Aplikacija za sparivanje igrača na turniru

Komljenović, Davor

Undergraduate thesis / Završni rad

2019

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:617875

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-07-13**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Stručni studij

APLIKACIJA ZA SPARIVANJE IGRAČA NA TURNIRU

Završni rad

Davor Komljenović

Osijek, 2019.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac Z1S: Obrazac za imenovanje Povjerenstva za obranu završnog rada na preddiplomskom stručnom studiju

Osijek, 20.09.2019.

Odboru za završne i diplomske ispite

Imenovanje Povjerenstva za obranu završnog rada na preddiplomskom stručnom studiju

Ime i prezime studenta:	Davor Komljenović
Studij, smjer:	Preddiplomski stručni studij Elektrotehnika, smjer Informatika
Mat. br. studenta, godina upisa:	AI 4390, 22.09.2018.
OIB studenta:	20163237242
Mentor:	Doc.dr.sc. Ivan Aleksi
Sumentor:	
Sumentor iz tvrtke:	
Predsjednik Povjerenstva:	Prof.dr.sc. Željko Hocenski
Član Povjerenstva:	Doc.dr.sc. Tomislav Matić
Naslov završnog rada:	Aplikacija za sparivanje igrača na turniru
Znanstvena grana rada:	**Informacijski sustavi (zn. polje računarstvo)**
Zadatak završnog rada	Potrebno je izraditi internet aplikaciju za sparivanje igrača na turniru prema Berger i Swiss sustavu sparivanja igrača.
Prijedlog ocjene pismenog dijela ispita (završnog rada):	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 3 bod/boda Jasnoća pismenog izražavanja: 3 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene mentora:	20.09.2019.

Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija:	Potpis:
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA
I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O ORIGINALNOSTI RADA

Osijek, 02.10.2019.

Ime i prezime studenta:	Davor Komljenović
Studij:	Preddiplomski stručni studij Elektrotehnika, smjer Informatika
Mat. br. studenta, godina upisa:	AI 4390, 22.09.2018.
Ephorus podudaranje [%]:	11

Ovom izjavom izjavljujem da je rad pod nazivom**: Aplikacija za sparivanje igrača na turniru**

izrađen pod vodstvom mentora Doc.dr.sc. Ivan Aleksi

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

Sadržaj:

1. UVOD

Tema ovoga završnog rada je sparivanje igrača na turniru. U radu su implementriana dva najčešća načina sparivanja na turnirima „*round robin*“ i „*Swiss system*“. Implementacija navedenih algoritama izvršena je putem web aplikacije kako bi se omogućilo jednostavno kreiranje šahovskih turnira. Aplikacija omogućuje registraciju, kreiranje novih igrača, odabir vrste algoritma za sparivanje igrača i kreiranje turnira. U poglavalju o tehnologijama obrađene su korištene tehnologije za izradu aplikacije, a u poglavlju o algoritmima je pojašnjen način rada algoritama te njihovi nedostatci. U posljednjem poglavalju pojašnjen je način izrade web aplikacije.

1.1. Zadatak završnog rada

Potrebno je izraditi internet aplikaciju za sparivanje igrača na turniru prema Berger i Swiss sustavu sparivanja igrača.

2. TEHNOLOGIJE

Sve tehnologije korištene u izradi aplikacije su besplatne. Iste se koriste u razvoju Web aplikacija te obuhvaćaju područja front-end i back-end razvoja.

Korištene tehnologije su:

- XAMPP s inačicom PHP-a 7.3.0 i SQL-a 5.0.12-dev
- Laravel 5.8
- HTML 5
- CSS
- Bootstrap 4.3.1
- Javascript, jQuery 3.4.1
- AJAX

2.1. XAMPP

XAMPP je besplatna i multiplatforma razvijena od strane Apache Friends. XAMPP je slobodna programska podrška. XAMPP se sastoji od Apache HTTP servera, MySQL-a i interpretera za skripte napisane u PHP-u i Perl-u te je dostupan na Windowsu, Linuxu i MAC OS-u. Omogućuje jednostavno stvaranje servera za razvoj na lokalnoj mreži u svrhe testiranja i implementacije [1].

2.1.1 PHP

PHP (PHP:Hypertext Preprocessor) je skriptni jezik i prevoditelj namijenjen prvenstveno programiranju dinamičnih web stranica. PHP je slobodna programska podrška i prvenstveno se koristiti na Linux serverima. Osnovna značajka PHP je izvođenje programa na serverskoj strani (eng. Back End) [2].

PHP se može koristiti na tri osnovna načina:

- Skriptiranje sa strane poslužitelja
- Naredbeno skriptiranje
- GUI aplikacije sa strane klijenta

PHP radi na svim većim operacijskim sustavima kao što su Windows, Mac OS X i mnoge inačice Unixa uključujući Linux, FreeBSD, Debian, Ubuntu i Solaris.

Jedna od PHP-ovih najznačajnijih osobina je njegova široka podrška za baze podataka. PHP podržava sve veće poznate baze podataka (uključujući MySQL, PostgreSQL, Oracle, Sybase,

MS_SQL, DB2 i baze podataka kompatibilne s ODBC-om) te čak i mnoge malo poznate baze podataka. Podržane su čak i novije NoSQL baze podataka kao što su SQLite i MongoDB [3].

2.1.2. MySQL

Baza podataka je organizirani skup podataka pohranjen na sustavan način u vanjskoj memoriji računala. Pohranjeni podaci su istovremeno dostupni različitim korisnicima i aplikacijskim programima.

Prilikom izrade ove WEB aplikacije korišten je MySQL, a bazom podataka upravljano je pomoću phpMyAdmina i Laravel Tinker okvira. Primjer MySQL naredbi i njihov opis vidljiv je u tablici 2.1.

Tablica 2.1. – *MySQL naredbe*

Naredba	Opis
USE *[ime baze]*;	Koristi bazu podataka *[ime baze]*
DROP TABLE IF EXISTS *[table name];*	Obriši cijelu tablicu
CREATE TABLE *[ime tablice] (* *[ime polja] tip modifikatori* *);*	Stvori tablicu pod nazivom [ime tablice] s definiranim poljima i njihovim modifikatorima Modifiktori: PRIMARY KEY, AUTO_INCREMENT
INSERT INTO *[ime tablice]* **VALUES** *(...)*	Umetni novi zapis u tablicu *[ime tablice]*; Vrijednosti (eng. Values) moraju biti točnog tipa u točnom redosljedu.
SELECT * FROM *[ime tablice]* **WHERE** *[uvjet]* **ORDER BY** *[polje]*	Odaberi sva polja iz *[ime tablice] koja zadovljavaju [uvjet]*; sortiraj po zadanom polju
UPDATE *[ime tablice]* **SET** *[polje] = [vrijednost]* **WHERE** *[uvjet];*	Promjeni postojeći zapis. Koristi zadani uvjet kako bi odredio koje je zapise potrebno promjeniti i postavi vrijednosti specifiranih polja.
DELETE FROM *[ime tablice]* **WHERE** [uvjet]	Obriši sve zapise iz tablice *[ime tablice]* koji zadovoljavaju *[uvjet].*
CREATE VIEW *[ime pogleda]* **AS** [upit]	Kreiraj virtualnu tablicu koja sadržava rezultate kompleksnog upita kao što je spajanje više tablica.

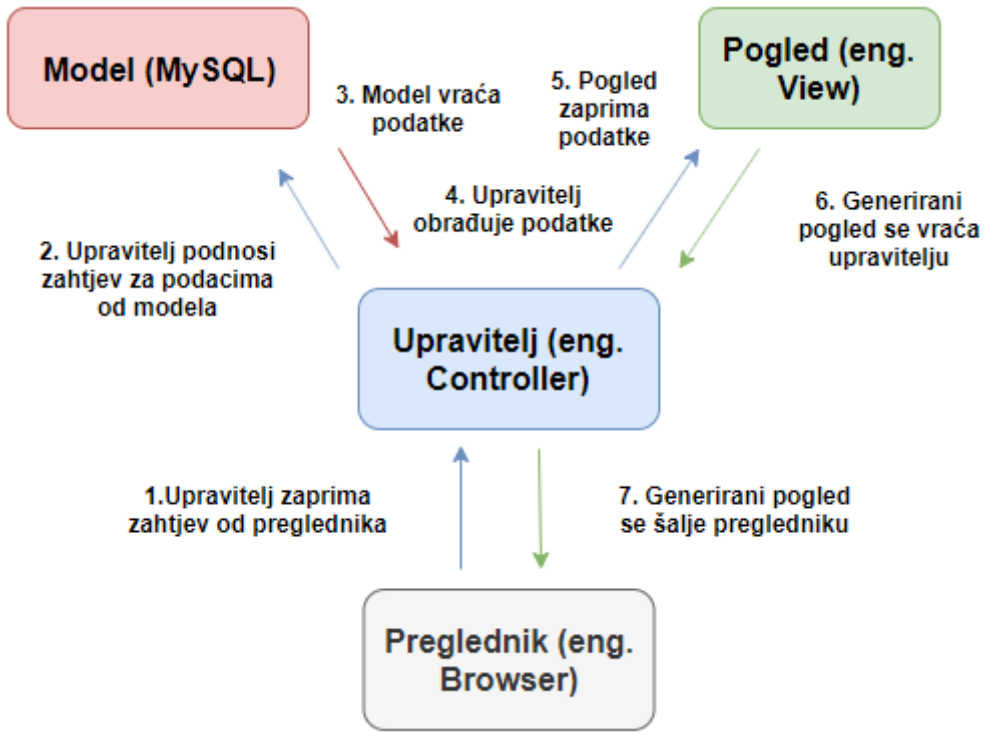
MySQL (eng. My Structured Query Language) sustav je za upravljanje bazom podataka. MySQL baze su relacijskog tipa, koje su pokazale kao jedan od najboljih načina pohranjivanja i pretraživanja velikih količina podataka i obično predstavljaju osnovu svakog informacijskog sustava, tj. temelj poslovnih subjekta koji svoje poslovanje baziraju na pohrani i dostupnosti kvalitetnih i brzih informacija [4]. MySQL serveri su predviđeni za važne, jako opterećene produkcijske sustave, ali su pogodni i za korištenje u manjim projektima.

