina* i *lbl\_golgosta* služe za raspoznavanje prethodna dva polja za unos. Gumb *btn\_spremi* služi za pohranu unosa golova domaćina i gosta u utakmici. Okvir *pauze\_frame* je okvir u kojem se prikazuju razdoblja pauze u sezoni, utakmice se ne odigravaju u tom razdoblju. Pod-okvir *pauze\_sadrzaj\_frame* je okvir unutar okvira *pauze\_frame* koji se koristi za dinamički sadržaj unutar većeg okvira kako bi se ispravno prikazale sve pauze koje je korisnik unio, bez obzira na količinu pauza. *Progress\_bar* je simulirana traka napretka koja je ostvarena uz pomoć *ttkbootstrap*-a. U samoj funkciji aplikacije nema ulogu, nego pruža korisniku par trenutaka da se osvrne na uneseno kolo. Na kraju koda koji konstruira glavni zaslon korisničkog sučelja, kreira se okvir s tablicom. Tablica je kreirana uz pomoć biblioteke *ttkbootstrap* te je organizirana i pregledna pri upotrebi aplikacije.



**Slika 3.3** – Donji dio sučelja

# Razvoj aplikacije

U ovoj cjelini dokumentacije obuhvaćaju se sve bitne funkcije koje objedinjuju aplikaciju. Cjelina je segmentirana na četiri dijela, prvi dio opisuje opće funkcije koje se globalno koriste u cijeloj aplikaciji, tj. koje su i najbitnije. Drugi dio opisuje funkcije koje se koriste prilikom programskog generiranja sezone, treći dio u kontrastu drugim opisuje one koje se koriste kada korisnik ručno unosi raspored. U posljednjem dijelu se obuhvaćaju funkcije vezane uz izvoz PDF dokumenta i funkcije za spremanje i učitavanje stanja lige.

## Opće funkcije

1. def pocni\_ligu(self):
2. if len(self.klubovi) < **2:**
3. messagebox.showerror("Greška", "Potrebna su barem 2 kluba za pokretanje lige.")
4. return
6. if len(self.klubovi) % 2 != 0:
7. messagebox.showerror("Greška", "Ne može biti neparan broj klubova.")
8. return
10. izbor = messagebox.askyesno("Raspored utakmica", "Želite li ručno unijeti raspored utakmica? Ukoliko ne, program će sam generirati utakmice za Vas.")
11. if izbor:
12. self.unesi\_raspored()
13. else:
14. pocetni\_datum = simpledialog.askstring("Početni datum lige", "Unesite početni datum lige (dd.mm.yyyy):")
15. try:
16. self.pocetni\_datum\_lige = datetime.strptime(pocetni\_datum, "%d.%m.%Y")
17. except ValueError:
18. messagebox.showerror("Greška", "Neispravan format datuma. Koristite format dd.mm.yyyy.")
19. return
21. samo\_vikendom = messagebox.askyesno("Raspored datuma", "Želite li da se utakmice igraju samo vikendom?")

24. self.unesi\_praznike()
26. self.kalendar = self.generiraj\_kalendar()
27. self.datumi = self.generiraj\_datume(len(self.kalendar), samo\_vikendom)
28. self.trenutno\_kolo = 1
29. self.prikazi\_utakmice()
31. self.entry\_klub.config(state="disabled")
32. self.btn\_dodaj\_klub.config(state="disabled")
33. self.btn\_pocni\_ligu.config(state="disabled")
34. self.izvezi\_raspored\_u\_pdf()

**Programski kod 5.1** – Funkcija počni ligu

Funkcija *počni\_ligu* pokreće ligu nakon što korisnik unese sve klubove. Generira raspored utakmica i datume te onemogućuje daljnji unos klubova. Prva dva uvjeta provjeravaju ima li barem dva unesena kluba i je li broj klubova paran. Zatim daje upit korisniku želi li ručno unositi utakmice ili da se utakmice programski generiraju. Logika oba načina generiranja utakmica objašnjena je u prethodnom poglavlju. Funkcija poziva funkcije *unesi\_raspored* ukoliko je korisnik odabrao ručni unos tj. funkcije *unesi\_praznike*, *generiraj\_kalendar* i *generiraj\_datume* ako je korisnik odabrao programsko generiranje.

1. def prikazi\_utakmice(self):
2. for widget in self.utakmica\_frame.winfo\_children():
3. widget.destroy()
4. self.gumbovi\_utakmica.clear()
6. # Dohvati utakmice i datume za trenutno kolo
7. if self.trenutno\_kolo > len(self.kalendar):
8. return
10. utakmice\_kola = self.kalendar[self.trenutno\_kolo - 1]
11. datumi\_kola = self.datumi[self.trenutno\_kolo - 1]
13. # Prikaz utakmica
14. for (domacin, gost), datum in zip(utakmice\_kola, datumi\_kola):
15. gumb\_text = f"{datum}: {domacin} - {gost}"
16. if (domacin, gost) in self.odigrano\_u\_kolu:
17. gumb = tk.Button(self.utakmica\_frame, text=gumb\_text, state="disabled", bg="gray")
18. else:
19. gumb = tk.Button(self.utakmica\_frame, text=gumb\_text,
20. command=lambda d=domacin, g=gost: self.odaberi\_utakmicu(d, g))

23. gumb.pack(fill="x", padx=5, pady=2)
24. self.gumbovi\_utakmica[(domacin, gost)] = gumb
26. # Ažuriraj naslov za trenutno kolo
27. self.lbl\_utakmica.config(text=f"Unos utakmica – kolo {self.trenutno\_kolo}")
28. def odaberi\_utakmicu(self, domacin, gost):
29. for (d, g), gumb in self.gumbovi\_utakmica.items():
30. if (d, g) in self.odigrano\_u\_kolu:
31. gumb.config(bg="gray", state="disabled")  # Iskoristene utakmice postaju sive i neaktivne
32. elif (d, g) == (domacin, gost):
33. gumb.config(bg="green", state="normal")  # Trenutno odabrana utakmica
34. else:
35. gumb.config(bg="blue", state="normal")  # Ostale utakmice ostaju plave
37. # Postavi trenutno odabrane klubove
38. self.klub1 = domacin
39. self.klub2 = gost

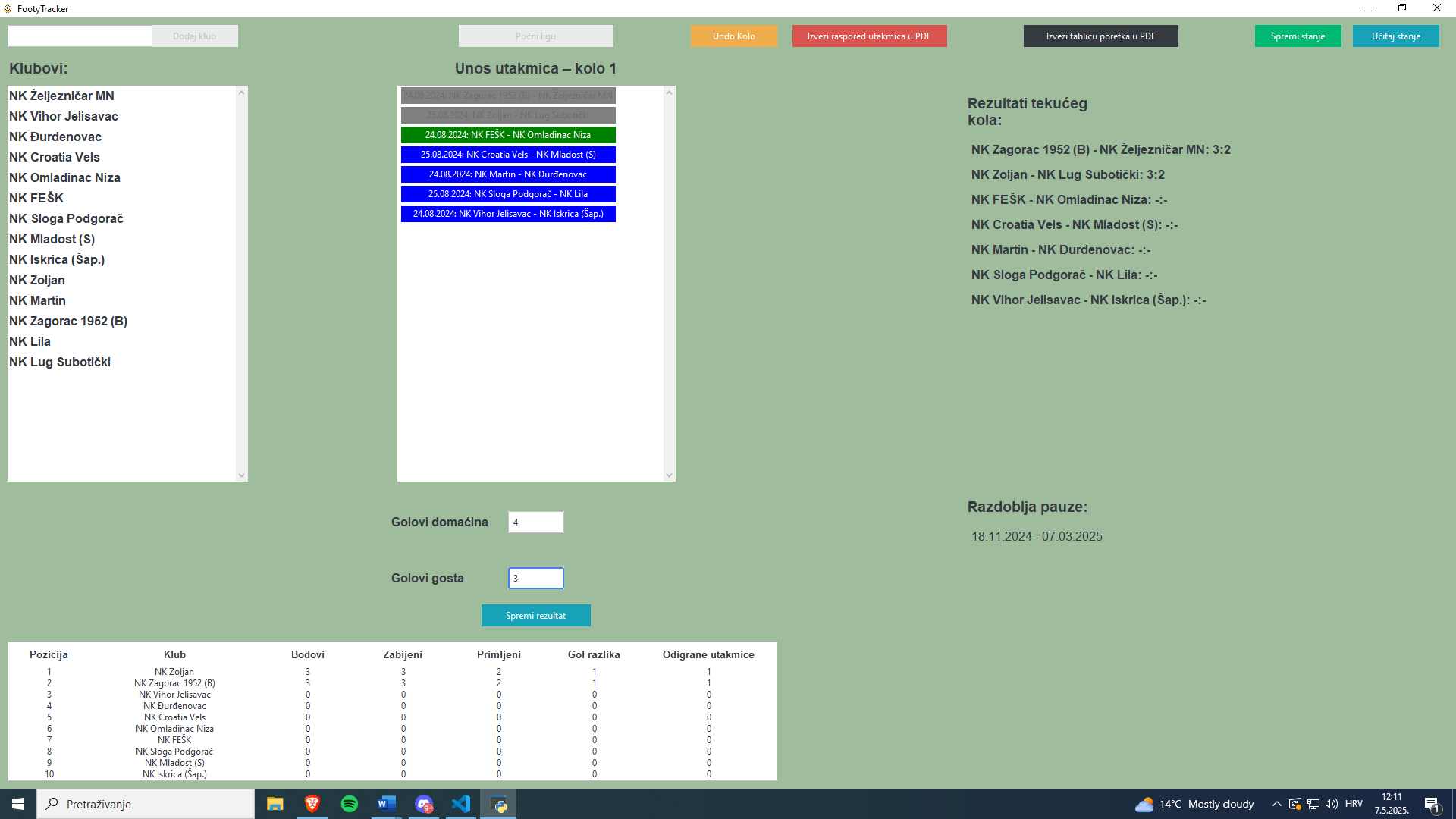
**Programski kodovi 5.2 i 5.3 –** Funkcije prikazi\_utakmice i odaberi\_utakmicu

