Web aplikacija za organizaciju, vođenje i pregled malonogometnih turnira

Baričević, Stjepan

Master's thesis / Diplomski rad

2017

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:200:448669

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-03-29**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

zir.nsk.hr

Image not found or type unknown

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

Sveučilišni studij

Web aplikacija za organizaciju, vođenje i pregled malonogometnih turnira

Diplomski rad

Stjepan Baričević

Osijek, 2017.

Sadržaj

1. UVOD 1
2. OPIS TURNIRA I ALGORITAM RASPOREĐIVANJA 2
2.1. MALONOGOMETNI TURNIRI 2
2.2. KORISNIČKI ZAHTJEVI I ULOGE KORISNIKA 5
2.3. ALGORITAM RASPOREĐIVANJA 6
3. KORIŠTENE TEHNOLOGIJE 11
3.1. ASP.NET 11
3.2. MSSQL 12
3.3. Entity framework i LINQ 13
3.4. AngularJs 14
3.5. Aplikacijski poslužitelji 14
4. POSLUŽITELJSKO PROGRAMSKO RJEŠENJE 15
4.1. Oblikovni obrazac i arhitektura 15
4.1.1. API 15
4.1.2 SERVISNI SLOJ I SLOJ REPOZITORIJA 16
4.1.3 STRUKTURA BAZE PODATAKA 17
5. KLIJENTSKO PROGRAMSKO RJEŠENJE 19
5.1. HTML 19
5.2. CSS 19
5.3. KORIŠTENJE KORISNIČKOG SUČELJA 21
6. ZAKLJUČAK 22
LITERATURA 23
SAŽETAK 24
ŽIVOTOPIS 26
PRILOZI 27

1. UVOD

U moderno vrijeme kada informatika i računarstvo eksponencijalno napreduju postoje mnoge tehnologije i tehnike koje programerima omogućuju izradu primjenjivih programa (engl. *application software*). Te tehnologije su postale vrlo bitan čimbenik u društvu. Osim poslovnih, razlog tomu je ponajviše olakšavanje i povećanje kvalitete života ljudi.

U radu će se često koristiti pojmovi aplikacije i internet aplikacije stoga ih je neophodno objasniti. Aplikacija ili primjenjivi program je najjednostavnije rečeno program koji omogućuje korisniku izvršenje određenog zadatka na uređaju. Postoji više vrsta aplikacija, a programsko rješenje ovog rada je internet aplikacija. To je podvrsta aplikacija kojima se pristupa preko internet preglednika. Može im se pristupiti bilo kada i bilo gdje ukoliko je korisnik povezan na internet mrežu.

Cilj ovog rada je izrada internet aplikacije za organizaciju, vođenje i pregled malonogometnih turnira. Odnosi se na ciljanu skupinu korisnika, kako i samo ime kaže, koji žele organizirati malonogometni turnir i imati uvijek uvid u stanje turnira. Vrste turnira za koje je aplikacija prilagođena su: doigravanje (engl. *playoff*), ligaški turnir (engl. *league*) i liga kup (engl. *league cup*).

Prvi dio sadrži detaljni prikaz problema koji je analiziran prije izrade programskog rješenja. Nakon toga objašnjen je algoritam pomoću kojeg je on riješen.

U drugom dijelu rada detaljnije će biti objašnjene tehnike i tehnologije korištene u programskom rješenju te zašto je baš takav pristup odabran.

U trećem dijelu biti će objašnjeno samo programsko rješenje na poslužiteljskoj i klijentskoj strani te najbitniji dijelovi koda i metode korištenje u izradi. Sadrži i najbitnije dijelove grafičkog sučelja aplikacije u vidu slika.

Na kraju slijedi zaključak u kojem se nalazi osvrt na cijeli rad, mogućnost poboljšanja programskog rješenja i novih spoznaja.

2. OPIS TURNIRA I ALGORITAM RASPOREĐIVANJA

2.1. MALONOGOMETNI TURNIRI

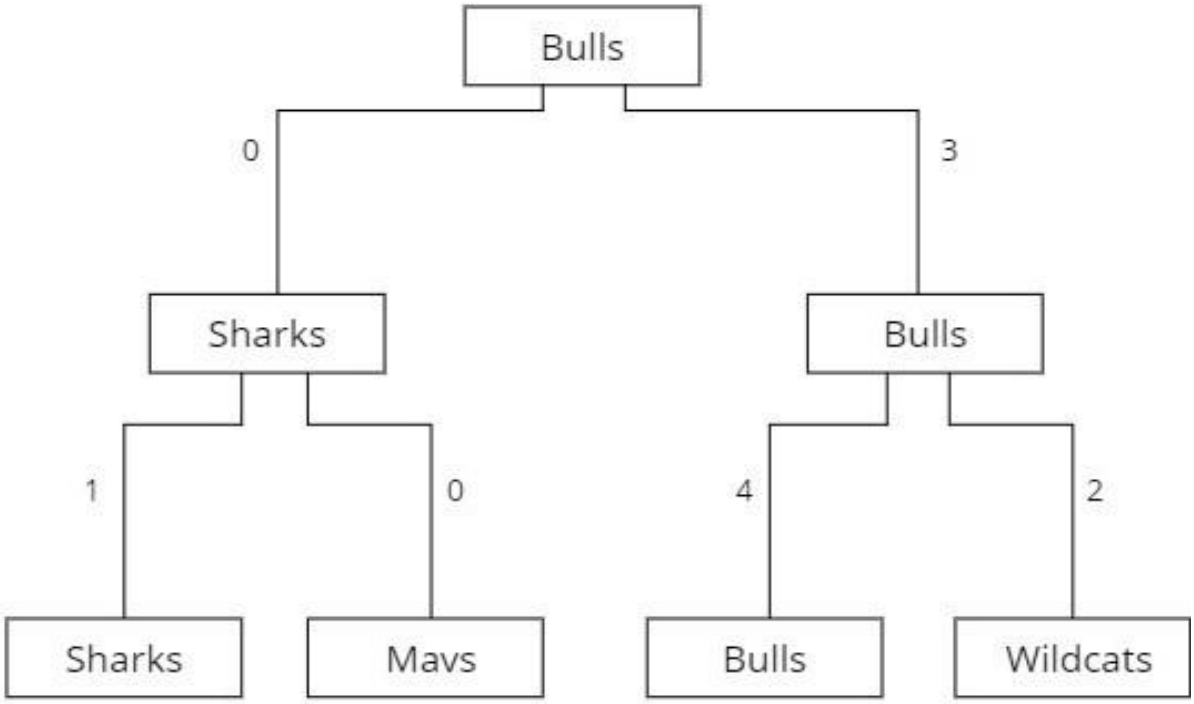
Nogomet kao sport je globalno poznat i gotovo da ne postoji mjesto gdje se ne igra. Isto tako razvijaju se inačice te igre u vidu malog nogometa i futsala. Bitno je da su te dvije vrste dva različita sporta. Ponajviše se razlikuju u pravilima. Npr. futsal ili dvoranski nogomet se igra s četiri igrača u polju i vratarom dok se mali nogomet igra s pet igrača i vratarom. Obje vrste ovog sporta su iznimno popularne. Procjenjuje se da se futsalom bavi oko trideset milijuna ljudi u barem sto zemalja svijeta.

Postoje različiti načini organizacije turnira. Primjerice neki turniri se igraju u ligaškom formatu, takozvane lige (engl. *league*), gdje svaka momčad protiv svake igra jednom ili više puta. Pobjednik je ona momčad koja skupi najviše bodova. Za pobjedu momčad dobije tri boda, za neriješen rezultat jedan bod, a za poraz nijedan. Ukoliko u ligaškom formatu postoje četiri momčadi svaka od njih mora odigrati ukupno tri utakmice ako se igra jednokružno, odnosno šest puta se igra dvokružno. Primjer tablice takvog formata vidljiv je na slici 2.1.

Tim	Odigrano	Pob.	Por.	Neod.	Bod.
Sharks	2	2	0	0	6
Mavs	2	1	1	0	4
Bulls	2	0	1	1	1
Wildcats	2	0	0	2	0