2.2. Laravel

Laravel je programski okvir temeljen na objektno orijentiranom PHP-u te koristi MVC (eng. Model-View-Controller) arhitekturu. Razvoj Laravel okruženja započeo je Taylor Otwell 2011.godine [5]. Ova aplikacija izrađena je u Laravel inačici 5.8 , a trenutno je najavljena nova inačica 6.0.

Programski okviri kao Laravel, Symfony, Silex i Lumen olakšavaju razvoj aplikacija i pružaju standardizirane smjernice za razvoj aplikacija. Programski okviri u sebi sadržavaju funkcionalnosti kao što je registracija i prijava korisnika, rukovanje datotekama, rukovanje putanjama (eng. Routes) i mnoge druge. Tako se pojednostavljuje proces izrade aplikacije i povećava sigurnost. Naime, budući da razvojni tim Laravela zaprima primjedbe svih korisnika programskog okvira vrlo brzo je moguće prepoznati nedostatke i ranjivosti te iste otkloniti.

MVC (engl. Model-View-Controller) arhitektura je način dizajna koji se često koristi prilikom razvoja korisničkih sučelja. MVC arhitektura dijeli programsku logiku na tri glavna dijela modele, poglede i upravitelje (Sl 2.1.). Ovakva arhitektura postala je iznimno popularna za dizajniranje WEB aplikacija. Popularni programski jezici kao što su JavaScript, Python, Ruby, PHP i #C imaju MVC programske okvire namijenjene za razvoj WEB aplikacija.



Slika 2.1. – *MVC arhitektura u Laravelu*

Eloquent je ORM (eng. Object-relational mapping) alat. Eloquent pruža jedno sučelje za rad s više različitih vrsta baza podataka [5]. ORM je programska tehnika za pretvaranje podataka između nekompatibilnih sustava koristeći objektno orijentirani programski jezik.

2.3. HTML

HTML (engl. HyperText Markup Language) je prezentacijski jezik za izradu web stranica. Hipertekst dokument stvara se pomoću HTML jezika. HTML jezikom oblikuje se sadržaj i stvaraju se hiperveze hipertext dokumenata. Prikaz teksta omogućuje web preglednik. Temeljna zadaća HTML jezika je uputiti web preglednik kako prikazati hipertekst [6]. Primjer HTML koda i oznaka vidljiv je na slici 2.2.

HTML	Izlaz
`<h1> Heading 1 tekst </h1>`	Heading 1 tekst
`<h2> Heading 2 tekst </h2>`	Heading 2 tekst
`<h3> Heading 3 tekst </h3>`	Heading 3 tekst

Slika 2.2. – *Primjer HTML koda i rezultata*

2.4. CSS

CSS je akronim za Cascading Style Sheet. CSS je jezik koji služi za definiranje stila web stranice. CSS se najčešće koristi zajedno s HTML jezikom pomoću kojeg se definiraju struktura i sadržaj web-stranica. CSS je jedna od temeljnih tehnologija na kojim se temelji suvremeni web. Riječ cascading označava kaskadnu primjenu CSS-pravila. CSS je ključan za stvaranje vizualno zanimljivih i privlačnih stranica. Iznimno važna mogućnost CSS-a je promjena svih elemenata jednim pravilom, ali i mogućnost primjene pravila na samo jedan dio koda [7].

2.5. Bootstrap

Bootstrap je front end programsko okruženje za izradu dizajna web stranica i web aplikacija. Bootstrap je CSS programsko okruženje namijenjeno za stvaranje responzivnih stranica s fokusom na razvoj stranica koje su prilagođene prikazu na mobilnim uređajima. Naime, Bootstrap sadrži predefinirane klase za pojedine elemente na web stranici ili aplikaciji. Navedene klase prilagođene su za razvoj responzivnih stranica i imaju moderan dizajn. Koristeći bootstrap moguće je vrlo brzo razviti modernu stranicu pogodnu za korištenje na više uređaja.

2.6. Javascript

JavaScript je skriptni programski jezik te spada u front end tehnologije budući da se izvršava u web pregledniku na strani korisnika. Koristi se kako bi učinio web stranice i aplikacije interaktivnim. HTML, CSS i JavaScript čine temeljne tehnologije za izradu front enda web stranica i aplikacija.

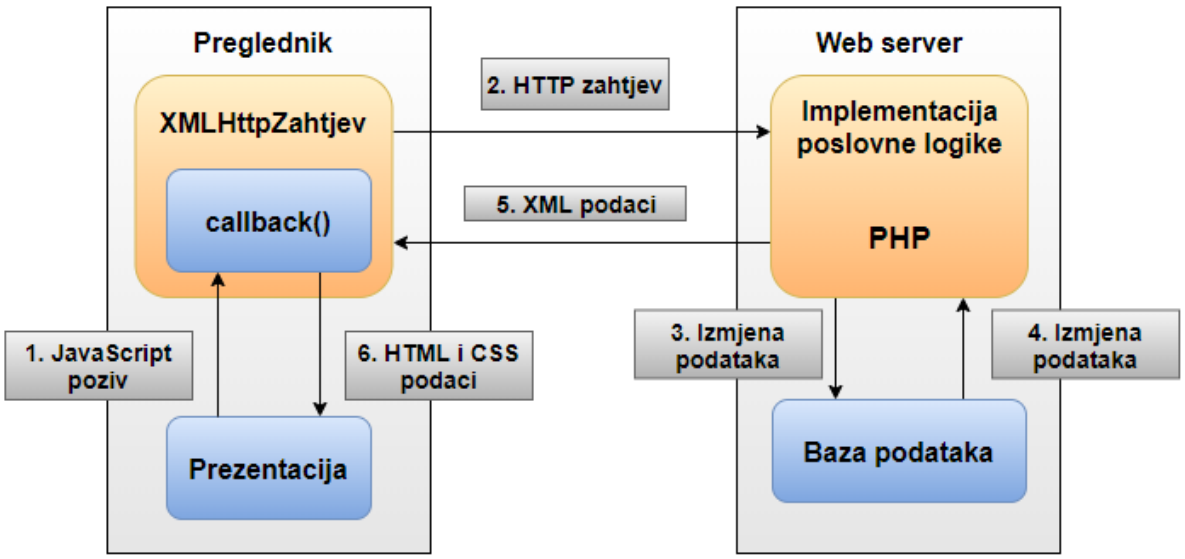
JavaScript omogućuje DOM manipulaciju. DOM manipulacija omogućuje kreiranje novih elemenata, mijenjanje postojećih i izvršavanje promjena na temelju određenih događaja i slično. Također, JavaScript omogućuje izvršavanje programske logike na strani klijenta i validaciju podataka kojima nije potrebno opterećivati server.

2.6.1. jQuery

jQuery je JavaScript biblioteka dizajnirana kako bi pojednostavilo manipuliranje DOM-om, ali i kontrolu događaja (engl. events), CSS animacija i Ajax-a. U kolovozu 2019.godine čak 74.2% Web stranica koristi jQuery [8].

2.7. AJAX

AJAX (engl. Asynchronous JavaScript and XML) je set tehnika za razvoj weba koje koriste mnoge web tehnologije na strani klijenta kako bi kreirali asinkronu web aplikaciju. Web aplikacije koje koriste AJAX imaju mogućnost slanja i primanja podataka sa servera asinkrono (u pozadini) bez osvježavanja trenutne stranice (Sl. 2.3.). Danas se najčešće koristi JSON format umjesto XML formata.



Slika 2.3. – *Primjer Ajax zahtjeva (eng. request)*

3. ALGORITMI ZA SPARIVANJE IGRAČA NA TURNIRIMA

3.1. Berger ili „*round robin*“ sustav

„*Round robin*“ turnir je natjecanje u kojem svaki natjecatelj igra protiv svakog drugog natjecatelja. Izraz „*round robin“* dolazi do francuskog izraza „*ruban“* koji u prijevodu znači vrpca. Pobjednik je natjecatelj s najviše pobjeda.

3.1.1 Prednosti „*round robin*“ sustava

U teoriji je *„round robin“* najpošteniji način određivanja ukupnog pobjednika turnira budući da svaki natjecatelj igra sa svakim protivnikom. Prilikom stvaranja rasporeda mečeva ne postoje nositelji na temelju prijašnjih rezultata te se tako osiguravaju isti uvjeti za sve natjecatelje. Ovaj sustav omogućava igračima da u slučaju lošijeg nastupa u jednom od mečeva imaju mogućnost nastavka natjecanja te da i dalje imaju dobre mogućnosti za pobjedu na cjelokupnom turniru. U ovakvom sustavu je element sreće reduciran na minimum budući da ne postoji mogućnost teškog ždrijeba i ispadanja zbog istoga u ranijoj fazi turnira.

Velika prednost „*round robin“* turnira je mogućnost kvalitetnog određivanja ukupnog pobjednika, ali i poretka svih ostalih sudionika turnira. Što je osobito važno za turnire koji nagrađuju više sudionike bodovima za sljedeća natjecanja ili nagradama.

Najpoznatiji primjeri „*round robin*“ turnira u današnjem vremenu su engleski Premiership, španjolska La Liga, talijanska Seria A i njemačka Bundesliga. U navedenim ligama osamnaest do dvadeset nogometnih klubova igra dvokružni „*round robin“* turnir kako bi se omogućilo da svaka ekipa odigra dvije utakmice protiv svakog protivnika. Jednu na domaćem terenu, a drugu na gostujućem terenu. Postotak pobjeda, neriješenih ishoda i poraza domaćih ekipa prema podacima za međunarodni nogomet tijekom promatranog perioda od ukupno 6730 utakmica iznosio je 50.3% (3385) pobjeda domaćih ekipa, 24.6% (1656) neriješenih ishoda i 25.1% (1689) poraza [9].

