Programska podrška za šah

Milas, Marko

Undergraduate thesis / Završni rad

2018

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Split, University of Split, Faculty of science / Sveučilište u Splitu, Prirodoslovno-matematički fakultet**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:166:806531

Rights / Prava: Attribution-NonCommercial-ShareAlike 4.0 International/Imenovanje-Nekomercijalno-Dijeli pod istim uvjetima 4.0 međunarodna

Download date / Datum preuzimanja: **2024-03-28**

Repository / Repozitorij:

Repository of Faculty of Science

zir.nsk.hr

UNIVERSITY OF SPLIT

dabar
DIGITALNI AKADEMSKI ARHIVI I REPOZITORIJI

SVEUČILIŠTE U SPLITU

PRIRODOSLOVNO-MATEMATIČKI FAKULTET

ZAVRŠNI RAD

PROGRAMSKA PODRŠKA ZA ŠAH

Marko Milas

Split, rujan 2018.

Temeljna dokumentacijska kartica

Završni rad

Sveučilište u Splitu
Prirodoslovno-matematički fakultet
Odjel za Informatiku
Ruđera Boškovića 33, 21000 Split, Hrvatska

PROGRAMSKA PODRŠKA ZA APLIKACIJU ŠAH

Marko Milas

SAŽETAK

Tema ovog rada je upoznavanje sa igrom Šah, njenim pravilima i povijesti, te terminom Programske podrške. Vidjet ćemo što znači programska podrška u kontekstu Šaha. Zatim slijedi upoznavanje sa jednom od trenutačno najpopularnijih web biblioteka koja se zove React, a nakon toga slijedi primjer implementacije programske podrške za Šah.

Ključne riječi: Šah, React, React Native, Programska podrška, aplikacija, igra

Rad je pohranjen u knjižnici Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu

Rad sadrži: 31 stranicu, 28 grafičkih prikaza, 0 tablica i 16 literaturnih navoda. Izvornik je na hrvatskom jeziku.

Mentor: **Izv.prof.dr.sc. Saša Mladenović,** *izvanredni profesor*
Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu

Neposredni voditelj:

Dr. sc. Goran Zaharija, *poslije-doktorand*
Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu

Ocjenjivači: **Izv.prof.dr.sc. Saša Mladenović,** *izvanredni profesor*
Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu

Dr. sc. Goran Zaharija, *poslije-doktorand*
Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu

Divna Krpan, *viši predavač*
Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu

Rad prihvaćen: **rujan 2018**

Basic documentation card

Thesis

University of Split
Faculty of Science
Department of Informatics
Ruđera Boškovića 33, 21000 Split, Croatia

SOFTWARE SUPPORT FOR CHESS GAME

Marko Milas

ABSTRACT

Purpose of this thesis is to show how game of Chess and software support can work together. Thesis first covers game of Chess, its rules, and goes throught Chess history. Then, thesis explains what is software support, and gives examples of how Chess software support usually looks like. In the end, thesis shows simple example of Chess application made with React Native.

Key words: Chess, React, React Native, Software support, application, game

Thesis deposited in library of Faculty of science, University of Split

Thesis consists of: 31 pages, 28 figures, 0 tables i 16 references.
Orginal Language: Croatian

Mentor: **Saša Mladenović,Ph.D,** *Associate Professor of Faculty of Science, University of Split*

Supervisor:

Goran Zaharija,Ph.D, *Postdoctoral Researcher of Faculty of Science, University of Split*

Reviewers: **Saša Mladenović,Ph.D ,** *Associate Professor of Faculty of Science, University of Split*

Goran Zaharija,Ph.D, *Postdoctoral Researcher of Faculty of Science, University of Split*

Divna Krpan, *Senior lecturer of Faculty of Science, University of Split*

Thesis accepted: **september 2018**

Sadržaj:

1 Uvod 1
2 Šah 2
2.1 Nastanak i razvoj šaha 2
2.2 Pravila igre 3
2.2.1 Kralj 3
2.2.2 Dama 6
2.2.3 Lovac 7
2.2.4 Skakač 8
2.2.5 Top 8
2.2.6 Pješak 9
3 Programska podrška 11
3.1 Sistemska programska podrška 11
3.2 Aplikacijska programska podrška 11
4 Programska podrška za Šah 12
5 Primjer programske podrške za Šah 13
5.1 React 13
5.2 React Native 14
5.3 Šah i React Native 15
5.3.1 Kreiranje korisnika 16
5.3.2 Glavni dio aplikacije 17
5.3.3 Izrada sučelja za igranje s protivnikom 20
5.3.4 Pregled otvorenih izazova 24
5.3.5 Implementacija različitih alata 26
5.3.6 Korisnikov profil 28
6 Zaključak 31
7 Popis literature 32

1 Uvod

Svrha ovog rada je upoznavanje sa programskom podrškom za aplikaciju Šah. U prvom poglavlju upoznat ćemo se sa igrom Šah. Upoznat ćemo se sa povijesti i pravilima šaha. Vidjet ćemo kako se svaka figura kreće i u kojim situacijama je ta figura poželjna.

Zatim ćemo u drugom poglavlju upoznat pojam programske podrške. Obradit ćemo pojmove Sistemske programske podrške i Aplikacijske programske podrške.

U trećem poglavlju vidjet ćemo kako se šahovska programska podrška mijenjala kroz povijest i koje su oznake moderne šahovske programske podrške.

U četvrtom poglavlju je obrađen primjer programske podrške za aplikaciju šah. Upoznat ćemo se sa modernom web bibliotekom *React*. Vidjet ćemo koje su osnovne ideje *React-a* i vidjeti kako se *React* može primijeniti i na mobilne uređaje u verziji koju nazivamo *React Native*.

Zatim ćemo implementirat primjer aplikacije za android uređaje izrađene u React Native-u. Vidjet ćemo razlike između *React-a* i *React Native-a*. U primjeru ćemo koristiti razne pakete i servise.

Koristit ćemo *Node.js* i *Express* kao poslužitelj na koji se spaja aplikacija i služit će nam za slanje poruka i poteza među igračima. Za kreiranje korisničkih računa, te za pohranu podataka koristit ćemo *Firebase*.

U primjeru ćemo implementirat sve što moderna programska podrška za šah treba imati.

2 Šah

Šah (perz. šāh: vladar, kralj, car), jedna je od najstarijih i najraširenijih igara na ploči. Šah je igra za dva igrača. Postoje 3 tipa: klasični (stolni šah), dopisni šah i problemski šah. Šahovska ploča je kvadratnog oblika, podijeljena u 64 (8×8) polja, obojanih naizmjenično svjetlom i tamnom bojom (redovno se govori "bijeli" i "crni"). Svaki igrač na početku igre ima 16 šahovskih figura, od toga osam pješaka, po jedan par lovaca, skakača i topova te kralja i kraljicu. Jedan igrač igra s figurama bijele, a drugi s figurama crne boje.

Igra s ovakvom pločom i figurama postoji barem od 6. stoljeća, ali pravila o kretanju figura mijenjala su se, pa su pravila konačno definirana tek u 19. stoljeću.