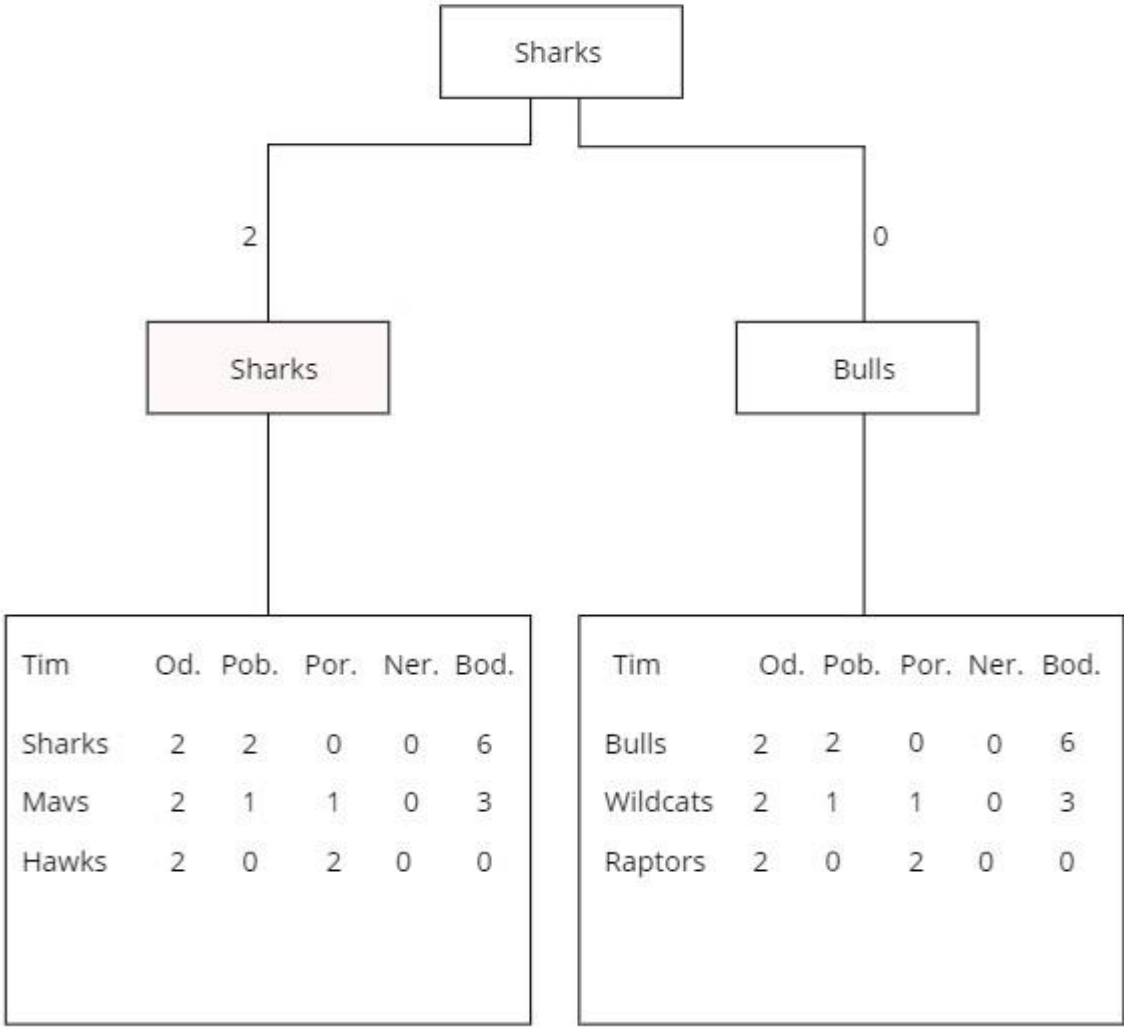
Sl. 2.1. Primjer rasporeda ligaškog tipa turnira

Postoje i turniri koji se igraju na izbacivanje (engl. *playoffs*). Nazivaju se i kupovima. U takvom formatu najbolji način odigravanja je da ukupan broj momčadi bude djeljiv s brojem četiri. Igra se po kolima. Ako broj momčadi nije djeljiv s četiri potrebno je neku ili više momčadi osloboditi utakmice u u kolu u kojem su višak te se ona izravno plasira u daljnju fazu natjecanja. Primjerice ako je broj momčadi u turniru četiri, mogu igrati prva i druga te treća i četvrta. To je prvo kolo i pobjednici tih kola igraju međusobno za ukupnog pobjednika. Veći problem je rasporediti primjerice pet momčadi. Mogu igrati prva i druga, treća i četvrta, a peta je slobodna. U iduće kolo prolaze pobjednici prvog kola, dakle dvije momčadi i ona koja je bila slobodna. Od te tri momčadi dvije igraju međusobno, a jedna je slobodna i prolazi direktno u finale gdje igra s pobjednikom utakmice iz prijašnjeg kola. Ukupno gledajući pravilo je da prvome kolu svaka momčad igra protiv jednog protivnika te ako pobijedi plasira se u iduće kolo i igra protiv nekog drugog pobjednika. Tako se igra dok jedna momčad ne pobijedi. Momčad kada izgubi prvu utakmicu ispada s turnira.



Sl. 2.2. Primjer rasporeda turnira koji se igra na izbacivanje

Treći, ujedno i najpoznatiji format odigravanja turnira je grupni i zatim na izbacivanje (engl. *league cup*). Sastoji se od elemenata prve dvije vrste. Kod ovakvog tipa turnira momčadi prvo igraju grupnu fazu. Svaka grupa predstavlja jednu podgrupu u turniru gdje svaka momčad igra protiv svake jednom. Većinom najbolja momčad ili najbolje dvije prolaze u daljnu fazu natjecanja gdje se igra na izbacivanje dok se ne dobije ukupan pobjednik.



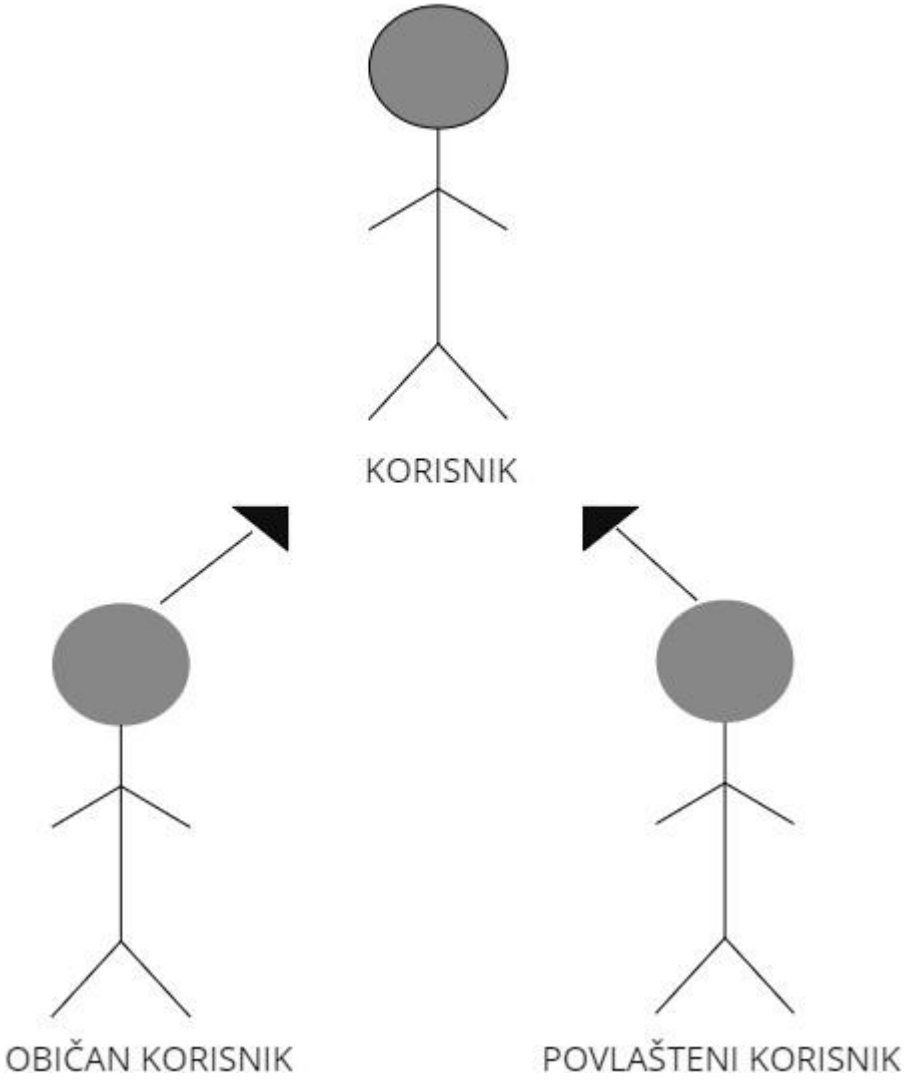
Sl. 2.3. Primjer liga kup rasporeda turnira

U početcima održavanja ovakvih turnira organizatori su se različito snalazili prilikom raspoređivanja utakmica, bilježenja momčadi i igrača, izračuna datuma odigravanja utakmica, vođenja statistika i sve ostalo što je potrebno kako bi se jedan malonogometni ili futsal turnir održali. Napretkom tehnologije nameće se pitanje zašto, kao i mnoštvo drugih stvari, ne bi računalo taj posao odradilo za organizatora? Smisao ove aplikacije leži u navedenome pitanju. Programsko rješenje u ovom radu naravno nije prvo takve vrste no u nekim elementima je bolja od postojećih rješenja koja postoje na internetu te se može koristiti u realnom vremenu.

2.2. KORISNIČKI ZAHTJEVI I ULOGE KORISNIKA

Iako je svaki turnir koji se održava zaseban u korijenu svi imaju iste elemente. Sastoje se od mjesta održavanja, tipa turnira, datuma početka, datuma završetka, timova koji igraju, igrača koji čine timove, sudaca, utakmica i na kraju rezultata i pobjednika. Svi ovi elementi predstavljaju zahtjeve za organizaciju i implementirani su u programsko rješenje.

Uloge korisnika u Internet aplikacijama mogu biti različite. Najčešće su to uloge običnih i povlaštenih korisnika gdje povlašteni mogućnosti ima mnogo veće mogućnosti od običnog.



Sl. 2.4. Uloge korisnika

U ovoj aplikaciji postoji samo jedna vrsta povlaštenog korisnika. To je organizator turnira. Organizator se prvo registrira i potom se prijavljuje. Kada se prijavi može kreirati nove turnire i nakon toga uređivati i brisati turnire koje je dodao. Ima mogućnost ručnog ili automatskog raspoređivanja utakmica turnira, dodavanja lokacije, sudaca, timova, igrača, tipa turnira itd. Nakon što doda osnovne elemente turnira može dodavati, brisati i mijenjati rezultate po kolima. Korisnici koji se ne registriraju, odnosno posjetitelji portala mogu pogledati najave

turnira, turnire u toku, najbolje igrače i timove, detalje i rezultate turnira te slike s turnira koje su unijeli organizatori. Svaki posjetitelj portala može se registrirati na portal i koristiti ga.

2.3. ALGORITAM RASPOREĐIVANJA

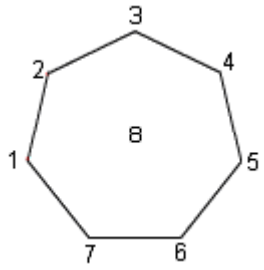
Kao što je navedeno postoje različiti tipovi turnira. Svaki tip turnira potrebno je drugačije rasporediti i svaki ima poseban algoritam, tj. drugačiji postupak raspoređivanja elemenata jedne utakmice. Ligaški tipovi turnira se mogu podijeliti u dvije osnovne kategorije raspoređivanja. To su raspoređivanje kada ima paran broj momčadi i kada ima neparan broj momčadi. Za raspoređivanje ligaških turnira najpoznatiji je takozvani „*round-robin*“ postupak.