Prema navedenim podacima jasno je vidljivo važnost osiguravanja jednakih uvjeta za sve sudionike natjecanja. Prilikom kreiranja *„round robin“* turnira za natjecanja u šahu iznimno je važno omogućiti svim sudionicima jednak broj mečeva odigranih s bijelim šahovskim figurama.

William F. Streeter analizirao je rezultate 5598 šahovikih mečeva odigranih na internacionalnim šahovskim turnirima između 1851. i 1932.godine. Streeter je saznao da u

38.12% slučajeva pobjeđuje igrač s bijelim figurama, 31.31% igrač s crnim figurama, a izjednačenim ishodom završava 31.76% mečeva [10].

Nadalje, „*round robin"* turnir se koristi na natjecanjima kao što je Svjetsko Prvenstvo u nogometu, Olimpijske igre u košarci i mnogim drugim natjecanjima sličnog formata. Naime, navedena natjecanja započinju grupnom fazom u kojoj svi sudionici grupe igraju jednostruki „*round robin*" kako bi se odredile ekipe koje napreduju u sljedeću fazu natjecanja te kako bi se odredili protivnici u eliminacijskom dijelu turnira.

3.2.2. Nedostatci „*round robin"* sustava

Glavni i osnovni nedostatak „*round robin"* turnira je trajanje u odnosu na druge vrste turnira. Također veliki broj kasnijih mečeva često nema rezultatski značaj te isti gube interes promatrača. Nadalje, potrebno je odrediti načine rješavanja situacija u kojem dva ili više natjecatelja ima jednak broj bodova nakon završetka turnira. Neki od načina za određivanje pobjednika u slučaju jednakog broja bodova su međusobni susreti, trajanje susreta, gol razlika, a u krajnjim slučajevima i bacanje novčića ili neki drugi način odabira pobjednika na sreću.

Ako se „*round robin"* turnir koristi za određivanje natjecatelja koji prolaze u sljedeću fazu natjecanja postoji mogućnost da se jedan ili više natjecatelja matematički osigura prolazak u sljedeću fazu natjecanja te zbog toga svoje preostale mečeve odigraju s maksimalnim zalaganjem. Naime, u navedenom slučaju postoji mogućnost da ekipe koje će se sigurno plasirati u sljedeći krug pokušaju namjerno izgubiti kako bi izbjegle težeg protivnika. Primjer navedene situacije su Olimpijske igre 2012. godine u Londonu gdje su četiri ženska par u najtecanju u badmintonu diskvalificirana zbog pokušaja namjernog gubljenja mečava kako bi izbjegli bolje rangirane protivnike u eliminacijskom dijelu turnira [11].

3.2. „*Swiss*" sustav

„*Swiss"* sustav sparivanja je neeliminacijski način sparivanja igrača s predodređenim brojem kola. U odnosu na „*round robin*" sustav potrebno je odigrati osjetno manji broj kola. Naime, u „*Swiss"* sustav nije nužno da svaki igrač igra protiv svakog protivnika. Za prvo kolo protivnici se određuju slučajnim odabirom ili na temelju ratinga. Nakon odigranog prvog kruga natjecatelji se susreću s drugim natjecateljima koji imaju sličan trenutni rezultat. Uparivanje se izvodi prema setu postavljenih pravila. Dva igrača se ne mogu susresti dva puta u istom turniru.

„*Swiss"* sustav se najčešće koristi na turnirima u kojim sudjeluje previše igrača da bi se mogao koristiti „*round robin*", a eliminacija igrača prije kraja turnira nije poželjna. Naime, kod

eliminacijskog turnira postoji mogućnost pobjede natjecatelja koji trenutno nije najbolji igrač, ali zbog „lošeg dana“ najboljeg igrača može pobijediti na turniru. Korištenjem „*Swiss*“ sustava natjecateljima se daje mogućnost da unatoč „lošem danu“ zadrže mogućnost osvajanja turnira ako dobro odigraju preostale mečeve na turniru. „*Swiss*“ sustav bi trebao rezultirati konačnim pobjednikom iako svaki igrač nije igrao sa svakim sudionikom turnira.

„Swiss“ sustav je prvi puta korišten na turniru u šahu koji je održan 1895.godine u Zurich-u. Naziv „*Swiss system*“ proizlazi iz navedenog događaja [12].

3.2.1 Proces sparivanja

Turnir u šahu koji koristi „Swiss“ sustav sparivanja protivnika započinje sparivanjem protivnika za prvo kolo na temelju ratinga ili nasumično. Nakon odigranog prvog kola svi sudionici igraju sljedeće kolo u kojem pobjednici igraju protiv pobjednika, a gubitnici protiv gubitnika. U sljedećim kolima protivnici se nastavljaju sparivati na temelju njihovih kumulativnih rezultata i tako svi igrači igraju s protivnicima koji imaju isti ili približno isti rezultat. Dva igrača ne smiju igrati dva puta međusobno tijekom turnira te se sparivanjem pokušava osigurati da svi igrači imaju jednak broj mečeva odigranih s bijelim i crnim figurama. Osobita važnost se posvećuje da se osigura da niti jedan igrač ne igra tri ili više puta zaredom s istom bojom.

Detaljna pravila sparivanja ovise o korištenoj varijaciji „Swiss“ sustava. Određene varijacije mogu biti iznimno komplicirane te se upravo u takvim slučajevima koriste računalni programi za sparivanje protivnika.

U šahu se najčešće koristi specifični set pravila za sparivanje koji se naziva *„Dutch“* sustav. Kada se koristi izraz „*Swiss*“ najčešće se podrazumijeva da se radi o *„Dutch“* sustavu sparivanja.

„*Dutch*“ način sparivanja protivnika započinje kreiranjem liste igrača na temelju njihovih prijašnjih rezultata, rating ili nasumičnog odabira (Sl. 3.1.).

#	Ime	Prezime	Rating
1	Magnus	Carlsen	2882
2	Fabiano	Caruana	2818
3	Liren	Ding	2805
4	Anish	Giri	2779
5	Maxime	Vachier-Lagrave	2778
6	Wesley	So	2776
7	Ian	Nepomniachtchi	2774
8	Levon	Aronian	2765

Slika 3.1. – *Igrači grupirani po ratingu*

Na temelju kreirane liste uparuje se gornja polovica igrača s donjom polovicom igrača tako da prvi igrač igra protiv petog igrača, drugo protiv šestog (Sl. 3.2.)... Postoji mogućnost modifikacije kako bi se izbjegli susreti istih protivnika ili kako bi se izbjegla mogućnost da igrač igra tri puta zaredom s istom bojom.

1. kolo						
1	Carlsen, Magnus	2882	vs.	5	Vachier-Lagrave, Maxime	2778
2	Caruana, Fabiano	2818	vs.	6	So, Wesley	2776
3	Ding, Liren	2805	vs.	7	Nepomniachtchi, Ian	2774
4	Giri, Anish	2779	vs.	8	Aronian, Levon	2765

Slika 3.2. – *Prvo kolo napravljeno korištenjem „Dutch“ sustava na temelju slike 3.1*

Nakon odigranog prvog kola igrači se ponovno grupiraju , ali po trenutnom rezultatu. Na primjer ako nije bilo neodlučenih rezultata četiri igrača će imati jedan bod, a četiri igrača nula bodova. Igrači s jednim bodom bit će grupirani u jednu grupu te poredani na isti način kao prilikom rangiranja za prvo kolo (Sl. 3.3.).

Grupa 1				Bodovi
1	Magnus	Carlsen	2882	1
2	Fabiano	Caruana	2818	1
3	Liren	Ding	2805	1
4	Anish	Giri	2779	1

Grupa 2				Bodovi
5	Maxime	Vachier-Lagrave	2778	0
6	Wesley	So	2776	0
7	Ian	Nepomniachtchi	2774	0
8	Levon	Aronian	2765	0

Slika 3.3. – *Igrači gruprani na temlju bodova*

Igrač na poziciji jedan igrat će protiv najbolje rangiranog igrača u donjoj polovici grupe točnije trećeg igrača (Sl. 3.4.). Isti proces bit će ponovljen i za grupu igrača s nula bodova. Navedeni proces se ponavlja dok se ne odigra predodređeni broj kola.

2. kolo						
1	Carlsen, Magnus	2882	vs.	3	Ding, Liren	2805
2	Caruana, Fabiano	2818	vs.	4	Giri, Anish	2779
5	Vachier-Lagrave, Maxime	2778	vs.	7	Nepomniachtchi, Ian	2774
6	So, Wesley	2776	vs.	8	Aronian, Levon	2765

Slika 3.4. – *Drugo kolo napravljeno korištenjem „Dutch“ sustava na temelju rezultata na slici 3.3.*

Druga poznata varijacija je Monradov sustav. Igrači se prvo rangiraju po njihovom rezultatu pa po njihovoj početnoj poziciji koja može biti određena po ratingu ili nasumično. Na taj način rangirani igrači igraju prvi s drugim, treći s četvrtim... Određene modifikacije su moguće kako bi se osiguralo da su sva uvjeti zadovoljeni.

U svim varijacijama igra se zadani broj kola koji se određuje na temelju broja sudionika. Nakon zadnje runde igrači su rangirani po njihovom rezultatu te se u slučaju jednakih rezultata koristi Buchholzov šahovski rating (suma svih rezultata protivnika) ili jedan od drugih načina za odlučivanje pobjednika.

3.2.2. Prednosti „*Swiss*“ sustava

Ako na turniru u kojem je korišten *„Swiss“* sustav nema mečeva koji su završili s neriješenim ishodom za utvrđivanje pobjednika turnira dovoljan je jednak broj kola kao kod eliminacijskog turnira te se broj kola računa putem formule:

$$\frac{\log(x)}{\log(2)} = broj\ kola \tag{3-1}$$

gdje je:

- x – broj sudionika turnira

Sukladno tome turnir s tri kola može igrati osam igrača, turnir s četiri kola može igrati šesnaest igrača... Ako je odigran manji broj kola postoji mogućnost nekoliko igrača završi turnir sa savršenim rezultatom bez da odigraju meč međusobno.