Funkcija *prikazi\_utakmice* prikazuje utakmice trenutnog kola u formi gumbova u okviru *utakmica\_frame* kako bi korisnik jednostavno unio rezultat za svaku utakmicu. Gumbovi se prikazuju u formatu „*dd.mm.gggg*: domaćin – gost“. Ukoliko se rezultat za neku utakmicu već unio, tada gumb postaje siv i onemogućen, a kada korisnik pritisne gumb za neku utakmicu taj gumb utakmice koji je trenutno aktivan mijenja boju u zelenu kako bi se poboljšalo korisničko iskustvo. Svi ostali gumbovi su plave boje

1. def prikazi\_pauze(self):
2. for widget in self.pauze\_sadrzaj\_frame.winfo\_children():
3. widget.destroy()
5. # Prikaz razdoblja pauze
6. if not self.praznici:
7. tekst = "Nema unesenih pauza."
8. else:
9. tekst = "\n".join([f"{pocetni.strftime('%d.%m.%Y')} - {zavrsni.strftime('%d.%m.%Y')}" for pocetni, zavrsni in self.praznici])
11. lbl = tb.Label(self.pauze\_sadrzaj\_frame, text=tekst, background="#9fbd9d", font=("Helvetica", 12), justify="left")
12. lbl.pack(anchor="nw", padx=5, pady=5)

15. def prikazi\_rezultate\_kola(self):
16. for widget in self.rezultati\_sadrzaj\_frame.winfo\_children():
17. widget.destroy()
19. # Prikaz rezultata tekućeg kola
20. if self.trenutno\_kolo > len(self.kalendar):
21. tekst = "Nema više kola."
22. lbl = tb.Label(self.rezultati\_sadrzaj\_frame, text=tekst, background="#9fbd9d", font=("Helvetica", 12), justify="left")
23. lbl.pack(anchor="nw", padx=5, pady=5)
24. else:
25. utakmice\_kola = self.kalendar[self.trenutno\_kolo - 1]
26. for domacin, gost in utakmice\_kola:
27. if (domacin, gost) in self.rezultati\_utakmica:
28. gol1, gol2 = self.rezultati\_utakmica[(domacin, gost)]
29. rezultat = f"{gol1}:{gol2}"
30. else:
31. rezultat = "-:-"
33. tekst = f"{domacin} - {gost}: {rezultat}"
34. lbl = tb.Label(self.rezultati\_sadrzaj\_frame, text=tekst, background="#9fbd9d", font=("Helvetica", 12, "bold"), justify="left")
35. lbl.pack(anchor="nw", padx=5, pady=5)

**Programski kod 5.4 –** Funkcije prikazi\_pauze i prikazi\_rezultate\_kola

Funkcija *prikazi\_pauze* se upotrebljiva ukoliko se korisnik odlučio za programsko generiranje rasporeda za utakmicu. Ako je korisnik unio da nema praznika, unutar okvira *pauze\_sadrzaj\_frame* se ispisuje „Nema unesenih pauza“ u suprotnom, svaka pauza se zapisuje u zasebni red te njen početni i završni datum. Funkcija *prikazi\_rezultate\_kola* prikazuje utakmice iz trenutnog kola i njihove pripadajuće rezultate ukoliko su uneseni. Utakmice su prikazane istim redoslijedom kao i utakmice u *utakmica\_frame* tj. ona koja je pri vrhu u *utakmica\_frame* biti će pri vrhu i u *rezultati\_sadrzaj\_frame*.

**Slika 5.1 –** Djelovanje funkcija prikazi\_pauze i prikazi\_rezultate kola

1. def create\_scrollable\_frame(self, parent):
2. canvas = tk.Canvas(parent, width=parent["width"])  # Postavi širinu Canvas-a
3. scrollbar = tk.Scrollbar(parent, orient="vertical", command=canvas.yview)
4. scrollable\_frame = tk.Frame(canvas)
6. scrollable\_frame.bind(
7. "<Configure>",
8. lambda e: canvas.configure(scrollregion=canvas.bbox("all"))
9. )
11. canvas.create\_window((0, 0), window=scrollable\_frame, anchor="nw")
12. canvas.configure(yscrollcommand=scrollbar.set)
14. canvas.pack(side="left", fill="both", expand=True)
15. scrollbar.pack(side="right", fill="y")
17. return scrollable\_frame

**Programski kod 5.5 –** Funkcija create\_scrollable\_frame

Ova funkcija se koristi nekoliko puta kroz cijeli kod. Koristi se kako bi se pojednostavilo stvaranje okvira čiji sadržaj korisnik može pomicati.

1. def undo\_kolo(self):
2. if self.trenutno\_kolo <= 0 or self.trenutno\_kolo > len(self.kalendar):
3. messagebox.showerror("Greška", "Trenutno kolo nije valjano za poništavanje.")
4. return

7. utakmice\_kola = self.kalendar[self.trenutno\_kolo - 1]
9. # Ukloni rezultate utakmica
10. for domacin, gost in utakmice\_kola:
11. if (domacin, gost) in self.rezultati\_utakmica:
12. gol1, gol2 = self.rezultati\_utakmica.pop((domacin, gost))
14. # Resetiraj statistiku klubova
15. self.statistika[domacin]["zabijeni"] -= gol1
16. self.statistika[domacin]["primljeni"] -= gol2
17. self.statistika[domacin]["gol\_razlika"] = self.statistika[domacin]["zabijeni"] - self.statistika[domacin]["primljeni"]
18. self.statistika[domacin]["bodovi"] -= 3 if gol1 > gol2 else (1 if gol1 == gol2 else 0)
19. self.statistika[domacin]["odigrane\_utakmice"] -= 1
21. self.statistika[gost]["zabijeni"] -= gol2
22. self.statistika[gost]["primljeni"] -= gol1
23. self.statistika[gost]["gol\_razlika"] = self.statistika[gost]["zabijeni"] - self.statistika[gost]["primljeni"]
24. self.statistika[gost]["bodovi"] -= 3 if gol2 > gol1 else (1 if gol1 == gol2 else 0)
25. self.statistika[gost]["odigrane\_utakmice"] -= 1
27. # Resetiraj odigrane utakmice za trenutno kolo
28. self.odigrano\_u\_kolu.clear()
30. # Ažuriraj
31. self.prikazi\_utakmice()
32. self.prikazi\_tablicu()
33. self.prikazi\_rezultate\_kola()
35. messagebox.showinfo("Undo Kolo", f"Kolo {self.trenutno\_kolo} je poništeno. Možete ponovno unijeti rezultate.")

**Programski kod 5.6** – Funkcija undo\_kolo

Funkcija *undo\_kolo* i njen pripadajući gumb na korisničkom sučelju omogućuju korisniku da poništi sve izmjene koje su se dogodile prilikom tekućeg kola, što korisniku daje višu razinu komfornosti prilikom korištenja aplikacije. Prvo se uklanjaju svi upisani rezultati u utakmicama, zatim se resetiraju sve promjene u statistici i poretku koje su se dogodile te se omogućuju svi gumbovi za utakmice i na kraju se ažurira korisničko sučelje.