POSTUPAK RASPOREĐIVANJA LIGAŠKOG TURNIRA S PARNIM BROJEM MOMČADI:

Kod parnog broja momčadi u jednokružnom odigravanju broj kola se dobije tako da se od broja momčadi oduzme broj jedan. Svaka momčad igra u svakom kolu i nijedna momčad nije slobodna tijekom odigravanja svih kola. Broj utakmica po kolu se dobije dijeljenjem broja momčadi s dva. Na primjer, ako se na turnir prijavi osam momčadi broj kola je sedam. Svaka momčad igra sedam puta i nije slobodna nijedno kolo. Broj utakmica u jednom kolu je četiri (osam podijeljeno s dva).

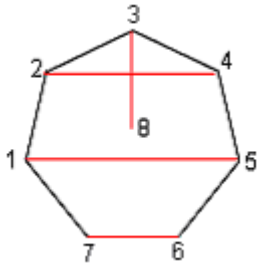
- U postupku je dan primjer s osam momčadi, brojevi od 1 do 8 predstavljaju momčadi

1. Poredati momčadi pomoću mnogokuta na način da je posljednji tim u centru mnogokuta, a svi ostali su poredani po vrhovima. Broj kuteva (vrhova) je jednak broju momčadi minus jedan.



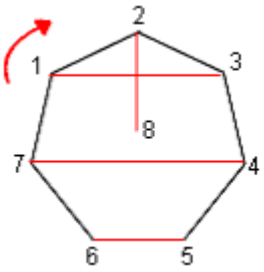
Sl. 2.5. Korak 1 postupka raspoređivanja ligaškog turnira s parnim brojem momčadi

2. Spariti momčadi koje se nalaze na suprotnim horizontalnim vrhovima mnogokuta. Vrh koji ostane spariti s centrom.



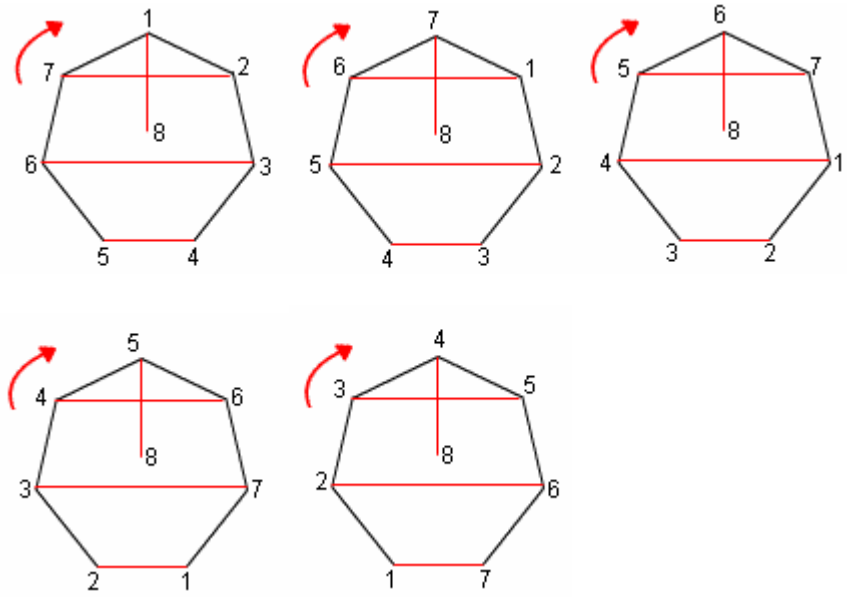
Sl. 2.6. Korak 2 postupka raspoređivanja ligaškog turnira s parnim brojem momčadi

3. Odrediti parove za pripadajuće kolo prema sparenim momčadima u prethodnom koraku. Po primjeru to su parovi (1,5), (2,4), (7,6) i (3,8).
4. Zarotirati mnogokut iz koraka 2 za jedno mjesto u desno. Momčad u centru se ne premješta.



Sl. 2.7. Korak 4 postupka raspoređivanja ligaškog turnira s parnim brojem momčadi

5. Spariti momčadi koje su dobivene rotiranjem(kao korak 2) i odrediti parove dobivene rotiranjem(kao korak 3).
6. Nastaviti rotaciju mnogokuta i određivati parove dok se ne dođe do pozicije iz koje se krenulo.



Sl. 2.8. Korak 6 postupka raspoređivanja ligaškog turnira s parnim brojem momčadi

Mogući rezultat postupka je:

Tab. 2.1. Mogući rezultat postupka raspoređivanja ligaškog turnira s parnim brojem momčadi

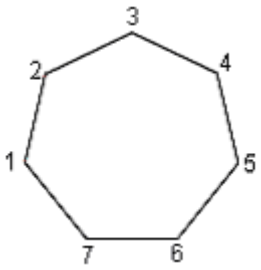
KOLO	PAR 1	PAR 2	PAR 3	PAR 4
1.	7,6	1,5	2,4	3,8
2.	6,5	7,4	1,3	2,8
3.	5,4	6,3	7,2	1,8
4.	4,3	5,2	6,1	7,8
5.	3,2	4,1	5,7	6,8
6.	2,1	3,7	4,6	5,8
7.	2,1	3,7	4,6	5,8

POSTUPAK RASPOREĐIVANJA LIGAŠKOG TURNIRA S NEPARNIM BROJEM MOMČADI:

Kod neparnog broja momčadi u jednokružnom odigravanju broj kola je jednak broju momčadi. Svaka momčad igra sa svakom drugom momčadi jednom i jedno kolo je slobodna. Broj utakmica po kolu dobije se tako da se od broja momčadi oduzme jedna i dobiveni broj se podijeli s dva. Na primjer, ako se na turnir prijavi sedam momčadi broj kola je sedam. Svaka momčad igra šest puta i točno jedno kolo je slobodna. Broj utakmica u jednom kolu je tri (sedam minus jedan i zatim se dijeli s dva).

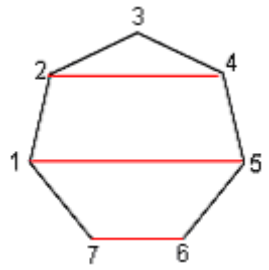
- U postupku je dan primjer sa sedam momčadi, brojevi od 1 do 7 predstavljaju momčadi

1. Poredati momčadi pomoću mnogokuta na način da se na svaki vrh mnogokuta postavi jedna momčad. Kuteva (vrhova) ima onoliko koliko je momčadi.



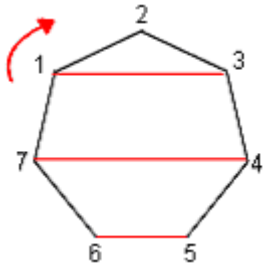
Sl. 2.9. Korak 1 postupka raspoređivanja ligaškog turnira s neparnim brojem momčadi

2. Spariti momčadi koje se nalaze na suprotnim horizontalnim vrhovima mnogokuta. Vrh koji ostane je momčad koja je to kolo slobodna.



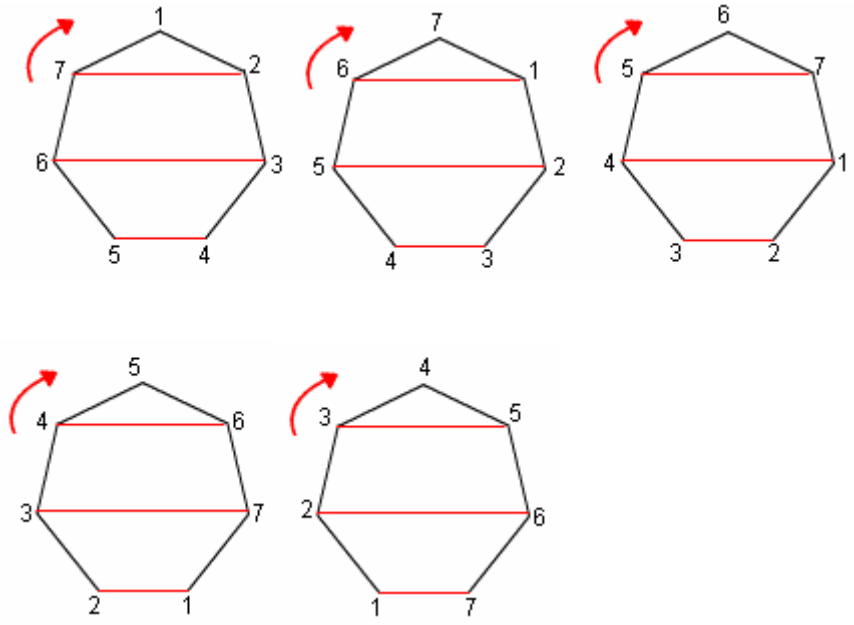
Sl. 2.10. Korak 2 postupka raspoređivanja ligaškog turnira s neparnim brojem momčadi

3. Odrediti parove za pripadajuće kolo prema sparenim momčadima u prethodnom koraku. Po primjeru to su parovi (1,5), (2,4) i (7,6). Momčad 3 je slobodna.
4. Zarotirati mnogokut iz koraka 2 za jedno mjesto u desno.



Sl. 2.11. Korak 4 postupka raspoređivanja ligaškog turnira s neparnim brojem momčadi

5. Spariti momčadi koje su dobivene rotiranjem(kao korak 2) i odrediti parove dobivene rotiranjem(kao korak 3).
6. Nastaviti rotaciju mnogokuta i određivati parove dok se ne dođe do pozicije iz koje se krenulo.