Iako je putem *„Swiss“* sustava moguće saznati pobjednika u istom broju kola *„Swiss“* sustav ima jednu značajnu razliku. Igrači nakon izgubljenog meča ostaju u turniru i imaju mogućnost poboljšati svoj konačni plasman, a u nekim slučajevima i pobijediti na turniru.

Nadalje, jedna od velikih prednosti je da *„Swiss“* turnir daje dobre rezultate za određivanje svih mjesta u poretku, a ne samo prvog mjesta u poretku kao što je to slučaj u eliminacijskom turniru. Naime, u eliminacijskom sustavu postoji mogućnost da finalist nije drugi najbolji igrač ili tim već da to može biti bilo koji od natjecatelja koje je porazio pobjednik turnira.

3.2.3 Nedostatci „Swiss“ sustava

Kod turnira koji koriste *„Swiss“* sustav postoji mogućnost da natjecatelj matematički osigura prvo mjesto prije završetka turnira. Ovakav ishoda ima određene nedostatke. Turnir ne završava očekivanim dvobojem za prvo mjesto na turniru te samim time gubi se interes promatrača. Nadalje, postoji mogućnost da budući da je pobjednik osigurao prvo mjesto ne odigra zadnji meč s punim zalaganjem te tim utječe na poredak ostalih natjecatelja. Neke varijacije *„Swiss“* sustava iz ovoga razloga koriste Gibsonovo pravilo po kojem matematički pobjednik igra s protivnikom koji nema šansu doći do jednog od mjesta koji osiguravaju nagradu, a natjecatelji koji i dalje imaju šansu za osvajanje jedne od nagrada u idealnom slučaju igraju međusobno kako bi odlučili o preostalim mjestima za koje se dodjeljuje nagrada [13]. Također u nekim natjecanjima Gibsonovo pravilo koristi se i za natjecatelje koji su osigurali napredak u sljedeću rundu natjecanja.

U turnirima koji koriste *„Swiss“* sustav završne runde imaju puno veći utjecaj na konačni rezultat. Naime, lošiji ulazak u turnir sa „*Swiss*“ sustavom može biti prednost jer će igrač u tome slučaju imati veću šansu biti uparen s lošijim protivnicima u završnim rundama. Igrači šaha to obično nazivaju „*Swiss Gambit*“ [14].

3.2.4 Varijacije *„Swiss“* sustava

Ubrzani parovi ili ubrzani *„Swiss“* sustav je varijacija koja se koristiti na većim turnirima s većim brojem igrača od optimalnog broja igrača za predviđeni broj rundi. Ova metoda ranije uparuje najbolje igrače s drugim najboljim igračima i tako smanjuje broj igrača sa savršenim rezultatima u početnim rundama. Na ovaj način se može postići duplo manji broj savršenih rezultata od uobičajenog broja savršenih rezultata nakon dvije runde [15].

Kod ovakvog načina uparivanja prva polovica igrača automatski dobiva jedan bod samo za potrebe sparivanja igrača. Nakon toga se prve dvije runde uparuju na standardan način uzimajući u obzir dodani bod. Naime, svi igrači su na početku poredani po ratingu te prva četvrtina igrača igra protiv druge četvrtine igrača. Odnosno prva polovica najbolje rangiranih igrača igra međusobno. Ako u ovom slučaju u većini slučajeva pobjede bolje rangirani igrači u sljedećem kolu će druga četvrtina igrati protiv treće četvrtine igrača, a prva četvrtina igrača će igrati međusobno te će tako broj očekivanih savršenih rezultata nakon dva kola biti smanjen s 25% na 12.5%.

Ubrzani *„Swiss“* sustav ne garantira da će se broj savršenih rezultata smanjiti. Naime, u slučaju pobjeda slabije rangiranih igrača postoji mogućnost jednakog broja savršenih rezultata kao i u standardnom sustavu.

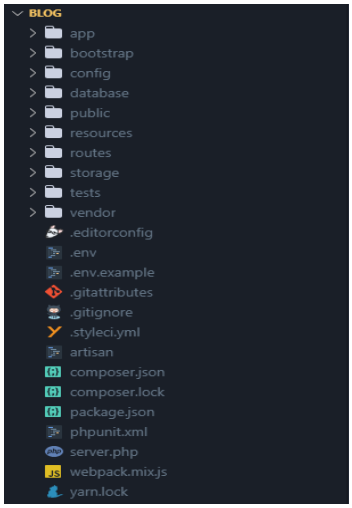
Danski sustav funkcionira na jednak način kao i Monradov sustav, ali ne postoji pravilo koje zabranjuje ponovni susret istih protivnika.

„Swiss“ sustav se najčešće koristi prilikom organizacije natjecanja u badmintonu, bridge-u, šahu, curlingu, go-u i mnogim kartaškim igrama. Također *„Swiss“* sustav koristi se za organizaciju mnogih natjecanja u elektroničkim sportovima kao što su Counter Strike: Global Offensive, DOTA2, Overwatch i Hearthstone.

4. REALIZACIJA APLIKACIJE ZA SPARIVANJE IGRAČA NA TURNIRU

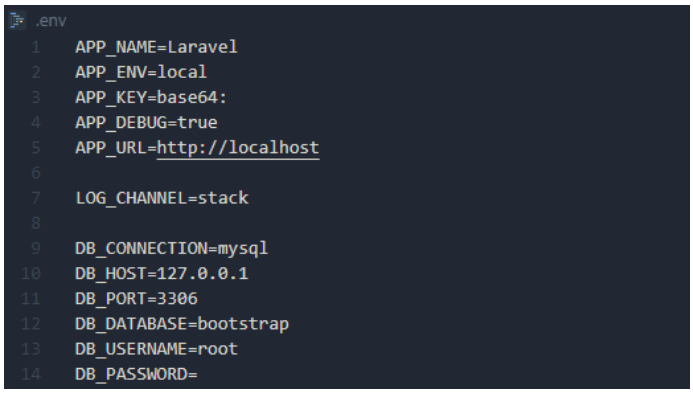
4.1. Podešavanje radne okoline

Podešavanje radne okoline započinje instalacijom XAMPP-a koji sadržava MySQL bazu podataka i PHP. Kako bismo mogli instalirati i koristiti Laravel programski okvir potrebno je instalirati Composer. Instalacija Laravela započinje naredbom „*composer global require laravel/installer*“. Kad je Composer instalira Laravel novi projekt započinjemo naredbom „laravel new [ime aplikacije]“ [16].



Slika 4.1. – *Početna struktura Laravel datoteka*

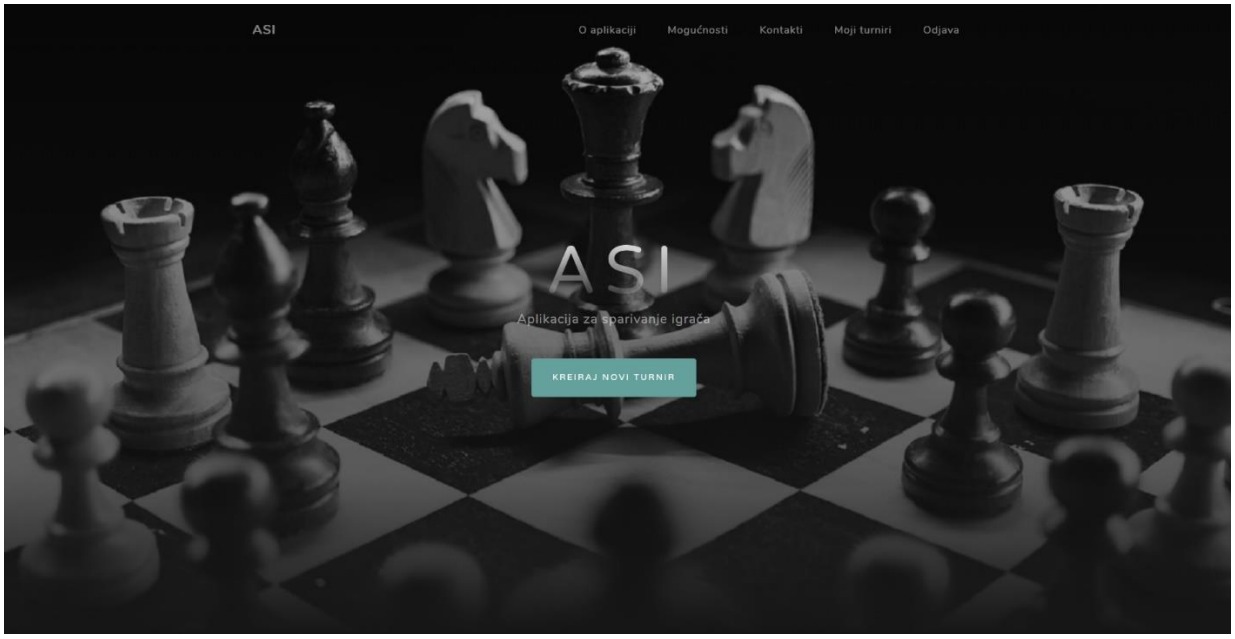
Nakon što je projekt kreiran imati će strukturu datoteka kao na slici 4.1. Sljedeći korak je podešavanje okoline u datoteci .env. Naime, u datoteci .env moguće je promijeniti ime aplikacije, ključ za enkripciju, dodati postavke za pristupanje bazi podataka i mnoge druge postavke ovisno korištenim paketima (Sl. 4.2.).



Slika 4.2. – *Osnovne postavke u .env datoteci*

4.2. Dizajn stranice

Izrada aplikacije za sparivanje igrača na turniru započela je izradom početne stranice u kojoj korištena kombinacija HTML-a, CSS-a, JavaScripta i Bootstrapa kako bi se kreirao moderan dizajn. Na početnoj stranici objašnjene su osnovne mogućnosti web aplikacije, a putem navigacijske trake moguće je pristupiti formi za prijavu ili registraciju korisnika (Sl. 4.3.).