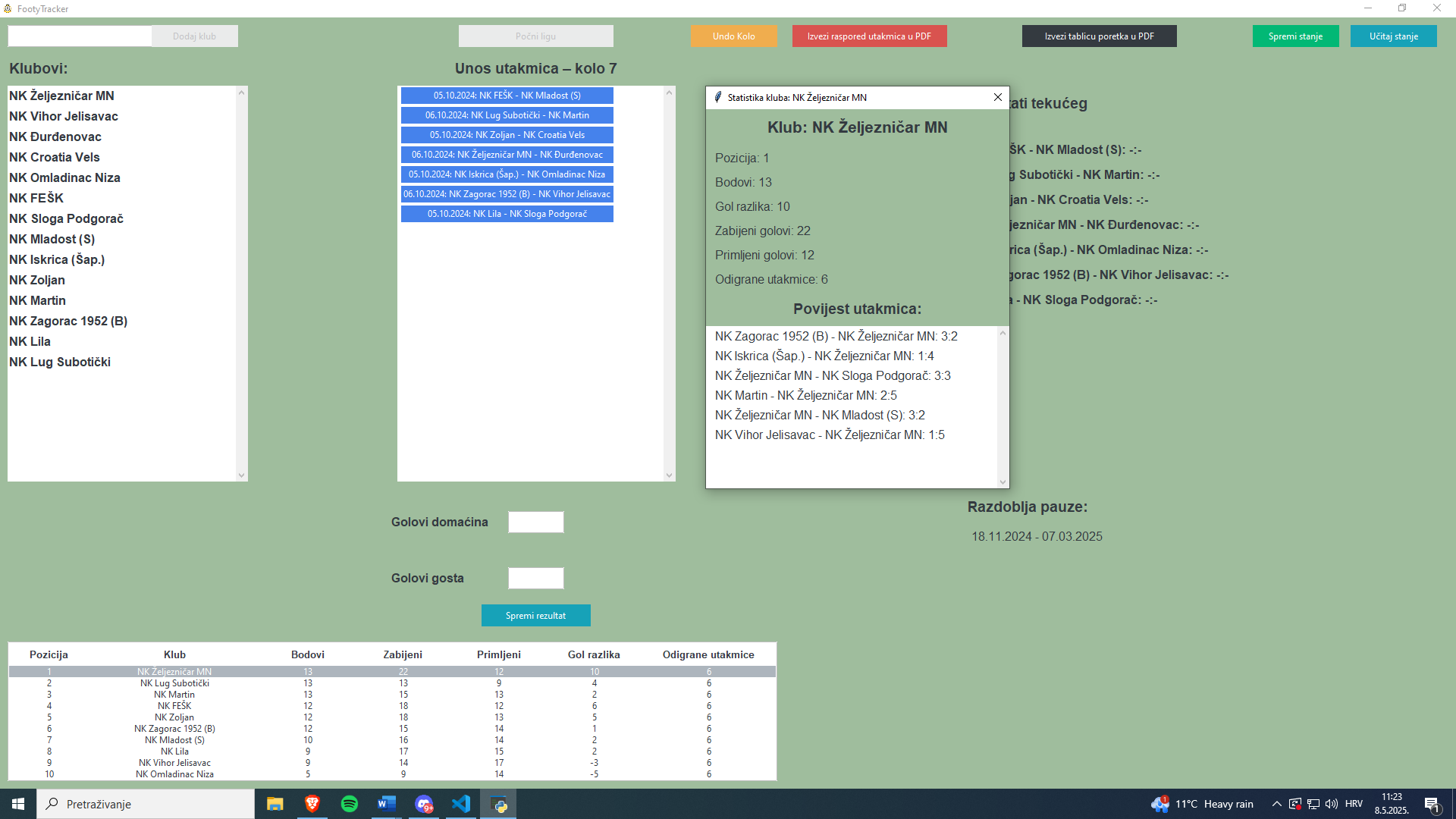
1. def resetiraj\_ligu(self):
2. # Očisti sve
3. self.kalendar = []
4. self.datumi = []
5. self.trenutno\_kolo = 0
6. self.odigrano\_u\_kolu.clear()
7. self.rezultati\_utakmica.clear()
8. self.praznici = []
10. # Resetiraj statistiku
11. for klub in self.klubovi:
12. self.statistika[klub] = {"bodovi": 0, "gol\_razlika": 0, "zabijeni": 0, "primljeni": 0, "odigrane\_utakmice": 0}
14. messagebox.showinfo("Nova sezona", "Nova sezona je započela!")
16. self.pocni\_ligu()

**Programski kod 5.7** – funkcija resetiraj\_ligu

Ova funkcija ne igra veliku ulogu u radu aplikacije, ali pruža korisniku mogućnost da započne novu sezonu bez potrebe da se ponovo unose klubovi – poboljšavajući korisničko iskustvo.

1. def prikazi\_detalje\_kluba(self, klub):
2. # Kreiraj novi prozor za prikaz statistike kluba
3. prozor = tk.Toplevel(self.root)
4. prozor.title(f"Statistika kluba: {klub}")
5. prozor.geometry("400x500")
6. prozor.configure(bg="#9fbd9d")
7. prozor.transient(self.root)  # Veže prozor za glavni prozor
8. prozor.grab\_set()  # Onemogućuje glavni prozor dok je ovaj otvoren
10. # Prikaz naziva kluba
11. lbl\_naziv = tb.Label(prozor, text=f"Klub: {klub}", font=("Helvetica", 16, "bold"), background="#9fbd9d")
12. lbl\_naziv.pack(anchor="n", pady=10)
14. # Prikaz statistike kluba
15. statistika = self.statistika[klub]
16. tb.Label(prozor, text=f"Pozicija: {self.dohvati\_poziciju\_kluba(klub)}", font=("Helvetica", 12), background="#9fbd9d").pack(anchor="w", padx=10, pady=5)
17. tb.Label(prozor, text=f"Bodovi: {statistika['bodovi']}", font=("Helvetica", 12), background="#9fbd9d").pack(anchor="w", padx=10, pady=5)
18. tb.Label(prozor, text=f"Gol razlika: {statistika['gol\_razlika']}", font=("Helvetica", 12), background="#9fbd9d").pack(anchor="w", padx=10, pady=5)
19. tb.Label(prozor, text=f"Zabijeni golovi: {statistika['zabijeni']}", font=("Helvetica", 12), background="#9fbd9d").pack(anchor="w", padx=10, pady=5)
20. tb.Label(prozor, text=f"Primljeni golovi: {statistika['primljeni']}", font=("Helvetica", 12), background="#9fbd9d").pack(anchor="w", padx=10, pady=5)
21. tb.Label(prozor, text=f"Odigrane utakmice: {statistika['odigrane\_utakmice']}", font=("Helvetica", 12), background="#9fbd9d").pack(anchor="w", padx=10, pady=5)
23. # Scrollable frame za povijest utakmica
24. lbl\_povijest = tb.Label(prozor, text="Povijest utakmica:", font=("Helvetica", 14, "bold"), background="#9fbd9d")
25. lbl\_povijest.pack(anchor="n", pady=10)
27. povijest\_frame = self.create\_scrollable\_frame(prozor)
28. for (domacin, gost), rezultat in self.rezultati\_utakmica.items():
29. if klub in (domacin, gost):
30. tekst = f"{domacin} - {gost}: {rezultat[0]}:{rezultat[1]}"
31. tb.Label(povijest\_frame, text=tekst, font=("Helvetica", 12), background="#FFFFFF").pack(anchor="w", padx=10, pady=2)
32. def povezi\_dvoklik\_na\_tablicu(self):
33. def on\_double\_click(event):
34. odabrani = self.tablica.focus()
35. if odabrani:
36. vrijednosti = self.tablica.item(odabrani, "values")
37. if vrijednosti:
38. klub = vrijednosti[1]
39. self.prikazi\_detalje\_kluba(klub)
41. self.tablica.bind("<**Double-1**>", on\_double\_click)

**Programski kodovi 5.8 i 5.9** – Funkcije prikazi\_detalje\_kluba i povezi\_dvoklik\_na\_tablicu

Funkcija *povezi\_dvoklik\_na\_tablicu* omogućuje korisniku da uz pomoć dva klika na lijevom gumbu miša na bilo koji klub unutar tablice otvara prozor s detaljima kluba. Funkcija *prikazi\_detalje\_kluba* otvara novi prozor koji sadrži naziv kluba, svu njegovu statistiku te okvir gdje se nalazi sva povijest odigranih utakmica tog kluba. Ove dvije funkcije poboljšavaju korisničko iskustvo te pomažu pri detaljnijem praćenju statistika klubova.