Sl. 2.12. Korak 6 postupka raspoređivanja ligaškog turnira s neparnim brojem momčadi

Mogući rezultat postupka je:

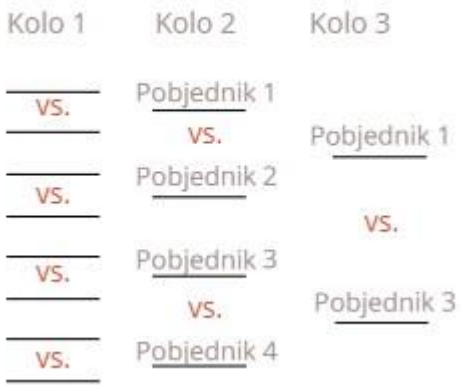
Tab. 2.2. Mogući rezultat postupka raspoređivanja ligaškog turnira s neparnim brojem momčadi

KOLO	PAR 1	PAR 2	PAR 3
1.	7,6	1,5	2,4
2.	6,5	7,4	1,3
3.	5,4	6,3	7,2
4.	4,3	5,2	6,1
5.	3,2	4,1	5,7
6.	2,1	3,7	4,6
7.	1,7	2,6	3,5

Tipovi turnira koji se igraju kao kup raspoređuju se na drugačiji način. Poznato je da poraz u ovakvom formatu donosi eleminaciju s turnira. Raspoređivanje je jednostavno ako je broj momčadi dva, četiri, osam, šesnaest, trideset dva, itd. No, što ako broj momčadi nije djeljiv s četiri(ne uključujući dva i tri). Ovdje zapravo nema nekog posebnog algoritma. U ovo radu je korisniku platforme odnosno organizatoru turnira dana restrikcija. Ono što korisnik mora, ako odabere kup turnir za tip turnira, je unijeti četiri, osam, šesnaest ili trideset dvije momčadi. Ovdje se postavlja pitanje što ako korisnik želi unijeti npr. dvanaest momčadi. Cilj je doći do prvog manjeg predodređenog broja – osam. Npr. korisnik može odrediti kvalifikacije za osam momčadi dok bi četiri bile slobodne. Od osam momčadi iz kvalifikacija dobije se četiri pobjednika te se oni mogu nadodati preostale četiri slobodne momčadi. U tom slučaju preostaje ukupno osam momčadi(iz kvalifikacija i slobodne). Isto je i za sve ostale brojeve momčadi. Cilj je doći do zadanih brojeva momčadi za kup, a hoće li se to dogoditi ždrijebom, kvalifikacijama ili nečim trećim na korisniku je.

POSTUPAK RASPOREĐIVANJA KUP TURNIRA:

1. Broj kola je jednak potenciji broja dva od broja momčadi.
    a. Ako je broj momčadi četiri broj kola je dva.
    b. Ako je broj momčadi osam broj kola je tri.
    c. Ako je broj momčadi šesnaest broj kola je četiri.
    d. Ako je broj momčadi trideset dva broj kola je pet.
2. U iduće kolo prolaze samo pobjednici.
3. Igra se dok u turniru ne ostanu samo dvije momčadi i jedna od njih pobijedi.



Sl. 2.13. Primjer raspoređivanja kup tip turnira

Posljednji tip turnira je grupni i zatim kup. Kao što je navedeno u prvom potpoglavlju ovakvi tipovi turnira se prvo igraju u grupama i nakon toga u kup formatu. Ako organizator želi organizirati ovakav tip turnira prvo mora dodati grupe. Kada se rasporede i odigraju po principu objašnjenom na početku ovog potpoglavlja (ligaški turniri s parnim i neparnim brojem momčadi) organizator odabire hoće li u nakon grupne faze ostati samo pobjednici grupa ili dvije najbolje plasirane momčadi u grupi. Kada to odabere aplikacija rasporedi preostale parove kako je objašnjeno u postupku raspoređivanja kup turnira dok se ne dobije pobjednik.

3. KORIŠTENE TEHNOLOGIJE

3.1. ASP.NET

ASP.NET je razvojna okolina napravljena od strane Microsofta kako bi programerima omogućila izradu dinamičkih *web* stranica, *web* aplikacija i *web* servisa.[1] Drugim riječima to je razvojna platforma koja služi za jednostavnu i učinkovitu izradu internet aplikacija koristeći dinamičke *web* stranice. Prva verzija objavljena je 2002. godine s .NET razvojnom okolinom kao nasljednik Microsoftove poslužiteljski skriptni jezik (engl. *active server pages, ASP*) tehnologije. Kao dio .NET razvojne okoline za razvoj aplikacije u ASP.NET-u moguće je koristiti bilo koji programski jezik koji se izvodi na zajedničkoj jezičnoj okolini (engl. c*ommon language runtime, CLR*). Primjeri programskih jezika koji se mogu izvoditi na CLR-u su: Visual Basic, Visual C++ i C#. Najnovija verzija .NET okoline uključujući ASP.NET i CLR je 4.7. Detalji izvođenja programskog koda na CLR-u biti će objašnjeni kasnije. Za izradu rada korišteno je Microsoft Visual Studio 2015 Community razvojno okruženje.

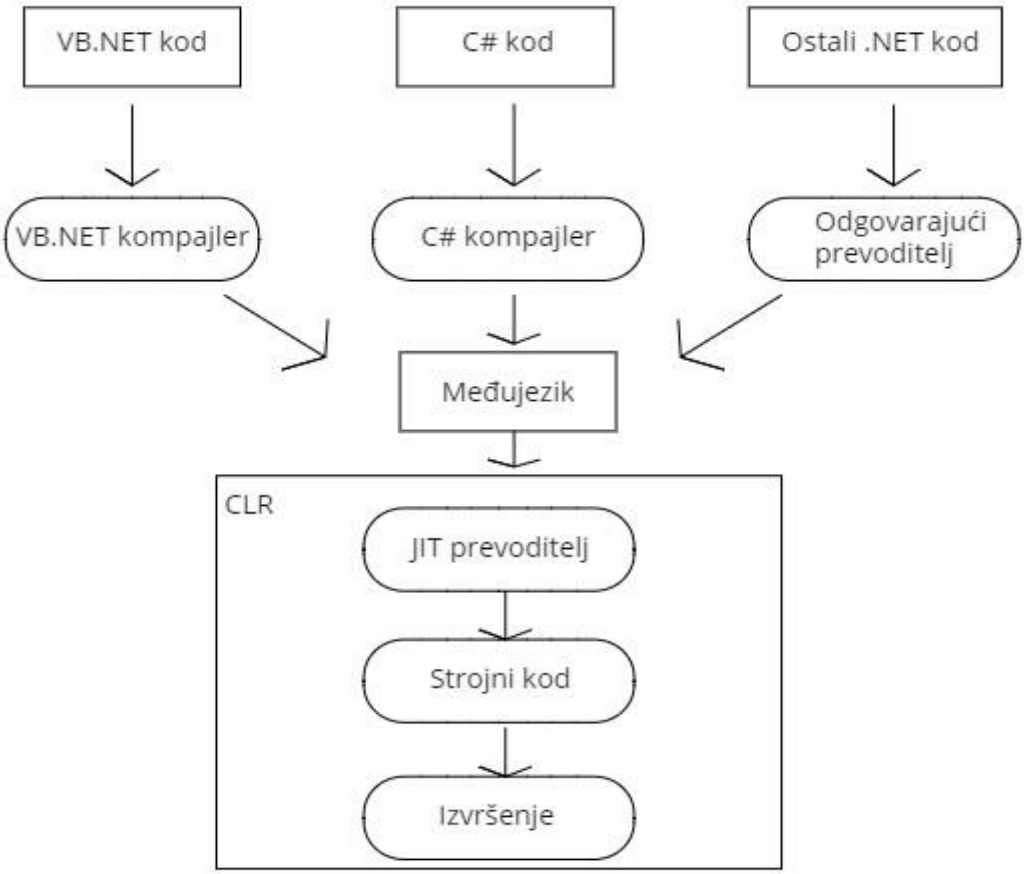
Dinamičke *web* stranice se generiraju na poslužiteljskoj strani nakon *http* zahtjeva klijentske strane. Poslužiteljska strana šalje stranicu u obliku *http* odgovora. Osim dinamičkih stranica poslužiteljska strana može sadržavati i statičke stranice. Prikaz odnosa poslužitelja i klijenta vidljiv je na slici 3.1.



Sl. 3.1. Prikaz http zatjeva i odgovora

Prednosti i razlozi za izradu ASP.NET razvojne okoline u odnosu na prvotni Microsoftov ASP su prije svega brzina i programski jezik. ASP skripte su se izvodile jako sporo i pisane su u jezicima poput VBScripta koji su imali potpuno drugačiju svrhu.

ASP.NET aplikacije se prevode u međujezik koji se zove Microsoft Intermeidate Language (MSIL) te se u trenutku izvršavanja aplikacije MSIL prevodi u strojni kod. Prevođenje iz MSIL-a u strojni kod se naziva „točno na vrijeme“ (engl. *just in time*) prevođenje, a odvija se kao i na cijelom .NET razvojnom okruženju. JIT prevođenje se odvija prvi i svaki put kada se naprave promjene u kodu. Prevođenje ne postoji kod zahtjeva za web stranicom.