Slika 4.3. – *Početna stranica aplikacije*

4.3. Baza podataka

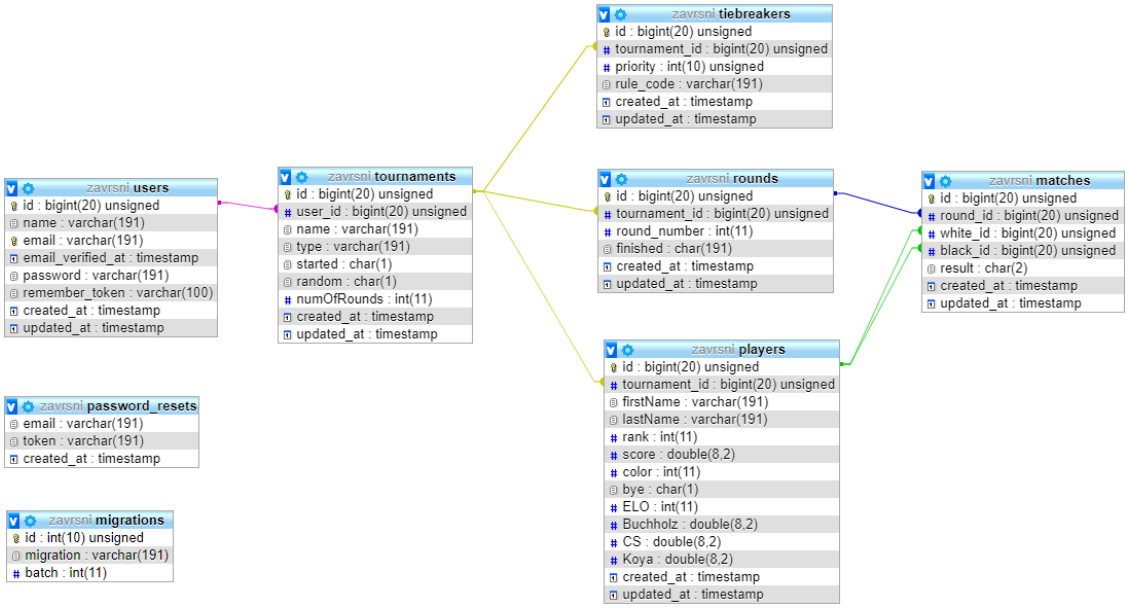
Kako bi se korisnik mogao prijaviti potrebno je izraditi modele i migracije za bazu podataka. Laravel koristi vlastiti „*schema builder*“ koji olakšava kreiranje tablica u bazi podataka (Sl. 4.4.).

```
public function up()
{
    Schema::create('tournaments', function (Blueprint $table) {
        $table->bigIncrements('id');
        $table->bigInteger('user_id')->unsigned();
        $table->string('name');
        $table->string('type');
        $table->char('started', 1)->default('F');
        $table->char('random', 1)->default('F');
        $table->integer('numOfRounds')->nullable();
        $table->timestamps();

        $table->foreign('user_id')->references('id')->on('users')->onDelete('cascade');
    });
}
```

Slika 4.4. – *Primjer migracije u Laravelu napravljene pomoću „schema buildera“*

Baza podataka za Aplikaciju za sparivanje igrača na turniru sastoji se od šest osnovnih tablica mečevi (eng.matches), igrači (eng. players), kola (eng. rounds), turniri (eng. tournaments),

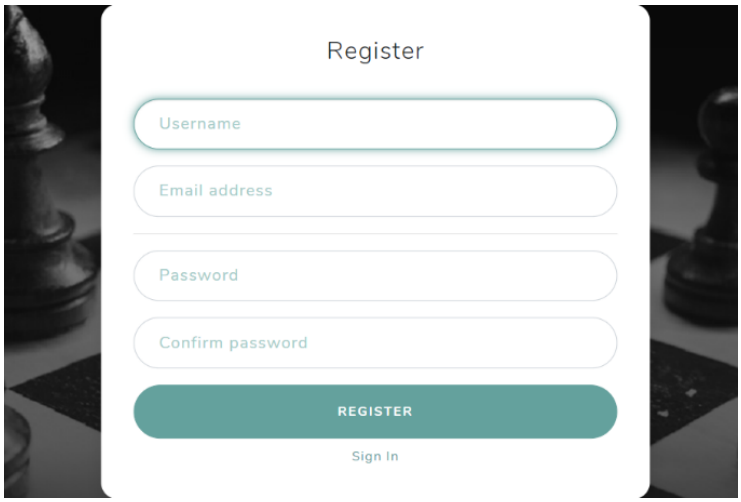
korisnici (eng. users), tiebreakers i tablica koje kreira Laravel automatski, a koriste se resetiranje zaporke i praćenje migracija (Sl. 4.5.).



Slika 4.5. – *ER dijagram baze podataka*

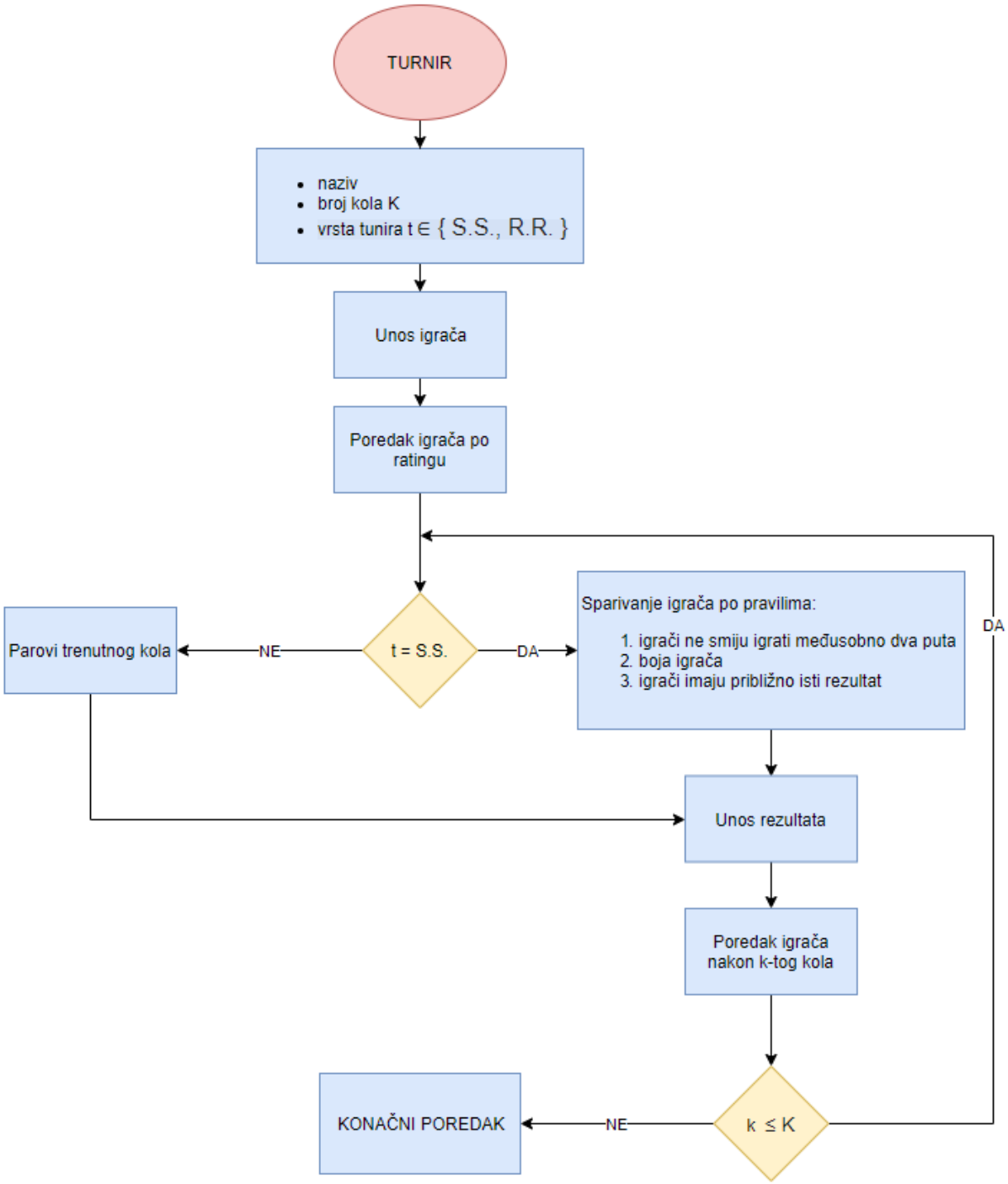
4.4. Registracija i prijava korisnika

Laravel pruža vlastito rješenje za autorizaciju korisnika. Kako bi se kreirali sve potrebne datoteke za Laravel autorizaciju potrebno je u konzolu upisati sljedeću naredbu „*php artisan make:auth*“ te nakon toga naredbu „*php artisan migrate*“ kako bi se potrebne tablice migrirale u bazu podataka [16]. Izgleda pogleda za prijavu i registraciju prilagođen je kako bi odgovarao potrebama aplikacije za sparivanje igrača (Sl 4.6.).



Slika 4.6. – *Izgled forme za registraciju*

4.5. Tok programa



Slika 4.7. – *Tok programa*

Na slici 4.7. vidljiv je tok programa na temelju kojeg će biti pojašnjen rad aplikacije od trenutka kreiranja turnira do završetka istoga te formiranja konačnog poretka.

4.6. Kreiranje novog turnira

Kreiranje novog tunira započinje popunjavanjem forme s podacima o turniru kao što su ime turnira, vrsta turnira, pravila za određivanje pobjednika u slučaju istoga broja bodova i broj kola ako je tip turnira „*Swiss*“ (Sl. 4.8.).