**Slika 5.2** – Primjer prozora sa statistikom kluba

1. def spremi\_rezultat(self):
2. try:
3. gol1 = int(self.entry\_gol1.get())
4. gol2 = int(self.entry\_gol2.get())
5. except ValueError:
6. messagebox.showerror("Greška", "Molimo unesite valjane brojeve golova.")
7. return
9. if gol1 < **0** or gol2 < 0:
10. messagebox.showerror("Greška", "Brojevi golova ne mogu biti negativni.")
11. return
13. if self.klub1 is None or self.klub2 is None:
14. messagebox.showerror("Greška", "Nema odabranih klubova za utakmicu.")
15. return
17. if (self.klub1, self.klub2) in self.odigrano\_u\_kolu:
18. messagebox.showerror("Greška", "Rezultat za ovu utakmicu je već unesen.")
19. return
21. # Ažuriraj statistiku
22. self.statistika[self.klub1]["zabijeni"] += gol1
23. self.statistika[self.klub2]["zabijeni"] += gol2
24. self.statistika[self.klub1]["primljeni"] += gol2
25. self.statistika[self.klub2]["primljeni"] += gol1
26. self.statistika[self.klub1]["gol\_razlika"] = self.statistika[self.klub1]["zabijeni"] - self.statistika[self.klub1]["primljeni"]
27. self.statistika[self.klub2]["gol\_razlika"] = self.statistika[self.klub2]["zabijeni"] - self.statistika[self.klub2]["primljeni"]
29. if gol1 > gol2:
30. self.statistika[self.klub1]["bodovi"] += 3
31. elif gol2 > gol1:
32. self.statistika[self.klub2]["bodovi"] += 3
33. else:
34. self.statistika[self.klub1]["bodovi"] += 1
35. self.statistika[self.klub2]["bodovi"] += 1
37. # Povećaj broj odigranih utakmica
38. self.statistika[self.klub1]["odigrane\_utakmice"] += 1
39. self.statistika[self.klub2]["odigrane\_utakmice"] += 1
41. # Dodaj rezultat u rezultate utakmica
42. self.rezultati\_utakmica[(self.klub1, self.klub2)] = (gol1, gol2)
44. # Pohranjuje koje utakmice su odigrane
45. self.odigrano\_u\_kolu.add((self.klub1, self.klub2))
47. # Očisti unos rezultata
48. self.entry\_gol1.delete(0, tk.END)
49. self.entry\_gol2.delete(0, tk.END)
51. # Ažuriraj gumb za utakmicu
52. if (self.klub1, self.klub2) in self.gumbovi\_utakmica:
53. self.gumbovi\_utakmica[(self.klub1, self.klub2)].config(bg="gray", state="disabled")
55. # Ažuriraj rezultate tekućeg kola
56. self.prikazi\_rezultate\_kola()
58. self.prikazi\_tablicu()
60. # Provjeri je li kolo završeno
61. if self.odigrano\_u\_kolu == set(self.kalendar[self.trenutno\_kolo - 1]):
62. self.prikazi\_progress\_bar()

**Programski kod 5.10** – Funkcija spremi\_rezultat

Jedna od najsloženijih funkcija u cijelom kodu i cijela aplikacija ovisi o ispravnosti ove funkcije. Kada korisnik unese broj golova domaćina i gosta te pritisne gumb „Spremi Rezultat“ pokreće se ova funkcija. Kada poprimi valjane brojeve golova, ažurira statistiku oba kluba, pripisuje bodove i golove, dodaje utakmicu u povijest utakmica te njen rezultat. Čisti polja za unos rezultata, ažurira gumb za utakmicu da postane neaktivan te na kraju pokreće funkcije za ažuriranje rezultata kola i za ažuriranje tablice.

## Funkcije za programsko generiranje sezone

1. def generiraj\_kalendar(self):
2. klubovi = self.klubovi.copy()
3. broj\_klubova = len(klubovi)
4. broj\_kola = (broj\_klubova - 1) \* 2
5. broj\_utakmica\_po\_kolu = broj\_klubova // 2
7. # Generiraj sve moguće parove (domaćin, gost)
8. svi\_parovi = [(domacin, gost) for domacin in klubovi for gost in klubovi if domacin != gost]
9. random.shuffle(svi\_parovi)
11. # Skup za praćenje svih odigranih utakmica
12. odigrani\_matchupi = set()
14. # Lista za spremanje rasporeda
15. kalendar = []
17. for kolo\_index in range(broj\_kola):
18. print(f"\n--- Generiranje kola {kolo\_index + 1} ---")
19. kolo = []
20. dostupni\_klubovi = set(klubovi)  # Klubovi dostupni za ovo kolo
21. brojac = 0  # Brojač za praćenje popunjenih utakmica
23. # Pokušaj generirati kolo dok ne bude ispravno popunjeno
24. while brojac < **broj\_utakmica\_po\_kolu:**
25. valjani\_par = None
26. for par in svi\_parovi:
27. domacin, gost = par
28. if (
29. domacin in dostupni\_klubovi and
30. gost in dostupni\_klubovi and
31. par not in odigrani\_matchupi
32. ):
33. valjani\_par = par
34. break
36. if valjani\_par is None:
37. break
39. # Dodaj valjani par u kolo
40. domacin, gost = valjani\_par
41. kolo.append(valjani\_par)
42. odigrani\_matchupi.add(valjani\_par)
43. dostupni\_klubovi.remove(domacin)
44. dostupni\_klubovi.remove(gost)
45. brojac += 1
47. # Ako kolo nije popunjeno, pokušaj do 5 puta popuniti preostale utakmice
48. for \_ in range(5):
49. if brojac == broj\_utakmica\_po\_kolu:
50. break  # Ako je kolo popunjeno, prekini pokušaje
52. for par in svi\_parovi:
53. domacin, gost = par
54. if (
55. domacin in dostupni\_klubovi and
56. gost in dostupni\_klubovi and
57. par not in odigrani\_matchupi
58. ):
59. kolo.append(par)
60. odigrani\_matchupi.add(par)
61. dostupni\_klubovi.remove(domacin)
62. dostupni\_klubovi.remove(gost)
63. brojac += 1
64. if brojac == broj\_utakmica\_po\_kolu:
65. break
67. # Ako kolo i dalje nije popunjeno, na silu upari preostale klubove
68. if brojac < broj\_utakmica\_po\_kolu:
69. print(f"Kolo {kolo\_index + 1} nije popunjeno nakon 5 pokušaja. Na silu uparujem preostale klubove...")
70. slobodni\_klubovi = list(dostupni\_klubovi)
71. while len(slobodni\_klubovi) > 1:
72. domacin = slobodni\_klubovi.pop(0)
73. gost = slobodni\_klubovi.pop(0)
74. kolo.append((domacin, gost))
75. odigrani\_matchupi.add((domacin, gost))
76. brojac += 1
78. # Dodaj kolo u kalendar
79. kalendar.append(kolo)
80. print(f"Kolo {kolo\_index + 1} završeno: {kolo}")
82. return kalendar

**Programski kod 5.11** – Funkcija generiraj\_kalendar

Funkcija *generiraj\_kalendar* koristi se prilikom programskog generiranja utakmica za sezonu. U funkciji se za generiranje utakmica koristi biblioteka *random* kako bi utakmice istinski bile nepredvidive. Za ispravnost parova unutar cijele sezone i unutar kola, prolazi se više provjera uvjeta, koristi se skup *dostupni\_klubovi* iz kojeg se filtriraju klubovi tokom svakog kola. Obzirom da nasumično generiranje u jednostavnim petljama ne generira željen broj parova svaki put, dodana je još jedna petlja koja se vrti pet puta za svako prazno mjesto unutar kola kako bi se osigurao točan broj utakmica u kolu. Ukoliko i dalje nisu ispunjene sve utakmice za kolo, na kraju program upari preostale klubove bez uvjetovanja jer logikom eliminacije, i ti klubovi što su preostali su kompatibilni jedno za drugo. Rezultat ove složene funkcije s rigoroznim uvjetovanjem i petljama je učestalo i precizno generiranje utakmica za svako kolo bez ikakvih logičkih grešaka u sastavljanju kalendara.

1. def generiraj\_datume(self, broj\_kola, samo\_vikendom=True):
2. datumi = []
3. danas = self.pocetni\_datum\_lige  # Početni datum lige
5. for \_ in range(broj\_kola):
6. kolo\_datumi = []
8. if samo\_vikendom:
9. # Pronađi prvi slobodan vikend za cijelo kolo
10. while True:
11. subota = danas if danas.weekday() == 5 else danas + timedelta(days=(5 - danas.weekday()))
12. nedjelja = subota + timedelta(days=1)
14. # Provjeri je li cijelo kolo unutar praznika
15. if any(pocetni <= subota <= zavrsni or pocetni <= nedjelja <= zavrsni for pocetni, zavrsni in self.praznici):
16. danas = nedjelja + timedelta(days=1)  # Pomakni na prvi ponedjeljak
17. continue
18. break
20. # Dodaj utakmice za subotu i nedjelju naizmjenično
21. utakmice\_u\_kolu = len(self.klubovi) // 2
22. for i in range(utakmice\_u\_kolu):
23. if i % 2 == 0:  # Alterniraj: parne utakmice subotom
24. trenutni\_datum = subota
25. else:  # Neparne utakmice nedjeljom
26. trenutni\_datum = nedjelja
28. kolo\_datumi.append(trenutni\_datum.strftime("%d.%m.%Y"))
30. danas = subota + timedelta(days=7)  # Pomakni na sljedeću subotu
31. else:
32. # Pronađi prvi slobodan ponedjeljak za cijelo kolo
33. while danas.weekday() != 0 or any(pocetni <= danas <= zavrsni for pocetni, zavrsni in self.praznici):
34. danas += timedelta(days=1)
36. # Dodaj utakmice za svaki dan u tjednu
37. utakmice\_u\_kolu = len(self.klubovi) // 2
38. for i in range(utakmice\_u\_kolu):
39. trenutni\_datum = danas + timedelta(days=(i % 7))  # Rotiraj dane unutar tjedna
40. kolo\_datumi.append(trenutni\_datum.strftime("%d.%m.%Y"))
42. danas += timedelta(days=7)  # Pomakni na sljedeći tjedan
44. datumi.append(kolo\_datumi)
46. return datumi

**Programski kod 5.12 –** Funkcija generiraj\_datume

Funkcija *generiraj\_datume* se također kao i funkcija *generiraj\_kalendar* koristi pri programskom generiranju utakmica. Njena uloga je da svakoj generiranoj utakmici obzirom na parametre koje je korisnik zadao i na kolo u kojem se utakmica nalazi da zada ispravan termin za utakmicu. Ovu funkcija je ostvarena uz pomoć biblioteke *datetime*. Funkcija osigurava da je datum prve utakmice u sezoni na dan ili nakon dana koji korisnik zada za početak sezone. Također osigurava da svako kolo ima svoj tjedan, tj. svoj vikend ukoliko je izbor za odigravanje vikendom odabrano. Također preskače sve datume koji spadaju unutar praznika. Ne dozvoljava da se kolo počinje na dan koji nije ponedjeljak (odnosno subota) i ne dozvoljava da utakmice prekorače u drugi tjedan.