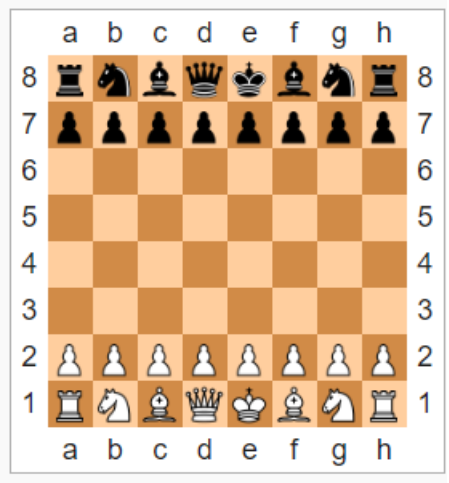
2.1 Nastanak i razvoj šaha

Prva je igra na dasci sa šahovskim elementima *čaturanga*, koja se u Indiji oko 600. razvila iz *aštapade*, igre na ploči s 8 × 8 polja. Igrala su ju četiri igrača, svaki s po 8 figura (predvodnik, slon, konj, bojna kola, 4 vojnika pješaka). Kockom se određivalo tko vuče potez. Svrha igre bila je uništiti sve protivničke figure. Iz te igre u Perziji je nastao *čatrang*, koji igraju 2 protivnika. Glavnu figuru nazivali su šah (kralj), a uveli su i novu figuru pod imenom farzin (savjetnik). Svrha je igre matiranje protivničkoga kralja. Od VIII. do X. st. Arapi su ju pod imenom *šatrandž* prenijeli u Europu. Kralj, top i skakač kretali su se kao i danas. Pješak se kretao po jedno polje naprijed, a uzimao jedno polje u koso. Slon se kretao u koso, s 1. na 3. polje, a mogao je preskakati figuru na 2. međupolju. Arapski firzan (dama) kretao se po polju u koso.

Igra se ubrzala nakon 1497. kada je *Juan Ramírez de Lucena* izdao knjigu u kojoj se dama i lovac vuku kao danas. Potez uzimanja pješaka *en passant* prihvaćen je potkraj XIX. st. Španjolski šahist *Ruy López de Segura* 1561. spominje današnju *rokadu*. Najjači šahisti svojega doba bili su Francuzi *François André Danican,* poznat kao *Philidor* (1726–95), *i L. Ch. M. de La Bourdonnais* (1795–1840) te Amerikanac *Paul Morphy* (1837–84) koji je 1858. pobjeđivao najbolje europske šahiste. Sat za ograničavanje trajanja razmišljanja u partiji uveden je 1883.

2.2 Pravila igre

Šah se igra na šahovskoj ploči (šahovnici) s osam redova (brojčanih oznaka 1 do 8) i osam stupaca (slovnih oznaka od a do h) tako da svako od 64 polja ima jedinstvenu oznaku, npr. a1, a2. U početnoj poziciji na šahovnici je 16 bijelih i 16 crnih figura – po jedan kralj i dama, po 2 topa, lovca i skakača te 8 pješaka (pijuna).



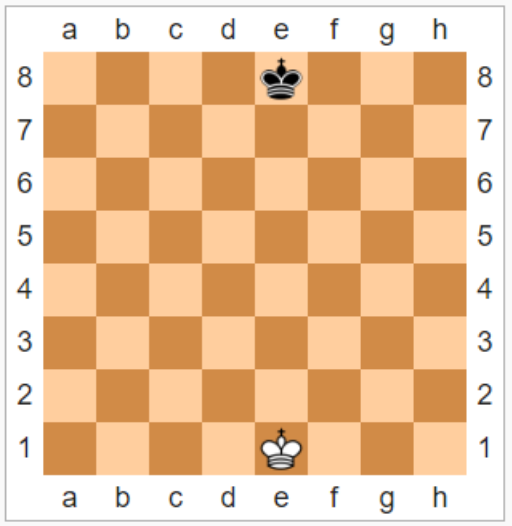
Slika 1. Početna pozicija u šahu

(https://bs.wikipedia.org/wiki/Šah)

2.2.1 Kralj

Kralj (♔, ♚) je najvažnija figura u šahu. Cilj igre je zarobiti protivnikovog kralja tako da bijeg nije moguć (mat). Ako kralja napada neka figura, tj. ako mu prijeti uzimanjem, kaže se da je u šahu i igrač mora ukloniti tu prijetnju u idućem potezu. Ako se to ne može uraditi, kaže se da je kralj matiran. Iako je kralj najvažnija figura, obično je i najslabija u igri dok se ne dođe u završnicu. Oznaka kralja u algebarskoj notaciji je K.