Slika 4.8. – *Form za kreiranje novog turnira*

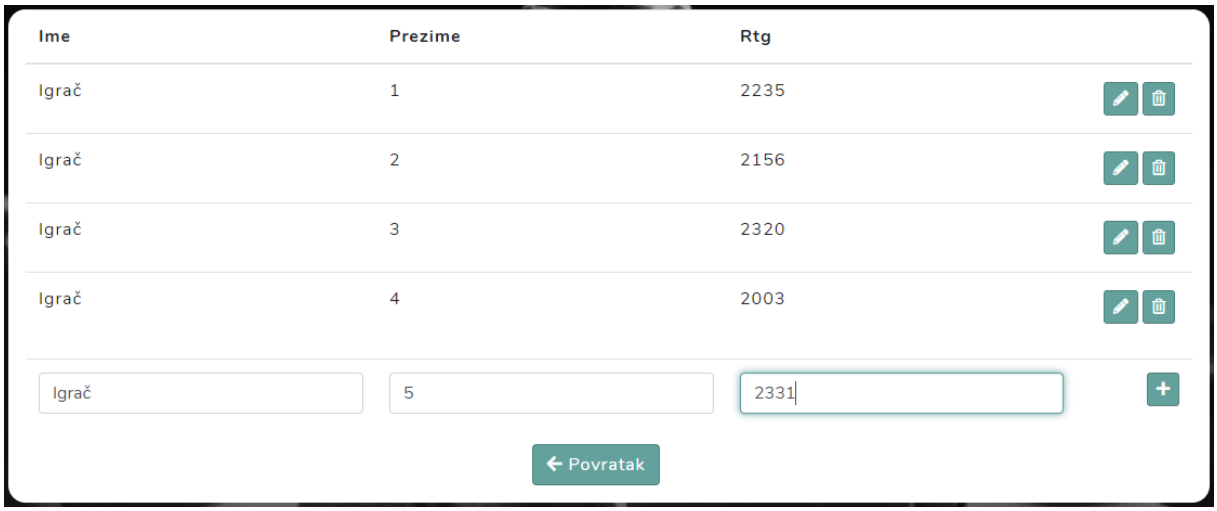
Podaci uneseni u formu šalju se na „controller“ koji izvršava validaciju unesenih podataka kako bi se osigurao unos svih potrebnih podataka za kreiranje novog turnira (Sl. 4.9.). Nakon validacije turnir se unosi u bazu podataka, a korisnika se preusmjerava na početnu stranicu kreiranog turnira.

```
$request->validate([
    'name'=>'required',
    'tournament_type'=> 'required|in:Round robin,Swiss',
    'rounds' => 'nullable|integer',
    'random' => 'required|in:T,F',
    'tiebreaker.*' => 'nullable|in:Buchholz,CS,Koya,ELO',
]);
```

Slika 4.9. – *Validacijska pravila za unos novog turnira*

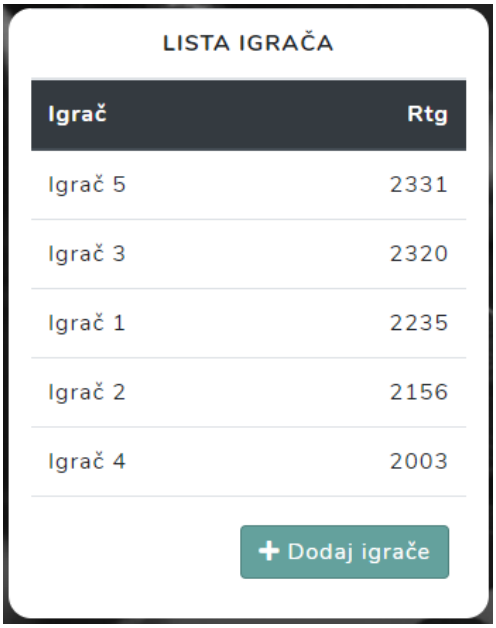
4.7. Unos igrača i poredak igrača po ratingu

Nakon što je turnir kreiran korisnik mora unijeti igrače i njihov rating (Sl. 4.10.). Za unos igrača korišten je Ajax-a kako bi se omogućilo brzo i jednostavno unošenje, promjena i brisanje igrača bez nepotrebnog osvježavanja stranice.



Slika 4.10. – *Forma za unos igrača*

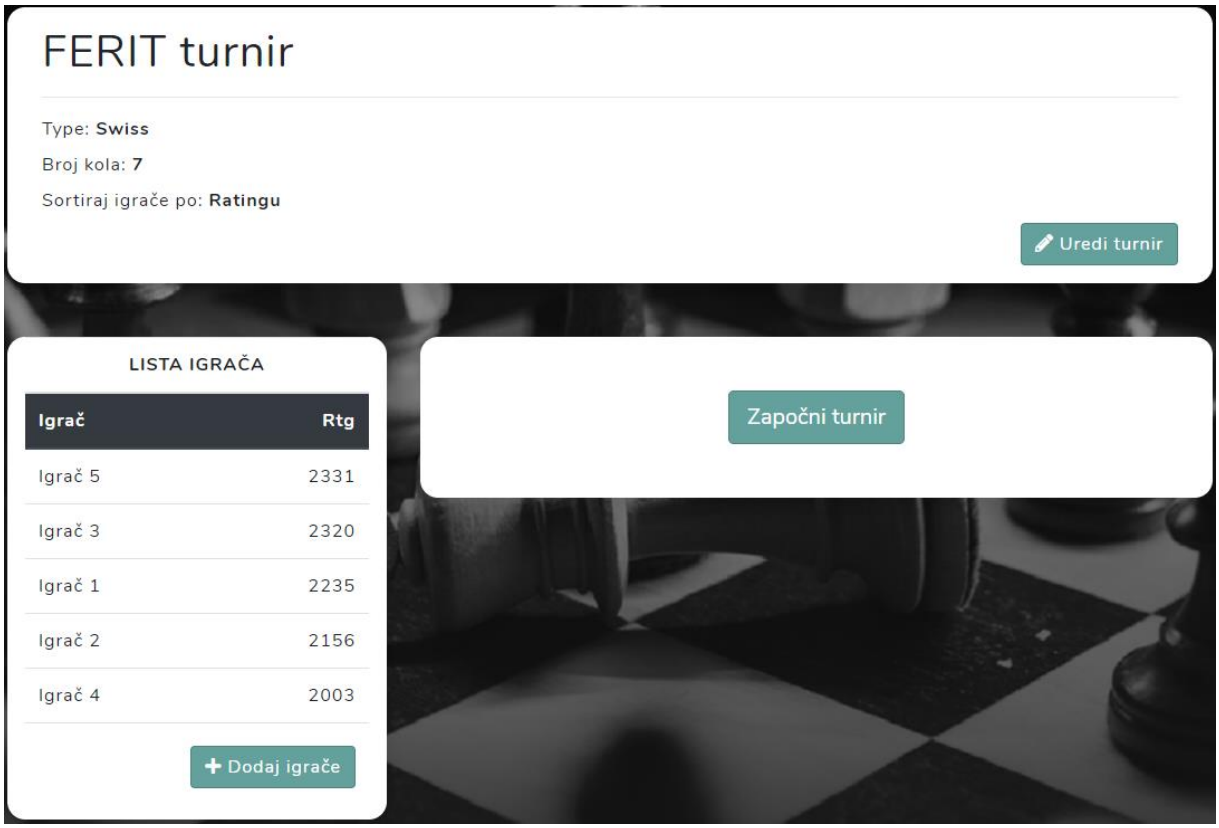
Nakon što su igrači uneseni bazu podataka formira se poredak na temelju ratinga (Sl. 4.11.). Također, korisnik može odabrati nasumičan poredak igrača prilikom kreiranja parova turnira.



Slika 4.11. – *Poredak igrača po ratingu*

4.8. Provjera vrste turnira i sparivanje igrača

Nakon što su uneseni svi sudionici turnira korisnik treba pritisnuti gumb započni turnir (Sl. 4.12.). Pritiskom navedenog gumba zaključavaju se sve postavke turnira te se provjerava vrsta turnira i kreiraju se parovi sukladno odabranoj vrsti turnira.



Slika 4.12. – *Početna stranica turnira i gumb „Započni turnir“*

4.8.1. „*Round robin*“

„*Round robin*“ turnir započinje automatskim kreiranjem svih kola budući da je konačan broj kola predodređen, a svaki igrač mora igrati protiv svakog igrača.

```php
// Generate the pairings for each round.
for ($i = 1; $i < $num_players; $i++) {

    $round = new Round;
    $round->tournament_id = $id;
    $round->round_number = $i;
    $round->save();

    $players_done = array();

    // Pair each player except the last.
    for ($player = 1; $player < $num_players; $player++) {
        if (!in_array($player, $players_done)) {

            // Select opponent.
            $opponent = $i - $player;
            $opponent += ($opponent < 0) ? $num_players : 1;

            // Ensure opponent is not the current player.
            if ($opponent != $player) {

                $match = new Match;
                $match->round_id = $round->id;

                // Choose colours.
                if (($player + $opponent) % 2 == 0 xor $player < $opponent) {
                    // Player plays white.
                    if($player != $bye){
                        $match->white_id = $players[$player-1]['id'];
                    }
                    if($opponent != $bye){
                        $match->black_id = $players[$opponent-1]['id'];
                    }
                } else {
                    // Player plays black.
                    if($opponent != $bye){
                        $match->white_id = $players[$opponent-1]['id'];
                    }
                    if($player != $bye){
                        $match->black_id = $players[$player-1]['id'];
                    }
                }

                $match->save();
```

```php
                $match = new Match;
                $match->round_id = $round->id;

                // Pair the last player.
                if ($i % 2 == 0) {
                    $opponent = ($i + $num_players) / 2;
                    // Last player plays white.
                    if($num_players != $bye){
                        $match->white_id = $players[$num_players-1]['id'];
                    }
                    if($opponent != $bye){
                        $match->black_id = $players[$opponent-1]['id'];
                    }
                } else {
                    $opponent = ($i + 1) / 2;
                    // Last player plays black.
                    if($opponent != $bye){
                        $match->white_id = $players[$opponent-1]['id'];
                    }
                    if($num_players != $bye){
                        $match->black_id = $players[$num_players-1]['id'];
                    }
                }

                $match->save();

                DB::commit();
            }

        return redirect()->back();
    }
```

Slika 4.13. – Kod za sparivanje igrača pomoću „round robin“ algoritma

Kod započinje „*for*“ petljom koja kreira nova kola (eng. rounds) i pokreće ugniježđenu „for“ petlju koja kreira parove na temelju proslijeđenog niza (eng. array) igrača (Sl. 4.13.).