1. def unesi\_praznike(self, parent=None):
2. try:
3. if parent is None:
4. parent = self.root
6. broj\_razdoblja = simpledialog.askinteger("Unos praznika", "Koliko razdoblja pauze želite unijeti?", parent=parent)
7. if broj\_razdoblja is None or broj\_razdoblja < **0:**
8. return
10. praznici = []
11. for i in range(broj\_razdoblja):
12. pocetni\_datum = simpledialog.askstring("Unos praznika", f"Unesite početni datum za razdoblje {i + 1} (dd.mm.yyyy):", parent=parent)
13. zavrsni\_datum = simpledialog.askstring("Unos praznika", f"Unesite završni datum za razdoblje {i + 1} (dd.mm.yyyy):", parent=parent)
15. try:
16. pocetni = datetime.strptime(pocetni\_datum, "%d.%m.%Y")
17. zavrsni = datetime.strptime(zavrsni\_datum, "%d.%m.%Y")
18. if pocetni > zavrsni:
19. messagebox.showerror("Greška", f"Početni datum za razdoblje {i + 1} ne može biti nakon završnog datuma.", parent=parent)
20. continue
21. praznici.append((pocetni, zavrsni))
22. except ValueError:
23. messagebox.showerror("Greška", "Neispravan datum ili format datuma", parent=parent)
25. # Spoji preklapajuće ili uzastopne praznike
26. praznici.sort()
27. spojeni\_praznici = []
28. for pocetni, zavrsni in praznici:
29. if not spojeni\_praznici or spojeni\_praznici[-1][1] < **pocetni** - timedelta(days=1):
30. spojeni\_praznici.append((pocetni, zavrsni))
31. else:
32. spojeni\_praznici[-1] = (spojeni\_praznici[-1][0], max(spojeni\_praznici[-1][1], zavrsni))
34. self.praznici = spojeni\_praznici
35. self.prikazi\_pauze()
36. except Exception as e:
37. messagebox.showerror("Greška", f"Došlo je do greške: {e}", parent=parent)

**Programski kod 5.13 –** Funkcija unesi\_praznike

Funkciju je također kao i prethodnu omogućila biblioteka *datetime*. Kada korisnik izabere programsko generiranje sezone, ova funkcija ispituje korisnika koliko razdoblja prazničkih želi, te za svaku koju je unio početak i kraj tog razdoblja. Ukoliko se neka razdoblja preklapaju, program ih sam povezuje u jedno veće razdoblje kako ne bi dolazilo do problema prilikom generiranja datuma.

## Funkcije za ručni unos sezone

1. def unesi\_raspored(self):
2. self.kalendar = []
3. self.datumi = []
4. odigrane\_utakmice = set()
5. if hasattr(self, "lbl\_pauze"):
6. self.lbl\_pauze.pack\_forget()
7. broj\_kola = (len(self.klubovi) - 1)\*2
9. # Kreiraj novi prozor za unos
10. prozor = tk.Toplevel(self.root)
11. prozor.title("Ručni unos rasporeda")
12. prozor.geometry("700x500")
13. prozor.transient(self.root)
14. prozor.grab\_set()
16. # Tablica za prikaz unosa
17. tablica = ttk.Treeview(prozor, columns=("Kolo", "Domaćin", "Gost", "Datum"), show="headings", height=15)
18. tablica.heading("Kolo", text="Kolo")
19. tablica.heading("Domaćin", text="Domaćin")
20. tablica.heading("Gost", text="Gost")
21. tablica.heading("Datum", text="Datum")
22. tablica.column("Kolo", width=50, anchor="center")
23. tablica.column("Domaćin", width=200, anchor="center")
24. tablica.column("Gost", width=200, anchor="center")
25. tablica.column("Datum", width=150, anchor="center")
26. tablica.pack(fill="both", expand=True, padx=10, pady=10)
28. trenutno\_kolo = 1
29. odabrani\_klubovi = set()
30. domacin = None

**Programski kod 5.14** – Funkcija unesi\_raspored

Funkcija *unesi\_raspored* postavlja zadane vrijednosti za broj kola koja su limitirana kako ne bi došlo do prekoračenja te otvara novi prozor veličine 700x500 piksela na kojem korisnik ima pregled svojih unosa. Za svaki unos je korisniku vidljivo kolo, domaćin, gost i datum te utakmice. Ova funkcija također ima četiri funkcije unutar sebe koje se opisuju u narednim stranicama dokumenta.

1. def otvori\_prozor\_domacina():
2. prozor\_domacin = tk.Toplevel(prozor)
3. prozor\_domacin.title("Odaberi domaćina")
4. prozor\_domacin.geometry("400x300")
5. prozor\_domacin.transient(prozor)
6. prozor\_domacin.grab\_set()
8. tk.Label(prozor\_domacin, text="Odaberi domaćina:", font=("Helvetica", 14, "bold")).pack(pady=10)
10. for klub in self.klubovi:
11. # Provjeri je li klub već ugostio sve ostale klubove
12. broj\_gostiju = sum(1 for kolo in self.kalendar for (domacin, gost) in kolo if domacin == klub)
13. if broj\_gostiju >= len(self.klubovi) - 1:
14. # Onemogući gumb ako je klub već bio domaćin svim ostalim klubovima
15. gumb = tk.Button(prozor\_domacin, text=f"{klub} (sve ugostio)", width=30, state="disabled", bg="gray")
16. elif trenutno\_kolo <= len(self.kalendar) and any(klub in (domacin, gost) for domacin, gost in self.kalendar[trenutno\_kolo - 1]):
17. # Onemogući gumb ako je klub već igrao u trenutnom kolu
18. gumb = tk.Button(prozor\_domacin, text=f"{klub} (već igra)", width=30, state="disabled", bg="gray")
19. else:
20. # Omogući gumb za odabir domaćina
21. gumb = tk.Button(prozor\_domacin, text=klub, width=20,
22. command=lambda k=klub: [odaberi\_domacina(k), prozor\_domacin.destroy()])
23. gumb.pack(pady=5)

**Programski kod 5.15 –** Funkcija otvori\_prozor\_domacina

Funkcija *otvori\_prozor\_domacina* spada pod funkciju *unesi\_raspored*. Kada korisnik počinje ručno unositi utakmicu, ova funkcija otvara prozor s popisom domaćina koje korisnik može odabrati. Ukoliko je klub već ugostio sve druge klubove, ili je igrao ovo kolo, gumb na kojem se nalazi naziv domaćina je onemogućen.

1. def otvori\_prozor\_gosta(domacin):
2. prozor\_gost = tk.Toplevel(prozor)
3. prozor\_gost.title("Odaberi gosta")
4. prozor\_gost.geometry("400x300")
5. prozor\_gost.transient(prozor)
6. prozor\_gost.grab\_set()
8. tk.Label(prozor\_gost, text="Odaberi gosta:", font=("Helvetica", 14, "bold")).pack(pady=10)
10. for klub in self.klubovi:
11. # Provjeri je li klub isti kao domaćin
12. if klub == domacin:
13. continue
14. # Provjeri je li klub već igrao u trenutnom kolu
15. elif trenutno\_kolo <= len(self.kalendar) and any(klub in (domacin, gost) for domacin, gost in self.kalendar[trenutno\_kolo - 1]):
16. continue
17. # Provjeri je li kombinacija domaćin-gost već unesena
18. elif any((domacin, klub) in kolo for kolo in self.kalendar):
19. continue
20. else:
21. # Omogući gumb za odabir gosta
22. gumb = tk.Button(prozor\_gost, text=klub, width=20,
23. command=lambda k=klub: [odaberi\_gosta(k), prozor\_gost.destroy()])
24. gumb.pack(pady=5)