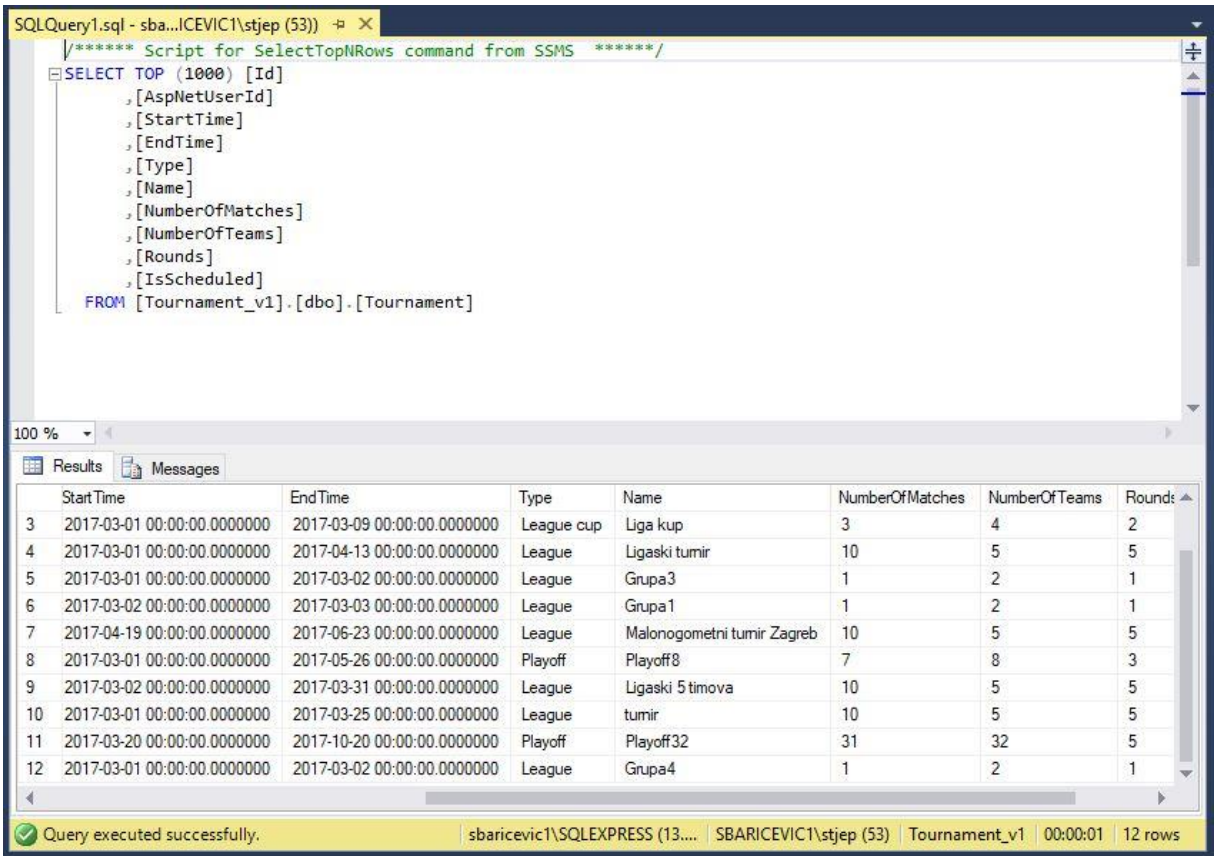
Sl. 3.2. Prevođenje ASP.NET aplikacije

3.2. MSSQL

Microsoft SQL Server (skraćeno MSSQL) je najnovije izdanje Microsoftove podatkovne platforme, koji za razliku od prijašnjih verzija sadrži nova svojstva i unaprjeđenja koja omogućuju bolje izvođenje, veću sigurnost te bogatije, ugrađene izvještaje i mogućnosti analiziranja.[2] MSSQL služi za za spremanje i primanje podataka na zahtjev od raznih aplikacija koje se mogu izvršavati na istom računalu na kojem se izvodi i MSSQL, ali i na različitom. Do sada su objavljena različita izdanja MSSQL-a. Posljednja stabilna verzija je SQL Server 2016 objavljena u lipnju 2016. Postoji više inačica MSSQL-a ovisno o namjeni i broju računala (korisnika) koriste bazu podataka. Neka od izdanja su:

- *Enterprise* – izdanje s najviše mogućnosti upravljanja bazom
- *Standard* – smanjene funkcionalnosti u odnosu na Enterprise izdanje
- *Express* – besplatna verzija, osnovne funkcionalnosti, smanjena memorija

MSSQL podržava različite tipove podataka kao što su *Integer, Float, Decimal, Char, Varchar, Binary, Text* i druge. Korisnik može sam definirati svoje tipove. Podaci se u bazu spremaju u datoteke s .mdf ekstenzijom. Podaci između klijenta i MSSQL-a se izmjenjuju preko Tabular data stream (TDS) protokola.



	StartTime	EndTime	Type	Name	NumberOfMatches	NumberOfTeams	Rounds
3	2017-03-01 00:00:00.0000000	2017-03-09 00:00:00.0000000	League cup	Liga kup	3	4	2
4	2017-03-01 00:00:00.0000000	2017-04-13 00:00:00.0000000	League	Ligaski turnir	10	5	5
5	2017-03-01 00:00:00.0000000	2017-03-02 00:00:00.0000000	League	Grupa3	1	2	1
6	2017-03-02 00:00:00.0000000	2017-03-03 00:00:00.0000000	League	Grupa1	1	2	1
7	2017-04-19 00:00:00.0000000	2017-06-23 00:00:00.0000000	League	Malonogometni turnir Zagreb	10	5	5
8	2017-03-01 00:00:00.0000000	2017-05-26 00:00:00.0000000	Playoff	Playoff8	7	8	3
9	2017-03-02 00:00:00.0000000	2017-03-31 00:00:00.0000000	League	Ligaski 5 timova	10	5	5
10	2017-03-01 00:00:00.0000000	2017-03-25 00:00:00.0000000	League	turnir	10	5	5
11	2017-03-20 00:00:00.0000000	2017-10-20 00:00:00.0000000	Playoff	Playoff32	31	32	5
12	2017-03-01 00:00:00.0000000	2017-03-02 00:00:00.0000000	League	Grupa4	1	2	1

Sl. 3.3. Prikaz jednostavne select naredbe i dohvaćenih rezultata

3.3. Entity framework i LINQ

Razvojna okolina entiteta (engl. *Entity framework*) je modelski odnosno mapiranje (engl. *Object-relational mapping*, ORM) tip razvojne okoline. ORM je tehnika koja omogućuje pretvaranje i mapiranje podataka između nekompatibilnih tipova varijabli, izraza, funkcija ili modula u objektno orijetiranim programskim jezicima i okolinama te omogućuje dizajn baze podataka i klasa u aplikaciji odvojenima. Takvom tehnikom spremanje podataka, koji su zapisani u domenskim objektima aplikacije, vrlo je jednostavno spremiti u relacijske baze podataka po principu da se klasa, odnosno nacrt objekta koji je korišten u aplikaciji, preslikava u tablicu u bazi podataka ili obrnuto. To omogućuje jednostavno održavanje i proširivost aplikacije i baze podataka.

Razvojna okolina entiteta programerima omogućuje jednostavan pristup podacima iz baze podataka putem LINQ upita (engl. *query*). LINQ upit je komponenta koja se može koristiti u .NET programskim jezicima kao dodatak izraza. Sadrži većinu standardnih SQL upita za bazu kao što su: *Select, Where*, *Join*, *OrderBy*, *Contains*, *Count* itd.

Razvojna okolina entiteta se može koristiti u tri scenarija. Prvi je kada se aplikacija izrađuje ili s obzirom na postojeću bazu podataka ili se dizajn aplikacije želi uvjetovati dizajnom baze. Takav pristup se naziva baza podataka prvo (engl. *database first*) pristup.

Drugi scenarij korištenja razvojna okoline entiteta je fokusiranje na izradu domenskih klasa aplikacije i zatim generiranja tablica baze s obzirom na te klase. Takav pristup se naziva kôd prvo *(*engl. *code first)* pristup.

Treći scenarij je izrada modela putem vizualnog dizajnera te generiranja domenskih klasa i tablica baze iz izrađenih modela. Takav pristup se naziva model prvi *(*engl. *model first)* pristup.

Do sada navedene tehnologije u ovom poglavlju korištene su isključivo na poslužiteljskoj strani aplikacije.

3.4. AngularJs

AngularJS je *JavaScript* razvojna okolina. Pojednostavljuje razvijanje web aplikacija koje se pokreću na korisničkoj strani u korisnikovu internet pretraživaću. Samim razvojem ove okoline uvelike je pojednostavljen razvoj jednostraničnih aplikacija (engl. *single page application*). U ovoj razvojnoj okolini *html* kod se šalje samo jednom te aplikacija ovisno o *web* adresi manipulira elementima stranice. Objavljen je 2010. godine. Od tada je izašlo nekoliko verzija, a najnovija je *Angular2* i od te verzije razvojni okvir je napisan od početka. Cilj *AngularJs*-a bio je pojednostavniti razvoj i testiranje jednostraničnih aplikacija razvijanjem razvojnog okvira koji prati MVC (engl. *Model View Controller*, MVC) oblikovni obrazac (engl. *design pattern*) što znači da se sastoji od modela, pogleda i kontrolera.

3.5. Aplikacijski poslužitelji

Aplikacijski poslužitelji u ovom kontekstu predstavljaju poslužitelje na kojima je aplikaciju moguće pokrenuti ili objaviti brzo i jednostavno. Radi se o internetskom informacijskom servisu (engl. *Internet Information Services*, IIS) Express verzija i samog IIS-a. IIS Express je samostalna djelomična verzija IIS-a optimizirana za programere.[3] Neke od bitnih karakteristika su: ne zahtijeva administratorska prava korisnika koji ga koristi na računalu, kompaktno radi s ASP.NET i PHP aplikacijama i više korisnika IIS Express-a mogu raditi neovisno na istom računalu. IIS je skup Internet poslužitelja (uključujući Web ili Hypertext Transfer Protocol - HTTP poslužitelj i File Transfer Protocol - FTP poslužitelj). Unutar IIS-a aplikacije se pokreću unutar aplikacijskih okruženja (eng. *application pool*). Aplikacijsko okruženje definira skup jednog ili više procesa (eng. *worker process*), konfiguriranih s zajedničkim postavkama koji poslužuju zahtjeve jedne ili više aplikacija sadržanih u danom aplikacijskom okruženju. Aplikacijsko okruženje koristi se da bi izolirali web aplikacije zbog bolje sigurnosti, dostupnosti i performansi. Radnički proces služi kao procesna granica koja odvaja svako aplikacijsko okruženje tako da u trenutku kada jedan proces ili aplikacija ima smetnju ili problem, ostale aplikacije ili radnički procesi nastavljaju normalnim tijekom rada. Po početnoj konfiguraciji aplikacije se pokreće na lokalnom IIS Express poslužitelju, a moguće ju je i objaviti na IIS-u.