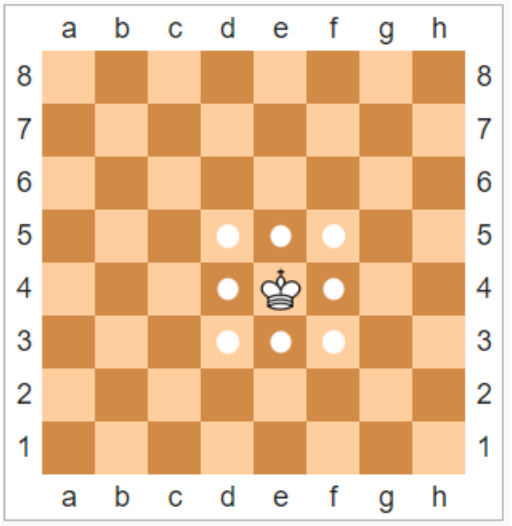
Bijeli počinje s kraljem na 1. redu, desno od dame. Crni počinje s kraljem točno nasuprot bijelog kralja. Kad su polja označena kao u algebarskoj notaciji, bijeli kralj na početku partije nalazi se na e1, a crni na e8.



Slika 2. Početna pozicija kraljeva

([**https://bs.wikipedia.org/wiki/Kralj_(šah)**](https://bs.wikipedia.org/wiki/Kralj_(šah))

Kralj se može kretati po 1 polje u svim pravcima osim ako dotično polje ne zauzima druga prijateljska figura ili ako bi se tim potezom kralj doveo u šah-poziciju. Rezultat ovoga je da suprotni kraljevi nikad ne mogu zauzeti polja odmah do onog drugog kralja (vidi članak opozicija), ali kralj može dati otkriveni šah kad se skloni s putanje lovcu, topu ili dami.

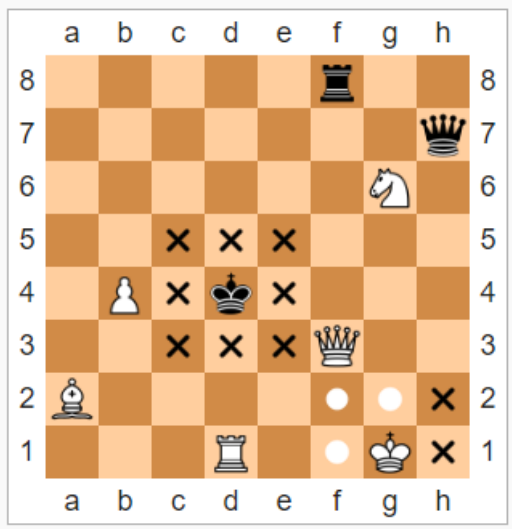


Slika 3. Mogući potezi kraljem kad nema drugih figura oko njega ili prijetnji

([**https://bs.wikipedia.org/wiki/Kralj_(šah)**](https://bs.wikipedia.org/wiki/Kralj_(šah)) **)**

Kralj također može izvesti i specijalni potez zajedno s topom, koji se zove rokada. U ovom potezu kralj se pomjera 2 polja prema jednom od njegovih topova, a top "preskače" kralja i staje na polju do njega. Rokada je dozvoljena samo u slučaju da se ni kralj ni top nisu prethodno pomicali, kada su polja između njih slobodna, kada kralj nije u šahu ili kada kralj ne mora preći preko

"napadnutih polja" ili završiti kretanje na napadnutom polju (polje pod udarom protivnikove figure). U zavisnosti od toga na koju se stranu pravi rokada, razlikuju se *velika* i *mala rokada*.



Slika 4. Mogući potezi kraljem kad mu prepreku čini granica table ili druge figure. Crni kralj ne može na polja pod udarom bijelog lovca, bijelog skakača, bijele dame ili bijelog pješaka, a bijeli kralj ne može na polja pod udarom crne dame. Bijeli je upravo odigrao Td1#, matiravši crnog kralja.
(https://bs.wikipedia.org/wiki/Kralj_(šah))

Kad igrač napadne protivnikovog kralja, kaže se da je taj kralj u šahu i napadnuti igrač mora odmah skloniti kralja iz opasnosti ili ga zaštititi drugom figurom. Postoje 3 načina da se kralj zaštiti od šaha:

- pomaknuti kralja na polje koje nije pod napadom
- postaviti neku figuru između kralja i figure koja ga napada, kako bi se prekinula linija vatre (ovo nije moguće kad je figura koja napada skakač ili kad se radi o dvostrukom šahu)
- uzeti figuru koja napada.

Ako nijedan od ova 3 načina nije moguć, kralj je doživio šah-mat i igrač čiji je to kralj izgubio je partiju.