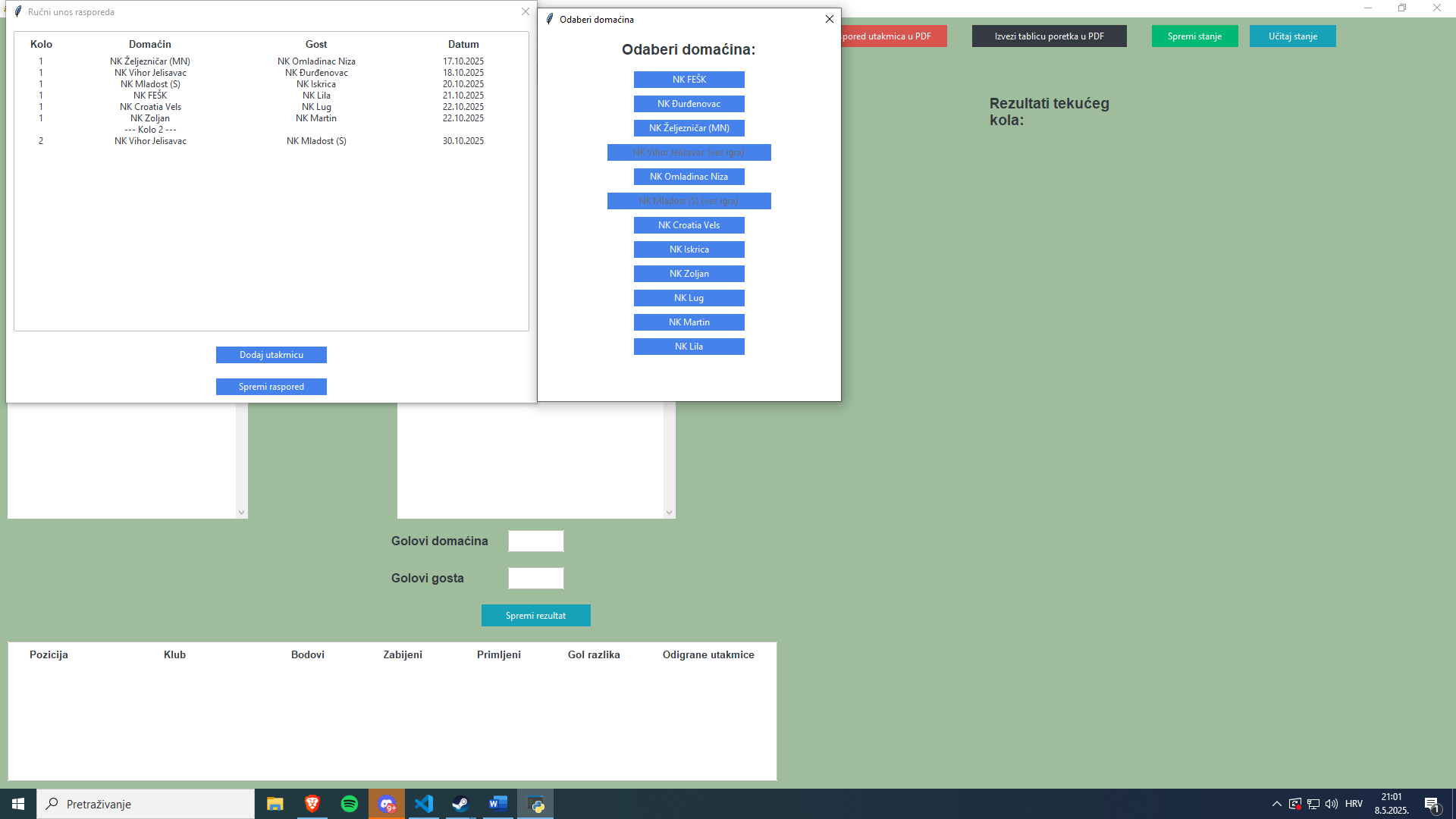
**Programski kod 5.16 –** Funkcija otvori\_prozor\_gosta

Funkcija je gotovo identična prethodnoj funkciji, samo se provjeravaju drugi uvjeti – da klub nije isti kao domaćin te da ista kombinacija domaćin – gost već nije bila unesena.

1. # Funkcija za odabir domaćina
2. def odaberi\_domacina(klub):
3. nonlocal domacin
4. domacin = klub
5. otvori\_prozor\_gosta(domacin)
6. # Funkcija za odabir gosta
7. def odaberi\_gosta(klub):
8. nonlocal trenutno\_kolo
9. nonlocal domacin
10. gost = klub
12. # Provjeri je li kombinacija domaćina i gosta već unesena
13. for kolo in self.kalendar:
14. if (domacin, gost) in kolo:
15. messagebox.showerror("Greška", f"Utakmica između {domacin} i {gost} već postoji u rasporedu.")
16. return
18. # Unos datuma
19. datum = simpledialog.askstring("Dodaj utakmicu", "Unesite datum (dd.mm.yyyy):", parent=prozor)
20. try:
21. datum\_obj = datetime.strptime(datum, "%d.%m.%Y")
22. except ValueError:
23. messagebox.showerror("Greška", "Neispravan format datuma. Koristite format dd.mm.yyyy.", parent=prozor)
24. return
26. # Dodaj utakmicu u tablicu i kalendar
27. tablica.insert("", "end", values=(trenutno\_kolo, domacin, gost, datum))
28. while len(self.kalendar) < **trenutno\_kolo:**
29. self.kalendar.append([])
30. self.datumi.append([])
31. self.kalendar[trenutno\_kolo - 1].append((domacin, gost))
32. self.datumi[trenutno\_kolo - 1].append(datum\_obj.strftime("%d.%m.%Y"))
33. odabrani\_klubovi.add(domacin)
34. odabrani\_klubovi.add(gost)
36. # Provjeri je li kolo popunjeno
37. if len(self.kalendar[trenutno\_kolo - 1]) == len(self.klubovi) // 2:
38. trenutno\_kolo += 1
39. if trenutno\_kolo > broj\_kola:
40. messagebox.showinfo("Raspored kompletiran", "Sva kola su unesena. Liga će sada započeti.")
41. self.zapocni\_ligu\_iz\_rasporeda(prozor)
42. return
43. tablica.insert("", "end", values=("", f"--- Kolo {trenutno\_kolo} ---", "", ""))  # Vizualni razdjelnik
44. odabrani\_klubovi.clear()
46. # Gumb za dodavanje utakmice
47. btn\_dodaj\_utakmicu = tk.Button(prozor, text="Dodaj utakmicu", command=otvori\_prozor\_domacina, bg="green", fg="white", width=20)
48. btn\_dodaj\_utakmicu.pack(pady=10)
50. # Gumb za spremanje rasporeda
51. btn\_spremi = tk.Button(prozor, text="Spremi raspored", command=lambda: self.zapocni\_ligu\_iz\_rasporeda(prozor), bg="gray", fg="white", width=20)
52. btn\_spremi.pack(pady=10)

**Programski kod 5.17 –** Funkcije za odabir domaćina i odabir gosta

Funkcija za odabir domaćina je puno jednostavnija od one za odabir gosta, jer funkcija *odabir\_gosta* također u sebi sadrži sve korake što dolaze nakon što korisnik jednostavno pritisne na gumb kluba kojeg želi da bude gost. Ukoliko je došlo do greške prilikom filtriranja valjanih i nevaljanih utakmica i korisnik izabere utakmicu koja se već dogodila, ova funkcija ne dozvoljava korisniku da to unese i forsira korisnika da odabere neku drugu kombinaciju. Nakon toga, korisnik unosi datum na koji će se utakmica odigrati, te se svi podaci o unesenoj utakmici memoriraju i provjerava se je li popunjeno to kolo, te se nastavlja na iduće ili se završava unos jer su sve utakmice za sezonu unesene.



**Slika 5.3 –** Primjer sučelja prilikom ručnog unosa, otvoreni su prozori „prozor\_domacin“ i „prozor“

## Funkcije s izvozom i uvozom datoteka

1. def spremi\_stanje(self, filename="liga\_stanje.pkl"):
2. try:
3. stanje = {
4. "klubovi": self.klubovi,
5. "statistika": self.statistika,
6. "kalendar": self.kalendar,
7. "datumi": self.datumi,
8. "trenutno\_kolo": self.trenutno\_kolo,
9. "odigrano\_u\_kolu": self.odigrano\_u\_kolu,
10. "praznici": self.praznici,
11. "rezultati\_utakmica": self.rezultati\_utakmica,
12. }
13. with open(filename, "wb") as f:
14. pickle.dump(stanje, f)
15. messagebox.showinfo("Spremanje", "Stanje lige je uspješno spremljeno.")
16. except Exception as e:
17. messagebox.showerror("Greška", f"Došlo je do greške prilikom spremanja: {e}")

**Programski kod 5.18 –** Funkcija spremi\_stanje

Funkcija *spremi\_stanje* se aktivira prilikom pritiska na njen pripadajući gumb na korisničkom sučelju. Ova funkcija pohranjuje sve trenutne podatke koje je korisnik unio, tj. stvorio svojim djelovanjem unutar aplikacije. Pohranu podataka omogućuje biblioteka *pickle*.