4.8.2. „*Swiss*“ sustav

U ovoj aplikaciji izrađen je „Swiss“ sustav koji slijedi tri osnovna pravila:

1. Igrači ne smiju igrati međusobno dva puta tokom turnira
2. Broj mečeva s bijelim i crnim figurama mora biti približno jednak
3. Spareni igrači moraju imati približno jednak rezultat

```
for($iColor = 1; $iColor <= $maxColor; $iColor++) {

    for($iScore = 0.5; $iScore <= $maxScore; $iScore = $iScore + 0.5){

        for($iMix = 0; $iMix < floor((count($whites) + count($blacks)) / 2); $iMix++){

            $itemW = $whites[$iMix];
            $itemB = $blacks[$iMix];
            unset($whites[$iMix]);
            unset($blacks[$iMix]);
            $whites[$iMix] = $itemB;
            $blacks[$iMix] = $itemW;

            for($iPushW = 0; $iPushW < count($whites); $iPushW++){

                for($iW = 0; $iW <= $iPushW; $iW++){
                    $itemW = $whites[$iW];
                    unset($whites[$iW]);
                    array_push($whites, $itemW);
                }

                $pushedWhites = array();

                $pushedWhites = $whites;

                for($iPushB = 0; $iPushB < count($blacks); $iPushB++){

                    for($iB = 0; $iB <= $iPushB; $iB++){
                        $itemB = $blacks[$iB];
                        unset($blacks[$iB]);
                        array_push($blacks, $itemB);
                    }
```

Slika 4.14. – *Ugniježđene „for“ petlje za sparivanje igrača pomoću „Swiss“ sustava*

Svakom igraču se prati broj mečeva s bijelim i crnim figurama. Prva i druga „for“ petlja na slici 4.14. osiguravaju da se upare protivnici s približno jednakim rezultatom i odgovarajućom bojom figura. Pravilo o boji može biti prekršeno u zadnjem kolu kako bi osigurali da se u slučaju istoga rezultata mogu susresti dva najbolja natjecatelja iako pravilo o broju mečeva s istom bojom to ne dopušta. Prije sparivanja igrači su podijeljeni u dvije grupe te preostale petlje ugniježđene na slici 4.14. mijenjaju pozicije igrača dok ne pronađu kombinaciju koja odgovara zadanim pravilima. Budući da aplikacija prolazi kroz sve moguće kombinacije tražeći kombinaciju parova koja će zadovoljiti zadana pravila postoji mogućnost da kreirani parovi neće uvijek biti isti kao u sustavima koji koriste „Dutch“ sustav napravljen prema smjernicama FIDE-a.

4.9. Dodatno bodovanje

Dodatno bodovanje korisiti se za odlučivanje poretka igrača u slučaju istoga broja bodova. Jedan turnir može koristiti više načina dodatnog bodovnja. Način odlučivanja dodatnog bodovanja se odlučuje prije početka turnira. U aplikacije je moguće odabrati tri vrste dodatnog bodovanja:

- Buchholz sistem
- Sonnenborn-Berger rezultat
- Koya rezultat

4.9.1. Buchholz sistem

Buchholzov sistem je kumulativ bodova svih protivnika s kojima je igrač igrao tijekom turnira.

Ime	Prezime	Rating	Buchholz	CS	Koya	Bodovi
Igrač	4	1882	8.5	6.75	7.5	**3.5**
Igrač	1	1952	8	6.25	7.5	**3.5**
Igrač	5	1793	9.5	2.5	7	**2**
Igrač	2	2156	6.5	1.5	3.5	**2**
Igrač	8	2331	6.5	2	3.5	**2**
Igrač	3	2103	6.5	2.25	6	**1.5**
Igrač	6	2003	9.5	2	9.5	**1**
Igrač	7	2162	9	0.75	7.5	**0.5**

Slika 4.15. – *Konačni poredak turnira u kojem je pobjednik odlučen pomoću Buchholz sistema*

Buchholz se računa pomoću formule:

$$Buchholz = \sum a \tag{4-1}$$

gdje je:

$\sum a$ - zbroj bodova svih protivnika s kojima je igrač igrao na turniru

Na slici 4.15. igrač 4 je igrao protiv igrača 2, 1, 6 i 5. Buchholz za igrača 4 iznosi:

$$Buchholz = 2 + 3.5 + 1 + 2 = 8.5 \tag{4-2}$$

Budući da je igrač 1 imao Buchholz rezultat od 8 u konačnom poretku na prvom mjestu je igrač 4.

4.9.2. Sonnenborn-Berger rezultat

Sonnenborn-Berger rezultat (CS) je kumulativ bodova svih protivnika koje je igrač pobijedio i polovice bodova protivnika s kojima je igrač odigrao neodlučeno.

Ime	Prezime	Rating	Buchholz	CS	Koya	Bodovi
Igrač	3	2033	13.5	10.25	10.5	**4**
Igrač	2	1923	11	8.25	6.5	**4**
Igrač	7	1648	10	6	6.5	**4**
Igrač	8	2111	14.5	4.5	12	**2.5**
Igrač	1	2142	11.5	2	8	**2**
Igrač	5	1798	13.5	4	10.5	**1.5**
Igrač	4	2312	12.5	1.5	8	**1**
Igrač	6	1982	13.5	1	10.5	**1**

Slika 4.16. - *Konačni poredak turnira u kojem je pobjednik odlučen pomoću Sonnenborn-Berger rezultata*

Na slici 4.16. konačni pobjednik turnira odlučen je pomoću Sonnenborn-Berger rezultata koji se računa putem formule:

$$CS = \sum a + \frac{\sum b}{2} \tag{4-3}$$

gdje je:

$\sum a$ – zbroj bodova protivnika koje igrač pobjedio

$\sum b$ – zbroj bodova protivnika s kojima je igrač remizirao

Igrač 3 pobjedio je igrače 7, 1 i 4, a neodlučeno je igrao s igračima 2 i 8. Zbog toga je njegov Sonnenborn-Berger rezultat iznosio:

$$CS = 4 + 2 + 1 + \frac{4+2.5}{2} = 10.25 \tag{4-4}$$

Iako su prva tri igrača imala isti broj bodova igrač 1 zauzeo je prvo mjesto zbog najvišeg Sonnenborn-Berger rezultata. Igrač 2 je završio turnir s Sonnenborn-Berger rezultatom 8.25 te je zbog toga zauzeo drugo mjesto, a igrač 7 je zauzeo treće mjesto zbog Sonnenborn-Berger rezultata.

4.9.3. Koya rezultat

Koya rezultat je kumulativ bodova svih protivnika s kojima je igrač igrao, a koji su ostvarili 50% ili više mogućih bodova na turniru.

Ime	Prezime	Rating	Buchholz	CS	Koya	Bodovi
Igrač	8	1982	9.5	6.5	8	**3**
Igrač	5	2019	8	5	6	**3**
Igrač	6	1982	9	6	6	**3**
Igrač	7	2203	6.5	2.5	3	**2**
Igrač	2	1933	9	2.25	6	**1.5**
Igrač	3	2312	7	1.25	5	**1.5**
Igrač	4	2136	7	1.5	5	**1.5**
Igrač	1	1832	8	1	5	**0.5**

Slika 4.17. - *Konačni poredak turnira u kojem je pobjednik odlučen pomoću Koya rezultata*

Koya rezultat računa se formulom:

$$Koya = \sum a \tag{4-5}$$

gdje je:

$\sum a$ – zbroj bodova svih protivnika koji su ostvarili 50% ili više mogućih bodova

Na slici 4.17. odluka o končanom pobjedniku donesena je na temelju Koya rezultata. Igrač 8 je igrao s igračima 7, 6, 5 i 4, ali su samo igrači 7, 6 i 5 imali zbroj bodova veći ili jednak 50% mogućih bodova. Koya rezultat za igrača 8 izračunat je na sljedeći način:

$$Koya = 2 + 3 + 3 = 8 \tag{4-3}$$

Budući da su igrači 5 i 6 imali Koya rezultat 6 konačni pobjednik turnira je igrač 8.