4. POSLUŽITELJSKO PROGRAMSKO RJEŠENJE

4.1. Oblikovni obrazac i arhitektura

Oblikovni obrazac je implementiran po model, pogled i upravitelj strukturi (engl. *Model, View, Controller*, MVC). MVC i njegove varijante dijele internet aplikaciju u tri odvojena dijela: model, pogled i upravitelj.[4] Koristi se u razvijanju programskih aplikacija za odvajanje pojedinih dijelova aplikacije u komponente ovisno o njihovoj namjeni. U ovom obrascu glavnu ulogu ima upravitelj koji služi kao posrednik unutar sve tri komponente obrasca. On dohvaća podatke iz modela, mijenja ih i obrađuje te ukoliko je potrebno šalje pogledu. Također, upravitelj može ažurirati i mijenjati pogled ovisno o podacima. Pogled je bilo kakav prikaz podataka putem korisničkog sučelja (engl. *graphical user interface*, GUI) koje dobije iz upravitelja. Podaci se mogu prikazivati kroz različite obrasce, dijagrame ili tablice. Model predstavlja podatke i implementira logiku domene podataka projekta. U većini slučajeva struktura modela je ista kao i tablica u bazi podataka čije podatke model predstavlja.

Arhitektura poslužiteljske strane temelji se na asinkronom sučelju za programiranje aplikacija (engl. *application programming interface*, API). Za tu svrhu korišteni su API upravitelji ugrađeni u ASP.NET razvojno okruženje. Oni čine najviši sloj ove aplikacije. Ispod upravitelja nalazi se servisni sloj (engl. *service layer*) pa sloj repozitorija (engl. *repository layer*). Najniži sloj aplikacije predstavlja baza podataka.

4.1.1. API

Sučelje za programiranje aplikacija u ovom radu predstavlja skup metoda i objekata koji se koriste za izmjenu podataka između klijentske i poslužiteljske strane. API razmjenjuje podatke putem JSON (engl. *javascript object notation*) formata podata. JSON je jednostavan i čovjeku razumljiv format prijenosa podataka i sastoji se od kombinacije ključ – vrijednost (engl. *key - value*) i nizova podataka (engl. *data arrays*). Na slici 4.1. nalazi se primjer JSON formata podataka.


```
{
    "IstekRokaDatum":"2017-12-11",
    "KnjigaId":"68623fda-83ac-4417-9c85-be48b61756ae",
    "KorisnikId":"68623fda-83ac-4417-9c85-be48b61756ae",
    "PosudenaDatum":"2017-11-11"
}
```


Sl. 4.1. Primjer JSON formata podataka

Klijentsko računalo koristeći korisničko sučelje šalje zahtjeve putem *http* protokola (engl. HyperText transfer protocol) prema poslužiteljskoj strani te mu poslužitelj odgovara u JSON formatu. Ukoliko zahtjevi za mijenjanjem podataka nisu ispravni ovaj sloj zahtjev odmah odbija. Kod ispravnih zahtjeva zahtjev se prosljeđuje u idući sloj. U oba slučaja API upravitelji vraćaju kod o statusu (engl. *status code*) zahtjeva. Primjer kodova o statusu nalazi se u tablici 4.2. Svoju namjeru klijentsko računalo predstavlja preko *http* glagola (engl. verbs). U tablici 4.1. nalaze se *http* glagoli korišteni u aplikaciji i njihov opis.

Tab. 4.1. Korišteni http glagoli

Glagol	Opis
Post	Dodavanje novih podataka
Put	Ažuriranje postojećih podataka
Get	Dohvaćanje skupa podataka
Delete	Brisanje podataka

Tab. 4.2. Korišteni kodovi o status

Kod	Ime	Značenje
200	OK	Zahtjev je uspješno zaprimljen i obrađen
400	NEPOZNAT ZAHTJEV	Nepoznat zahtjev (moguće da su podaci krivo formirani)
403	ZABRANJENO	Zahtjev je ispravan, ali korisnik vjerojatno nema prava za tu akciju
404	NIJE PRONAĐENO	Zahtjevani podaci nisu dostupni

4.1.2 SERVISNI SLOJ I SLOJ REPOZITORIJA

Servisni sloj (engl. *service layer*) služi kao posrednik ili usluga između sučelja za programiranje aplikacija i sloja repozitorija (engl. *repository layer*). U slučaju ispravnog zahtjeva upravitelj ga prosljeđuje do servisnog sloja pozivajući odgovarajuću metodu koja se dalje prosljeđuje u sloj repozitorija. Sloj repozitorija sadrži generički obrazac (engl. *generic pattern*) koji omogućuje pristup osnovnim metodama za upravljanje podacima neovisno kojem modelu, klasi ili sučelju pripadaju. Te osnovne metode nazivaju se i CRUD (engl. *create read update delete*) metode i služe za kreiranje, čitanje, ažuriranje i brisanje podataka. Ovo je jedini sloj koji komunicira s bazom podataka. To mu omogućuje razvojna okolina entiteta (engl. *entity framework*). U oba sloja sve klase su apstraktne (engl. *abstract*). To znači da implementiraju pripadajuća sučelja (engl. *interfaces*). Apstraktne su zbog olakšavanja pisanja testova za iste. Svaki od slojeva zasebno je asinkron i sadrži ubacivanje ovisnoti (engl. *dependency injection*, DI). To je tehnika koja omogućava prihvaćanje pritisnutog objekta na mjestu povućenog odjekta[5], odnosno jedan objekt može prikupljati ovisnosti o drugom objektu. Postoje različite vrste ubacivanja ovisnosti. U ovom radu je korišteno ubacivanje ovisnosti pomoću konstruktora. Ovaj tip ubacivanja ovisnosti poziva ovisnosti o drugim klasama kada se poziva konstruktor pripadajuće klase. Primjer takvog poziva nalazi se na slici 4.2.

```
– references | 0 exceptions
protected IMatchService MatchService { get; set; }
– references | 0 exceptions
protected ITournamentService TournamentService { get; set; }
– references | 0 exceptions
public MatchController(IMatchService service, ITournamentService tournamentService)
{
    this.MatchService = service;
    this.TournamentService = tournamentService;
}
```

Sl. 4.2. Primjer ubacivanja ovisnosti pomoću konstruktora

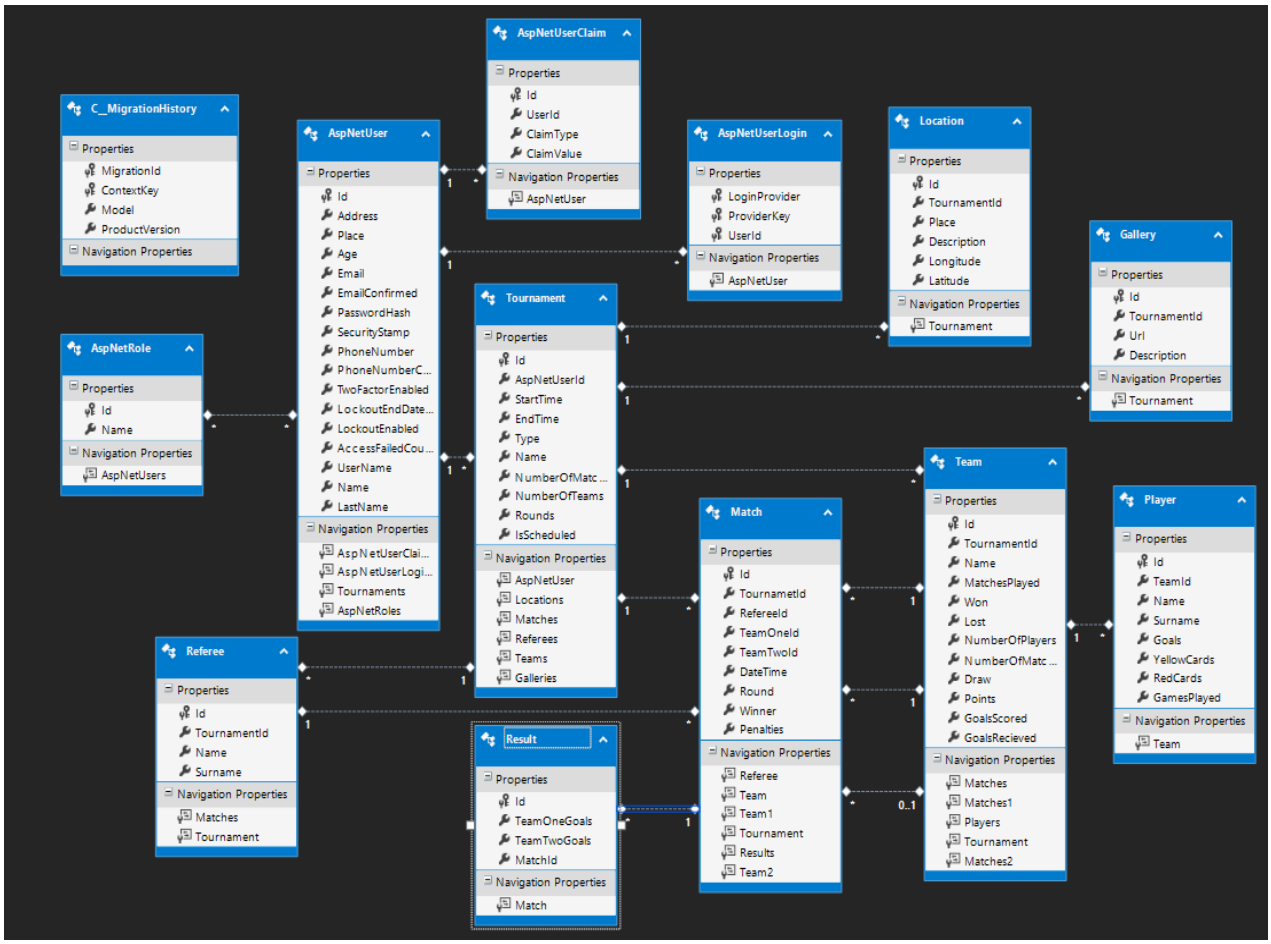
Slojevi aplikacije sadrže i različite modele podataka. Primjerice, sučelje za programiranje sadrži modele pogleda (engl. *view model*), a najniži, repozitorij sloj sadrži model baze podataka (engl. *database model*). Svaki od modela kroz slojeve ima istu strukturu no bitno je ne koristiti isti model kroz sve slojeve jer na taj način se omogućuje pristup bilo kojeg sloja podacima baze podataka što je potrebno izbjeći u svakoj aplikaciji neovisno o tehnologiji i arhitekturi. Izrada istih struktura modela omogućena je na jednostavan način preko automapiranja (eng. *automapping*). Za to je korišten *Automapper Nuget* paket. *Nuget* paketi su proširenja (engl. *extensions*) za Microsoft Visual Studio programska okruženja.