1. def ucitaj\_stanje(self, filename="liga\_stanje.pkl"):
2. try:
3. with open(filename, "rb") as f:
4. stanje = pickle.load(f)
5. self.klubovi = stanje["klubovi"]
6. self.statistika = stanje["statistika"]
7. self.kalendar = stanje["kalendar"]
8. self.datumi = stanje["datumi"]
9. self.trenutno\_kolo = stanje["trenutno\_kolo"]
10. self.odigrano\_u\_kolu = stanje["odigrano\_u\_kolu"]
11. self.praznici = stanje.get("praznici", [])  # Dodano: učitavanje prazničnih datuma
12. self.rezultati\_utakmica = stanje.get("rezultati\_utakmica", {})  # Dodano: učitavanje rezultata utakmica
14. # Ažuriraj UI nakon učitavanja
15. self.prikazi\_utakmice()
16. self.prikazi\_tablicu()
17. self.prikazi\_klubove()
18. self.prikazi\_rezultate\_kola()
19. self.prikazi\_pauze()
20. self.entry\_klub.config(state="disabled")
21. self.btn\_dodaj\_klub.config(state="disabled")
22. self.btn\_pocni\_ligu.config(state="disabled")
23. messagebox.showinfo("Učitavanje", "Stanje lige je uspješno učitano.")
24. except FileNotFoundError:
25. messagebox.showerror("Greška", "Datoteka sa stanjem lige nije pronađena.")
26. except Exception as e:
27. messagebox.showerror("Greška", f"Došlo je do greške prilikom učitavanja: {e}")

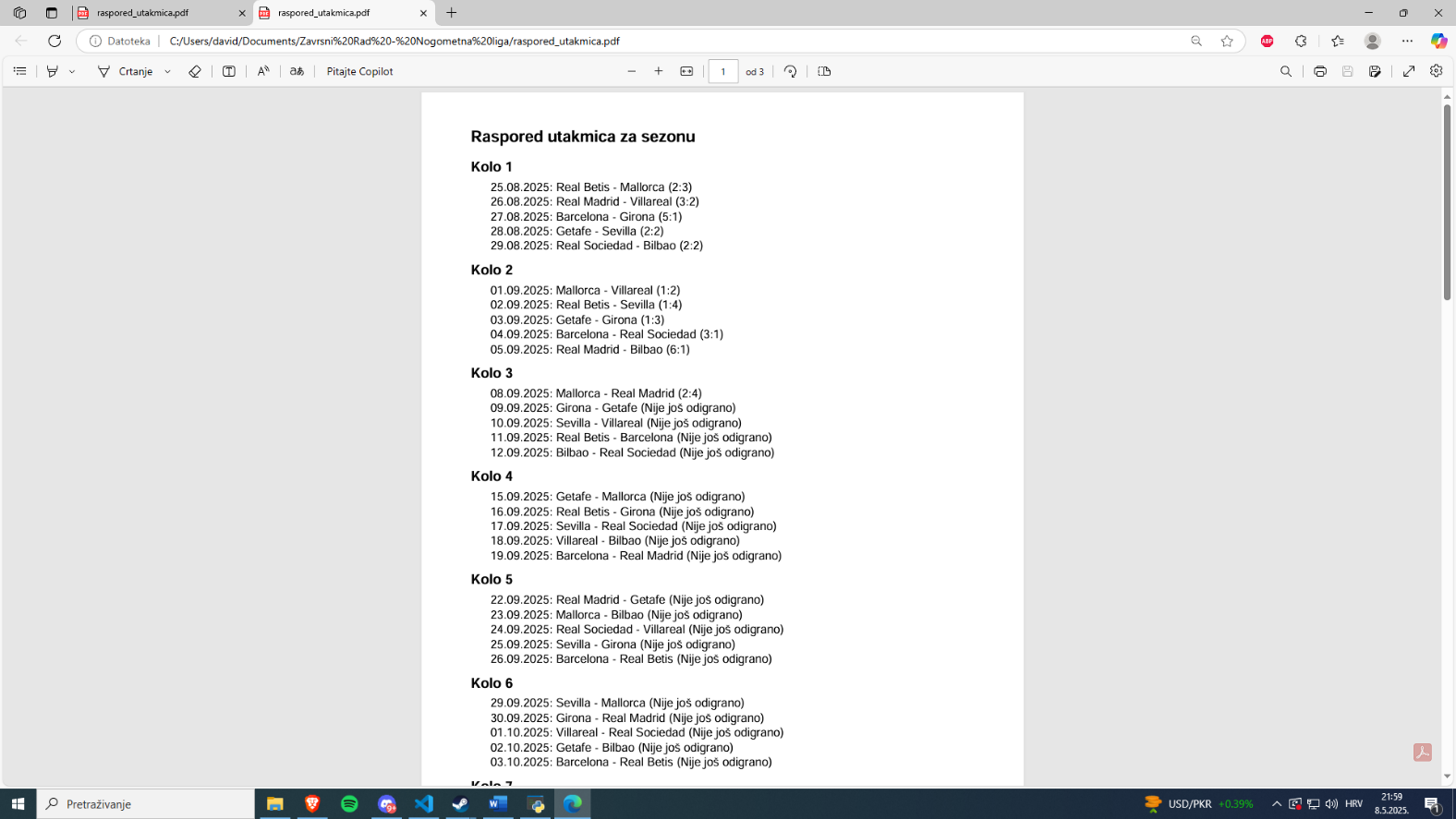
**Programski kod 5.19 –** Funkcija ucitaj\_stanje

Funkcija *ucitaj\_stanje* učitava .pkl datoteku koja je generirana pomoću funkcije *spremi\_stanje*. Učitavaju se sve pohranjene informacije, tj. klubovi, statistika klubova, sve utakmice u sezoni, datumi svake utakmice u sezoni, utakmice iz tekućeg kola i rezultate iz njega, praznička razdoblja te svi rezultati utakmica.

1. def izvezi\_raspored\_u\_pdf(self, filename="raspored\_utakmica.pdf"):
2. try:
3. c = canvas.Canvas(filename, pagesize=letter)
4. width, height = letter
5. y = height - 50  # Početna visina za ispis
7. c.setFont("Helvetica-Bold", 16)
8. c.drawString(50, y, "Raspored utakmica za sezonu")
9. y -= 30
11. for broj\_kola, (utakmice\_kola, datumi\_kola) in enumerate(zip(self.kalendar, self.datumi), start=1):
12. c.setFont("Helvetica-Bold", 14)
13. c.drawString(50, y, f"Kolo {broj\_kola}")
14. y -= 20
16. for (domacin, gost), datum in zip(utakmice\_kola, datumi\_kola):
17. rezultat = "Nije još odigrano"
18. if (domacin, gost) in self.rezultati\_utakmica:
19. gol1, gol2 = self.rezultati\_utakmica[(domacin, gost)]
20. rezultat = f"{gol1}:{gol2}"
22. c.setFont("Helvetica", 12)
23. c.drawString(70, y, f"{datum}: {domacin} - {gost} ({rezultat})")
24. y -= 15
26. #Dodaj novu stranicu
27. if y < 50:
28. c.showPage()
29. y = height - 50
31. y -= 10  #Razmak između kola
33. c.save()
34. messagebox.showinfo("PDF generiran", f"Raspored utakmica je uspješno izvezen u '{filename}'.")
35. except Exception as e:
36. messagebox.showerror("Greška", f"Došlo je do greške prilikom generiranja PDF-a: {e}")

**Programski kod 5.20 –** Funkcija izvezi\_raspored\_u\_pdf

Ova funkcija uz pomoć biblioteke *reportlab* izvozi raspored utakmica za sezonu u PDF dokument. Funkcija se automatski pokreće kada se započinje liga i generiraju sve utakmice, ali korisnik ju može pokrenuti pritiskom na gumb kada god želi. Sve utakmice kojima je unesen rezultat prije nego što je PDF dokument generiran će imati pripisan rezultat u svom redu u dokumentu.