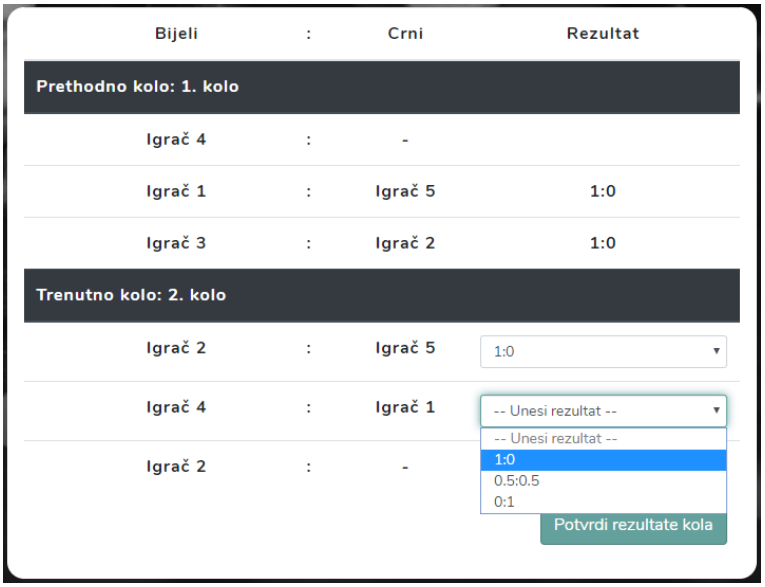
4.10. Unos rezultata i konačni poredak

Nakon svakog kola potrebno je unijeti rezultat. Unos rezultata izvršava u obliku 1:0, 0.5:0.5 i 0:1. Na temelju unesenih rezultata automatski se kreira trenutni poredak uzimajući u obzir odabrana „*tie breaker*“ pravila. Za određivanje konačnog poretka korištena je *„usort“* funkcija (Sl 4.18.). U aplikaciji je omoguće odabrati između Buchholz sustava, Sonneborn-Berger rezultata, Koya rezultata i ratinga te bilo koju kombinaciju navedenih pravila za odlučivanje pobjednika u slučaju istoga rezultata.

```php
// sort players by score and tiebreakers
usort($players, function($a, $b) use ($tiebreakers)
{
    //compare score
    if($a["score"] == $b["score"]){

        // check if first tiebreaker exists and compare players based on first tiebreaker rule
        if (array_key_exists(0, $tiebreakers)){
            if($a[$tiebreakers[0]["rule_code"]] == $b[$tiebreakers[0]["rule_code"]]){

                if (array_key_exists(1, $tiebreakers)){
                    if($a[$tiebreakers[1]["rule_code"]] == $b[$tiebreakers[1]["rule_code"]]){

                        if (array_key_exists(2, $tiebreakers)){
                            if($a[$tiebreakers[2]["rule_code"]] == $b[$tiebreakers[2]["rule_code"]]){

                                if (array_key_exists(3, $tiebreakers)){
                                    if($a[$tiebreakers[3]["rule_code"]] == $b[$tiebreakers[3]["rule_code"]]){
                                        return (0);
                                    }

                                    return (($a[$tiebreakers[3]["rule_code"]] < $b[$tiebreakers[3]["rule_code"]]) ? 1 : -1);

                                } else {
                                    return (0);
                                }
                            }

                            return (($a[$tiebreakers[2]["rule_code"]] < $b[$tiebreakers[2]["rule_code"]]) ? 1 : -1);

                        } else {
                            return (0);
                        }

                    }

                    return (($a[$tiebreakers[1]["rule_code"]] < $b[$tiebreakers[1]["rule_code"]]) ? 1 : -1);

                } else {
                    return (0);
                }

            }

            return (($a[$tiebreakers[0]["rule_code"]] < $b[$tiebreakers[0]["rule_code"]]) ? 1 : -1);

        } else {
            return (0);
        }

        return (($a["score"] < $b["score"]) ? 1 : -1);
```

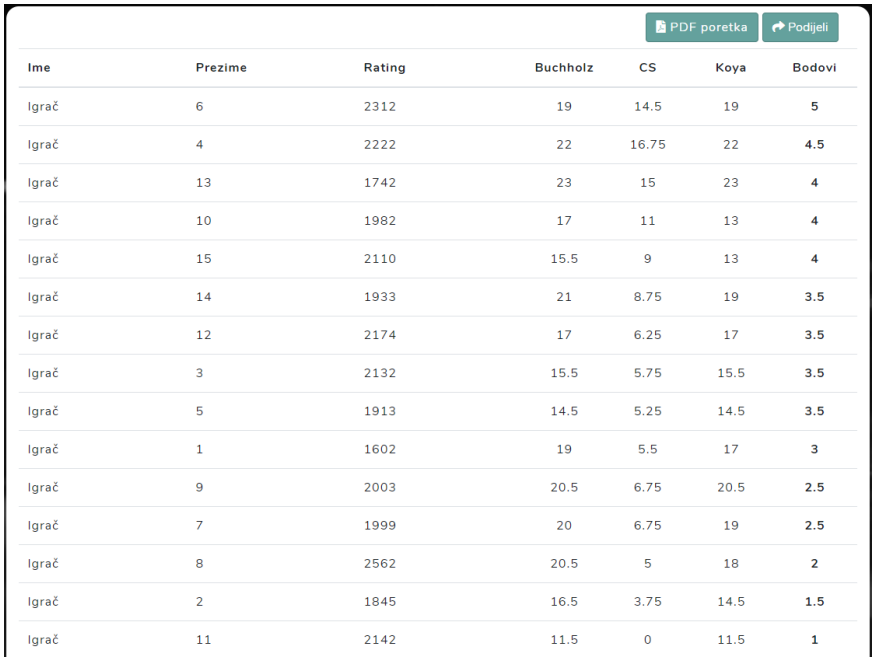
Slika 4.18. – *Određivanje trenutnog poretka*

Nakon što je unesen rezultat i potvrđeni su rezultati trenutnoga kola automatski se osvježava stranica turnira te su korisniku vidljivi parovi sljedećeg kola i omogućen je unos rezultata (Sl. 4.19.).



Slika 4.19. – *Rezultati prethonog kola i unos rezultata za trenutno kolo*

Na temelju unesenih rezultata automatski se kreira trenutni poredak. Ako je unesen rezultat za posljednje kolo automatski se završava turnir. Primjer konačnog poretka na turniru vidljiv je na slici 4.20.



Ime	Prezime	Rating	Buchholz	CS	Koya	Bodovi
Igrač	6	2312	19	14.5	19	**5**
Igrač	4	2222	22	16.75	22	**4.5**
Igrač	13	1742	23	15	23	**4**
Igrač	10	1982	17	11	13	**4**
Igrač	15	2110	15.5	9	13	**4**
Igrač	14	1933	21	8.75	19	**3.5**
Igrač	12	2174	17	6.25	17	**3.5**
Igrač	3	2132	15.5	5.75	15.5	**3.5**
Igrač	5	1913	14.5	5.25	14.5	**3.5**
Igrač	1	1602	19	5.5	17	**3**
Igrač	9	2003	20.5	6.75	20.5	**2.5**
Igrač	7	1999	20	6.75	19	**2.5**
Igrač	8	2562	20.5	5	18	**2**
Igrač	2	1845	16.5	3.75	14.5	**1.5**
Igrač	11	2142	11.5	0	11.5	**1**

Slika 4.20. – *Tablica konačnog poretka*

5.ZAKLJUČAK

U ovom završnom radu izrađena je aplikacija za sparivanje igrača na turnirima. Aplikacija koristiti „Swiss“ ili „round robin“ način sparivanja. „Round robin“ sustav je pošteniji način određivanja konačnog poretka budući da svaki igrač morati igrati protiv svakog igrača međutim ako je na turniru prijavljen prevelik broj igrača možemo koristiti „Swiss“ sustav sparivanja protivnika. Korisnik ima mogućnost odabira načina sparivanja, unosa igrača i njihovog ratinga, odabira broja kola za „Swiss“ sustav. Nadalje, aplikacija za sparivanja igrača omogućuje jednostavan unos rezultata i praćenje poretka igrača nakon svakog kola na temelju rezultata. Izrada aplikacije zahtijevala je detaljno planiranje kako bi se mogla kreirati odgovarajuća baza podataka i struktura projekta.

Za izradu aplikacije korišteni su HTML, CSS, Bootstrap, JavaScript, jQuery, Ajax, PHP i Laravel, a za bazu podataka korišten je MySQL.

LITERATURA

[1] XAMPP informacije, kolovoz 2019.godine
https://www.apachefriends.org/about.html

[2] PHP, rujan 2019.godine
https://www.php.net/manual/en/intro-whatis.php

[3] R. Lerdorf, K. Tatroe i P. MacIntyre, Programiranje PHP, O'Reilly Media, Zagreb, 2015.

[4] MySQL, kolovoz 2019.godine
https://hr.wikipedia.org/wiki/MySQL

[5] M. Stauffer, Laravel Up & Running, O'Reilly Media, Sebastopol, 2017.

[6] HTML informacije, kolovoz 2019.godine
https://hr.wikipedia.org/wiki/HTML

[7] E.A.Meyer, CSS: The Definitive Guide (3rd ed.), O'Reilly Media, Sebastopol, 2006.

[8] Usage of JavaScript libraries for websites, kolovoz 2019.godine
https://w3techs.com/technologies/overview/javascript_library/all

[9] J. Albert i R. H. Koning, Statistical Thinking in Sports, Chapman & Hall / CRC, 2007.

[10] First-move advantage in chess, kolovoz 2019.godine
https://w3techs.com/technologies/overview/javascript_library/all

[11] Olympics badminton: Eight women disqualified from doubles, kolovoz 2019.godine
https://www.thesprucecrafts.com/the-swiss-system-611537

[12] North American Word Game Players Association, NASPA Director Manual, Dallas TX, 2013.

[13] What is the Swiss System?, kolovoz 2019.godine
https://www.thesprucecrafts.com/the-swiss-system-611537

[14] T. Just i D. Burg, U.S. Chess Federation's Offical Rules of Chess, McKay, 2003.

[15] E. James, Chess For Dummies (2nd ed.), John Wiley & Sons, Hoboken, NJ, 2005.

[16] Laravel documentation version 5.8, rujan 2019.godine
https://laravel.com/docs/5.8

SAŽETAK

U ovom završnom radu izrađena je web aplikacije za sparivanje igrača pomoću „Swiss“ i „round robin“ algoritama za sparivanje igrača. Korisnici imaju mogućnost kreiranja vlastitog turnira i odabira načina sparivanja protivnika. Nakon svakog kola korisnik unosi rezultate te se automatski kreira sljedeće kolo i trenutni poredak. Za izradu aplikacije korišteni su HTML, CSS, Bootstrap, JavaScript, jQuery, Ajax, PHP i Laravel, a za bazu podataka korišten je MySQL.

Ključne riječi: Šah, Swiss, Round robin, Laravel, PHP

ABSTRACT

In this thesis web application that pairs up users using Swiss and round robin algorithm has been built. Users have opportunity to create their own tournament and to choose a method of pairing with a new opponent. When a round is finished, user inputs results and next round gets automatically generated as well as current rankings. For the creation of this application HTML, CSS, Bootstrap, JavaScript, jQuery, Ajax, PHP and Laravel were used, while MySQL was used for a database.

Keywords: Chess, Swiss, Round robin, Laravel, PHP

ŽIVOTOPIS

Davor Komljenović rođen je 17. lipnja 1995.godine u Vinkovcima. Od 2002. do 2010.godine pohađa Osnovnu školu Ivana Gorana Kovačića u Vinkovcima. Godine 2010. upisuje Tehničku školu Ruđera Boškovića u Vinkovcima koju završava 2014.godine. Akademske godine 2014/15 upisuje stručni studij Informatike na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija u Osijeku.

Davor Komljenović

_____