4.1.3 STRUKTURA BAZE PODATAKA

Fizički model baze podataka smješten je u SQL Server 2016 relacijskoj bazi koja služi za pohranu potrebnih i izvedenih podataka aplikacije. Naziv baze podataka je „Tournamet.db“. U tablici 4.3. nalazi se popis tablica iz baze podataka. Na slici 4.3. nalazi se prikaz tablica i njihovih veza u bazi.

Tab. 4.3. Imena i opis tablica baze podataka

Ime	Opis
AspNetUserClaim	Podaci o zahtjevima korisnika
AspNetUserLogin	Podaci o prijavama korisnika
AspNetUser	Podaci o korisniku – ime, prezime, korisničko ime, email, lozinka, …
AspNetUserRole	Podaci o ulozi korisnika
MigrationHistory	Podaci o migracijama baze
Tournament	Podaci o turniru – ime, tip, početak, kraj, …
Location	Podaci o lokaciji – mjesto, koordinate, opis,…
Gallery	Podaci o poveznici do slike i opis
Referee	Podaci o sudcu – ime, prezime, …
Match	Podaci o utakmici – vrijeme odigravanja, tim1, tim2, pobjednik, …
Result	Podaci o razultatu – golovi tima 1, golovi tima 2
Team	Podaci o timu – ime, pobijeđene, izgubljene, neriješene utakmice, bodovi, gol razlika,…
Player	Podaci o igraču – ime, prezime, broj golova, žuuti, crveni kartoni, …



Sl. 4.3. Tablice u bazi podataka i veze između njih

Tab. 4.4. Popis i objašnjenje veza između tablica

OPIS	OBJAŠNJENJE
*	Ne mora imati nijedan, a može imati više
1	Ima isključivo jedan
0..1	Ne mora imati nijedan ili ima isključivo jedan

5. KLIJENTSKO PROGRAMSKO RJEŠENJE

Na klijentskoj strani korišteni jezici i razvojne okoline su HTML, CSS i AngularJs.

5.1. HTML

Hipertext jezik za označavanje (engl. *hypertext markup language,* HTML) je prezentacijski jezik za izradu internet stranica.[6] Jednostavnije rečeno HTML je običan tekstualni zapis određenih znakova (engl. *tags*) koje internet preglednik razumije i prikazuje. HTML datoteke sadrže .html ili .htm ekstenzije. Svaki HTML dokument sastoji se od „<html> </html>“ znakova između kojih se nalaze „<head> </head>“ i „<body> </body>“. Unutar „head“ znakova upisuju se podaci o drugim podacima tj. podaci bitni za konfiguraciju dokumenta. Unutar „body“ znakova upisuje se sadržaj i znakovi koje će se prikazati u pregledniku. Najnovija vezija HTML-a je HTML5 koja je i korištena u ovom radu. Primjer HTML dokumenta i izgleda u web pregledniku nalazi se na slici 4.4. i 4.5.

```
<!DOCTYPE html>
<html>
    <head>
        <title>Primjer</title>
    </head>

    <body>
        <p>Ovo je primjer</p>
    </body>
</html>

```

Sl. 4.4. Primjer html koda

Ovo je primjer

Sl. 4.5. Izgled primjera sa slike 4.4. u internet pregledniku

5.2. CSS

CSS (engl. *cascading style sheets*) je stilski jezik koji se rabi za oblikovanje, formiranje i stiliziranje HTML dokumenata.[7] CSS služi za uređivanje stranice i raspoređivanje elemenata stranice. Najnovija CSS verzija je *CSS3* i korištena je u ovom radu. Primjer upotrebljavanja CSS jezika prikazan je na slici 4.6. i 4.7.

```
<!DOCTYPE html>
<html>
    <head>
        <title>Primjer</title>
        <style type="text/css">
            #paragraph {
                color: red;
            }
        </style>
    </head>

    <body>
        <p id="paragraph">Ovo je primjer</p>
    </body>
</html>

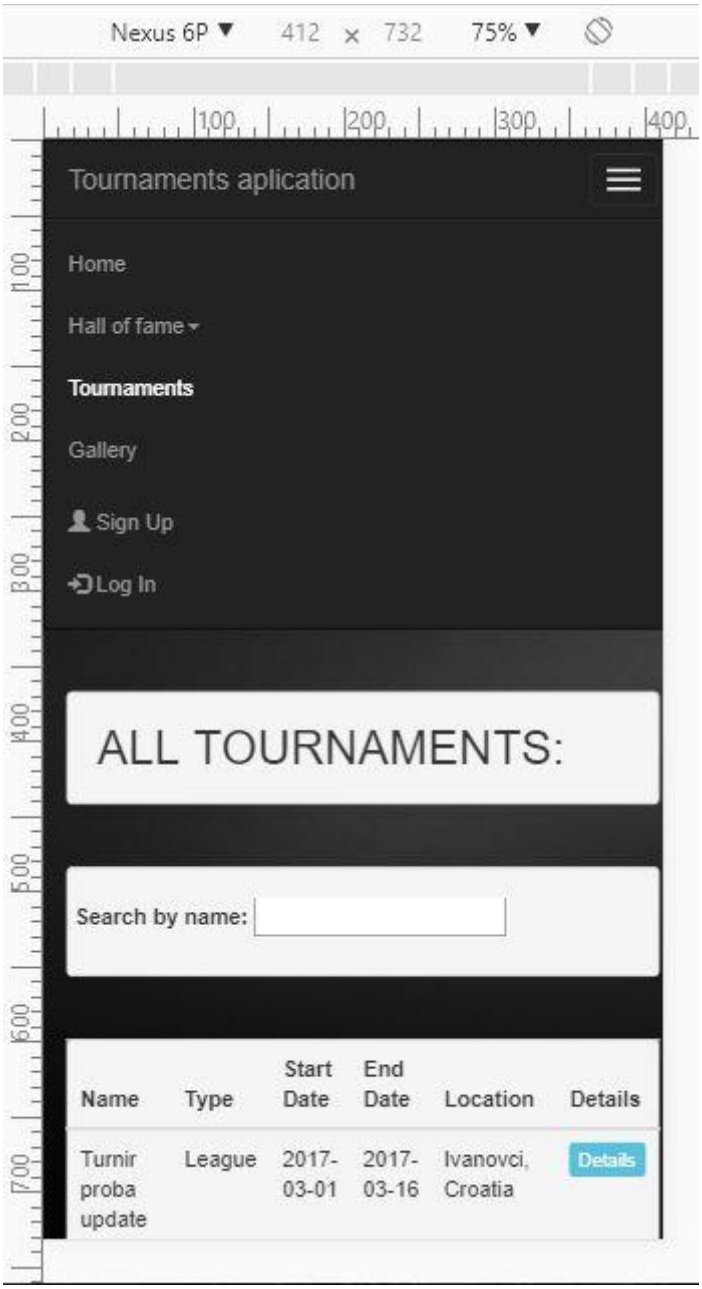
```

Sl. 4.6. Primjer html i css koda

Ovo je primjer

Sl. 4.7. Izgled primjera sa slike 4.6. u internet pregledniku

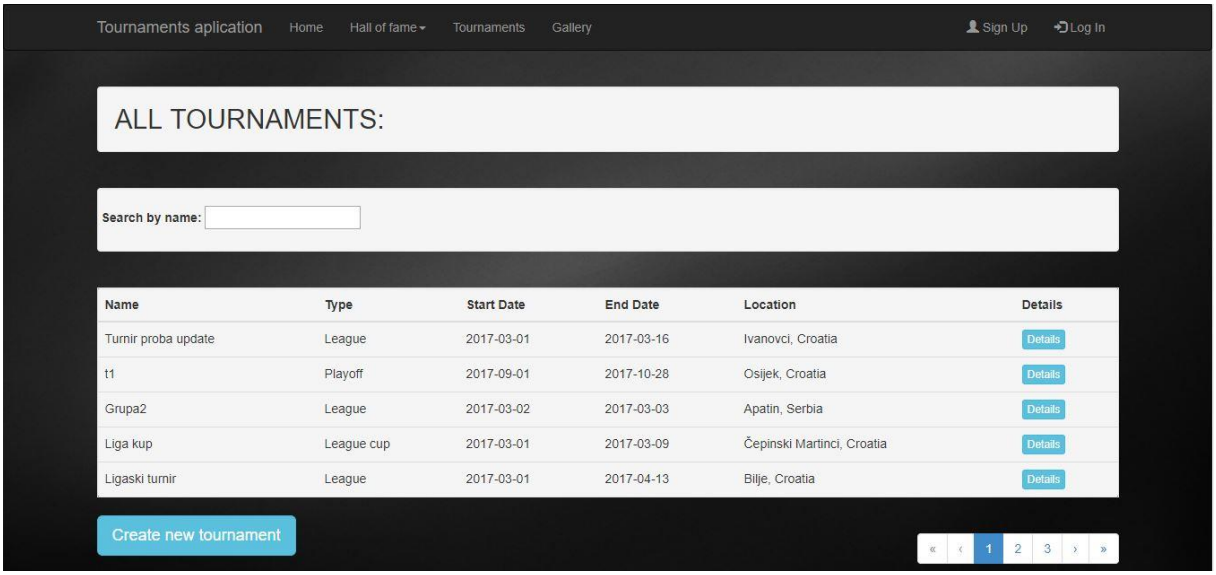
Kako bi aplikacija bila u potpunosti responzivna korišten je i *bootstrap*. Bootstrap je razvojno okruženje za dizajniranje web stranica i bazira se na HTML-u i CSS-u. Verzija korištena u ovom radu je Bootstrap3. *AngularJs* razvojna okolina je objašnjena u poglavlju 3.4. u ovom radu. Slika 4.8. prikazuje responzivnost stranice na mobilnom uređaju dok se korisničko sučelje na većim(računalnim) ekranima može vidjeti na slici 4.9.