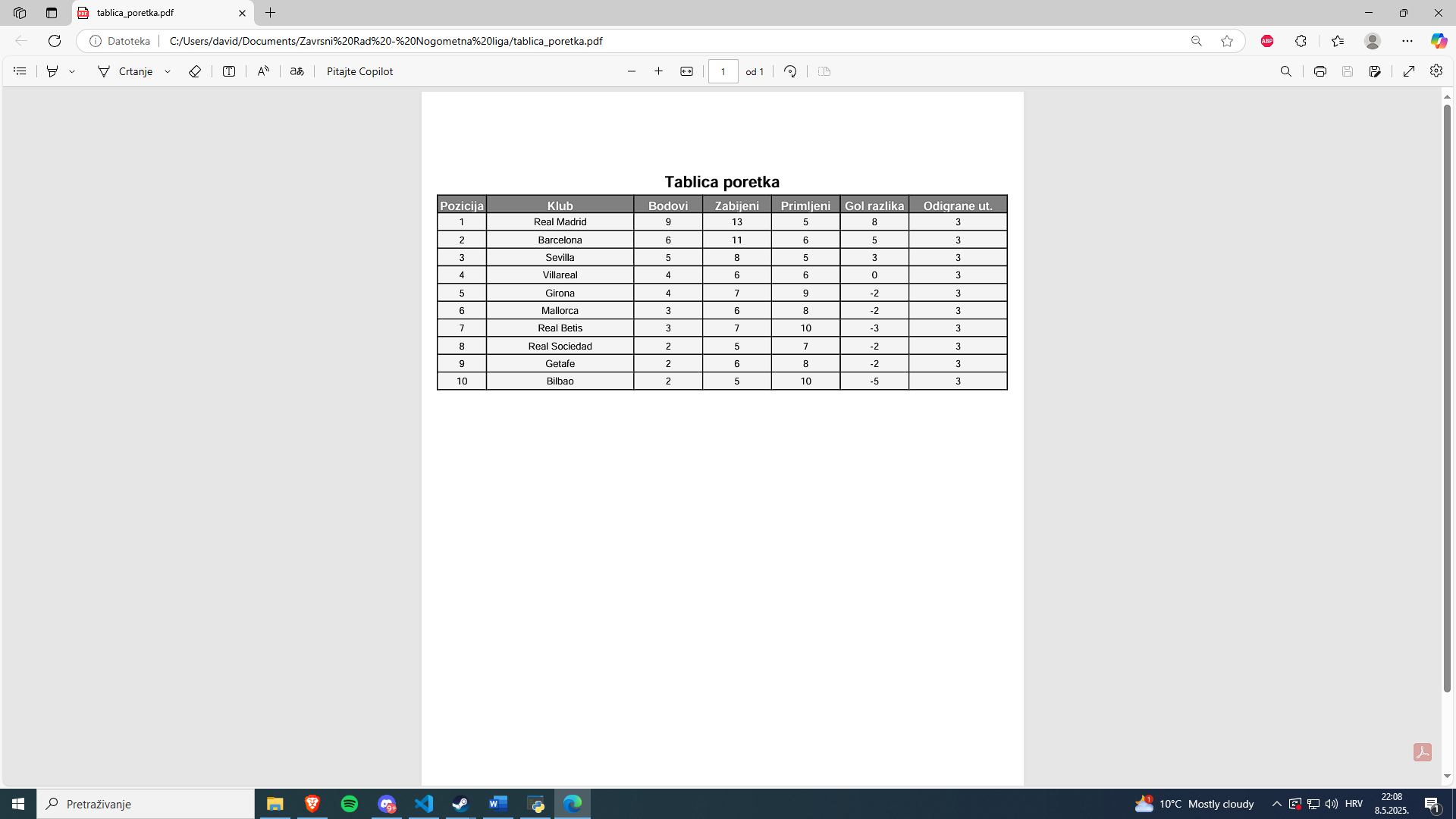


**Slika 5.4 –** Primjer PDF dokumenta rasporeda s nekoliko unešenih utakmica

1. def izvezi\_tablicu\_u\_pdf(self, filename="tablica\_poretka.pdf"):
2. try:
3. # Kreiraj PDF dokument
4. doc = SimpleDocTemplate(filename, pagesize=letter)
5. elements = []
7. # Naslov PDF-a
8. naslov = [["Tablica poretka"]]
9. naslov\_tablica = Table(naslov, colWidths=[500])
10. naslov\_tablica.setStyle(TableStyle([
11. ("ALIGN", (0, 0), (-1, -1), "CENTER"),
12. ("FONTNAME", (0, 0), (-1, -1), "Helvetica-Bold"),
13. ("FONTSIZE", (0, 0), (-1, -1), 16),
14. ("BOTTOMPADDING", (0, 0), (-1, -1), 12),
15. ]))
16. elements.append(naslov\_tablica)
18. # Zaglavlje tablice
19. data = [["Pozicija", "Klub", "Bodovi", "Zabijeni", "Primljeni", "Gol razlika", "Odigrane ut."]]
21. # Podaci iz tablice poretka
22. sortirano = sorted(self.statistika.items(), key=lambda x: (-x[1]["bodovi"], -x[1]["gol\_razlika"]))
23. for pozicija, (klub, stats) in enumerate(sortirano, start=1):
24. data.append([
25. str(pozicija),
26. klub,
27. str(stats["bodovi"]),
28. str(stats["zabijeni"]),
29. str(stats["primljeni"]),
30. str(stats["gol\_razlika"]),
31. str(stats["odigrane\_utakmice"]),
32. ])
34. # Kreiraj tablicu
35. tablica = Table(data, colWidths=[50, 150, 70, 70, 70, 70, 100])
36. tablica.setStyle(TableStyle([
37. ("GRID", (0, 0), (-1, -1), 1, colors.black),  # Vidljive linije
38. ("BACKGROUND", (0, 0), (-1, 0), colors.grey),  # Siva pozadina za zaglavlje
39. ("TEXTCOLOR", (0, 0), (-1, 0), colors.whitesmoke),  # Bijela boja teksta za zaglavlje
40. ("ALIGN", (0, 0), (-1, -1), "CENTER"),  # Poravnanje teksta
41. ("FONTNAME", (0, 0), (-1, 0), "Helvetica-Bold"),  # Bold font za zaglavlje
42. ("FONTSIZE", (0, 0), (-1, 0), 12),  # Veličina fonta za zaglavlje
43. ("FONTSIZE", (0, 1), (-1, -1), 10),  # Veličina fonta za podatke
44. ("BACKGROUND", (0, 1), (-1, -1), colors.whitesmoke),  # Bijela pozadina za podatke
45. ]))
47. elements.append(tablica)
49. # Spremi PDF
50. doc.build(elements)
51. messagebox.showinfo("PDF generiran", f"Tablica poretka je uspješno izvezena u '{filename}'.")
52. except Exception as e:
53. messagebox.showerror("Greška", f"Došlo je do greške prilikom generiranja PDF-a: {e}")

**Programski kod 5.21 –** Funkcija izvezi\_tablicu\_u\_pdf

Funkcija *izvezi\_tablicu\_u\_pdf*  opširnija je od prethodne funkcije zbog tog što je puno više grafički orijentirana. Dok su u *izvezi\_raspored\_u\_pdf* jednostavni redovi s utakmicama koji su segmentirani po kolima, ovdje se generira cijela tablica s poretkom koja je vrlo pregledna i jednostavna. U zaglavlju su svi podaci kao i u tablici koja se nalazi u aplikaciji, to su pozicija, naziv kluba, bodovi, zabijeni golovi, primljeni golovi, gol razlika te odigrane utakmice.



**Slika 4.5 –** Primjer tablice u generiranom PDF dokumentu

# Zaključak

Dokumentacija obuhvaća izradu aplikacije za praćenje lokalne nogometne lige koja omogućuje korisniku unos klubova, generiranje rasporeda (ručno/programski), unos rezultata, generiranje *„save file-a“* praćenje tablice poretka te izvoz u PDF dokument. Aplikacija je razvijena u programskom jeziku *Python*, uz korištenje *Tkinter* i *ttkbootstrap* biblioteka za grafičko sučelje, pickle modula za serijalizaciju podataka te reportlab biblioteke za generiranje PDF dokumenata.

Sustav je osmišljen tako da krajnjem korisniku pruži intuitivno, pregledno i stabilno radno okruženje. Posebna pažnja posvećena je logici rasporeda i validaciji unosa kako bi se osigurala konzistentnost lige. Modularna arhitektura i jasna podjela funkcionalnosti omogućuju jednostavno proširenje i održavanje aplikacije.

Tehnička dokumentacija prati svaki aspekt razvoja – od planiranja korisničkog sučelja, opisa ključnih funkcija, do načina izvoza i pohrane podataka. Aplikacija omogućava konzistentno i jednostavno praćenje i upravljanje lokalne nogometne lige, a njezina fleksibilnost i skalabilnost čine ju pogodnom i za buduće nadogradnje.

# Literatura

**[1] „**About“ sekcija web-stranice *Python-a,*

[**https://www.python.org/about/**](https://www.python.org/about/) **-** 9.5.2025.

**[2]** „tkinter — Python interface to Tcl/Tk“,

[**https://docs.python.org/3/library/tkinter.html**](https://docs.python.org/3/library/tkinter.html) - 9.5.2025.

**[3] „**ttkbootstrap“**,**

[**https://ttkbootstrap.readthedocs.io/en/latest/**](https://ttkbootstrap.readthedocs.io/en/latest/) **-** 9.5.2025.

**[4]** „tkinter.ttk — Tk themed widgets“,

[**https://docs.python.org/3/library/tkinter.ttk.html**](https://docs.python.org/3/library/tkinter.ttk.html) **-** 9.5.2025.

**[5]** ReportLab docs – „ReportLab PDF Toolkit“,

[**https://docs.reportlab.com/reportlab/userguide/ch1\_intro/**](https://docs.reportlab.com/reportlab/userguide/ch1_intro/) **-** 9.5.2025.

**[6]** Visual Studio Code,

[**https://code.visualstudio.com/**](https://code.visualstudio.com/) **-** 9.5.2025.