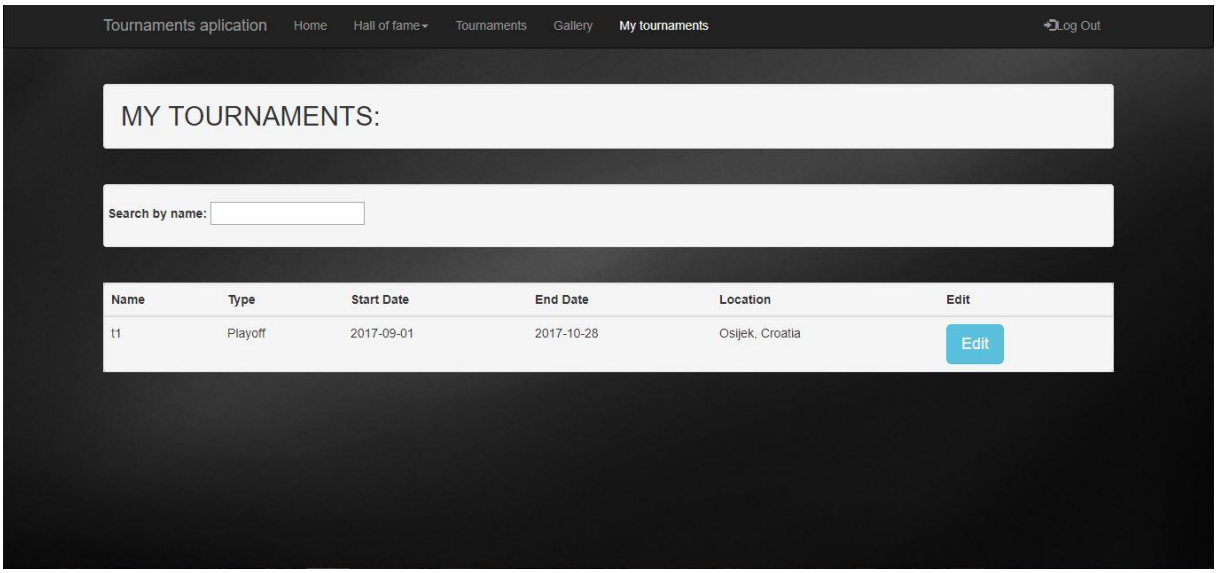
Sl. 4.8. Primjer responzivnosti stranice

5.3. KORIŠTENJE KORISNIČKOG SUČELJA

Aplikacija je izrađena kao javna aplikacija kojoj mogu pristupiti prijavljeni i neprijavljeni korisnici. Neprijavljeni imaju mogućnost pregleda najava turnira, najboljih igrača i timova, svih turnira i galerije te se mogu registrirati. Korisnici mijenjaju mogućnosti aplikacije putem navigacijske trake (engl. *navigation bar*). Nakon prijave korisnik ima mogućnost uređivanja svojih turnira.



Sl. 4.9. Primjer izgleda sučelja neprijavljenog korisnika



Sl. 4.10. Primjer izgleda sučelja prijavljenog korisnika

6. ZAKLJUČAK

Malonogometni turniri su globalni sport koji se igra u svim krajevima svijeta. U nekim dijelovima svijeta tradicija je da se igra s pet igrača, a u nekom da se igra sa šest igrača u polju. Isto tako pravila nisu uvijek ista. Postoje određeni algoritmi za olakšavanje problema organizatora u vidu raspoređivanja kola no u većini slučajeva i raspoređivanje i pregled cjelokupnog turnira vode se u papirnatom obliku.

Ovaj rad je nastao iz ideje da se omogući digitalni zapis svih turnira, neovisno o lokaciji, tipu turnira, broj igrača i ostalim parametrima zajedničkim parametrima turnira čime bi se kvaliteta praćenja turnira od strane navijača dovela na veću razinu. Rezultat rada je da je napravljena web aplikacija sa svim osnovnim mogućnostima nekog turnira. Prostora za napredak ima, a na to bi najviše utjecala povratna veza (engl. *feedback*) korisnika te bi se kroz održavanje aplikacije i objavljivanjem novih verzija kvaliteta cijele aplikacije poboljšala.

LITERATURA

[1] H.P. Halvorsen, ASP.NET Web programming, University College of Southeast Norway, Norway, 2016. dostupno na: http://home.hit.no/~hansha/documents/microsoft.net/tutorials/asp_net_web_programming/ASP.NET%20and%20Web%20Programming.pdf posjećeno: 5.9.2017.

[2] S. Varga, D. Cherry, J. D'Antoni, Introducing Microsoft SQL Server 2016, Microsoft Press, Washington, 2016. dostupno na:

http://csharpcorner.mindcrackerinc.netdnacdn.com/UploadFile/EBooks/11032016030502AM/PdfFile/introducing-microsoft-sql-server_2016.pdf posjećeno: 5.9.2017.

[3] V. Gopalakrishnan, IIS Express Overview, Microsoft, 2010. dostupno na: https://docs.microsoft.com/en-us/iis/extensions/introduction-to-iis-express/iis-express-overview posjećeno: 5.9.2017.

[4] N.Harrison, ASP.NET Succinctly, Syncfusion, Morrisville USA, 2015. dostupno na: http://files2.syncfusion.com/Downloads/Ebooks/ASP.NET_MVC_Succinctly.pdf posjećeno: 5.9.2017.

[5] Dependency injection patterns and practices, Microsoft, 2013. dostupno na: https://msdn.microsoft.com/en-us/library/dn178469(v=pandp.30).aspx posjećeno: 5.9.2017.

[6] L. Stevens, The Truth About HTML5 (For Web Designers), Indie Digital Pty Ltd, 2012.

[7] P. Shaw, CSS3 Succinctly, Syncfusion, Morrisville USA, 2015. dostupno na: https://hr.wikipedia.org/wiki/CSS pristupljeno 2.9.2017.

Ostale:

Glavaš I. ASP.NET tehnologija http://web.zpr.fer.hr/ergonomija/2003/glavas/03karakteristike.html pristupljeno 26.06.2017.

Entity framework, https://docs.microsoft.com/en-us/ef/ pristupljeno 26.06.2017.

SAŽETAK

U radu je implementiran algoritam za raspoređivanje različitih tipova turnira i mogućnost pregleda turnira kroz grafičko sučelje. Aplikacija je zamišljena kao javna aplikacija i mogu ju koristiti neregistrirani i registrirani korisnici. Neregistrirani imaju mogućnost pregleda turnira, a registrirani uz to imaju mogućnost organizacije vlastitih turnira. Za izradu poslužiteljske strane korištena je ASP.NET razvojna okolina, razvojna okolina entitety (engl. entity framework) i baza podataka u SQL Server 2016. Na korisničkoj strani korišteni su AngularJs razvojna okolina, HTML prezentacijski jezik i CSS stilski jezik. U teorijskom dijelu objašnjen je algoritam raspoređivanja, svaka korištena tehnologija zasebno, arhitektura i struktura aplikacije.

Ključne riječi: ASP.NET, AngularJs, malonogometni turniri, HTML, CSS

ABSTRACT

Web application for organization, management and review of futsal tournaments

An algorithm for dispositioning different types of tournaments and possibility of review tournament through graphical interface is implemented in this thesis. Application is imagined as public application and can be used by registered and unregistered users. Unregistered users can view tournament details and registered users have possibility to organize their own tournaments. Tools used for making server side are ASP.NET framework, entity framework and database in SQL Server 2016. On client side used tools are AngularJs framework, HTML markup language and CSS style sheet language. In the theoretical part, dispositioning algorithm, each used technology, architecture and structure of application are explained.

Keywords: ASP.NET, AngularJs, futsal tournaments HTML, CSS

ŽIVOTOPIS

Stjepan Baričević rođen je u Našicama 12. studenoga 1993. Osnovnu školu pohađao je i završio u Našicama u razdoblju od 2000. do 2008. godine. 2008. godine upisuje prirodoslovno matematičku gimnaziju u Srednjoj školi Našice i pohađa ju od 2008. do 2012. godine. Zatim upisuje preddiplomski studij na Elektrotehničkom fakultetu u Osijeku smjera računarstvo kojeg završava 2015. godine. Iste godine upisuje diplomski studij na spomenutom fakultetu, sada Fakultetu elektrotehnike, računarstva i informacijskih tehnologija, smjer računarstvo, izborni blok programsko inženjerstvo kojeg trenutno pohađa. Od svibnja 2017. radi kao student-razvojni programer u poduzeću GDi d.o.o.

Vlastoručni potpis:

_____

Stjepan Baričević

PRILOZI

Na CD-u:

1. Diplomski rad „Web aplikacija za organizaciju, vođenje i pregled malonogometnih turnira.docx“
2. Diplomski rad „Web aplikacija za organizaciju, vođenje i pregled malonogometnih turnira.pdf“
3. Izvorni kod aplikacije