Pat-pozicija za igrača koji je na potezu nastaje:

- kad nema legalnih poteza

- kad igračev kralj nije u šahu
- kad ne može igrati bilo kojom drugom figurom.

Ako dođe do ove pozicije, kaže se da je kralj u patu i partija se završava remijem. Igrač koji ima vrlo malo (ili nimalo) šanse da pobijedi često će pokušati varkom namamiti protivnika da greškom dovede njegovog kralja u pat-poziciju kako bi izbjegao poraz

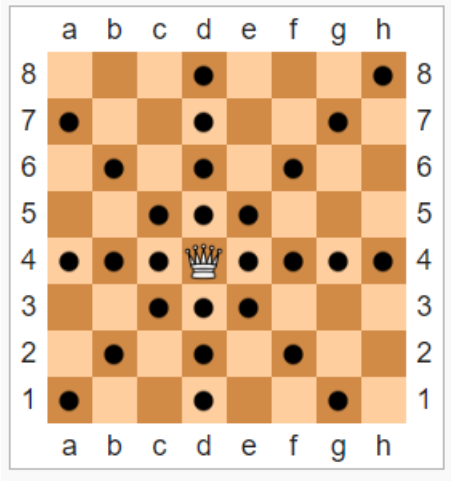
2.2.2 Dama

Dama ili Kraljica (♕,♛) je najjača figura u šahu. Zajedno sa topom, pripada tzv. teškim figurama. Praktično su potezi dame kombinacija poteza lovca i topa. Oznaka dame u algebarskoj notaciji je D (međunarodna oznaka je Q) .

Dama se u šahu može kretati u svakom pravcu, linearno ili dijagonalno, pod uvjetom da ne preskače ni jednu figuru, do prvog slobodnog polja.

Ponekad se u šahovskoj literaturi navode nepisana pravila za igranje s damom:

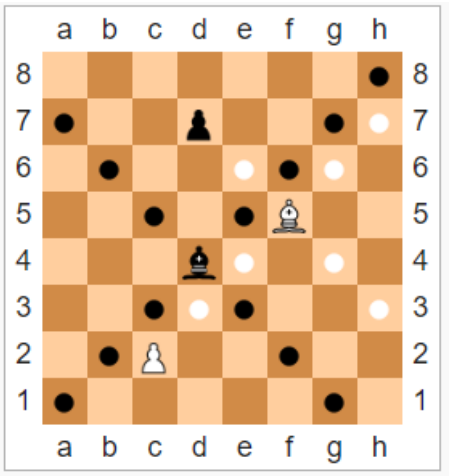
- Damom se ne bi trebalo igrati u samom otvaranju, sve dok se pješaci i lake figure ne razviju u početnoj fazi, te obično nakon rokade aktivirati damu.
- U središnjici partije, bi dama trebali biti centrirana, čime postiže najveći učinak. Treba težiti osvajanju slobodnih dijagonala i redova, čime se protivničkim figurama smanjuje polje djelovanja. Česta taktička varijanta je žrtva dame, kojom se može ostvariti velika materijalna prednost ili u konačnici mat.
- Najveća prednost dame je u završnici partije, kada na tabli ostaje malo figura, te se pokretljivost dame prikazuje u punom efektu.



Slika 5. Mogući potezi damom
(**https://bs.wikipedia.org/wiki/Šah**)

2.2.3 Lovac

Lovac (♗ , ♝) je pored skakača druga laka figura u šahu. Svaki igrač na početku partije ima dva lovca, jednog na crnom i jednog na bijelom polju. Lovac se kreće dijagonalno po šahovskoj tabli i u ovisnosti od početnog polja, kreće se po crnim ili po bijelim poljima, te se zbog položaja često nazivaju kraljev lovac i damin lovac. Lovac ne može preskakati druge figure. Oznaka lovaca u algebarskoj notaciji je L(međunarodna oznaka je B).Prednost lovca je u njegovoj pokretljivosti, jer može u jednom potezu, ukoliko nema drugih figura, preći čitavu dijagonalu. Loš položaj lovca je, ako je okružen vlastitim pješacima.

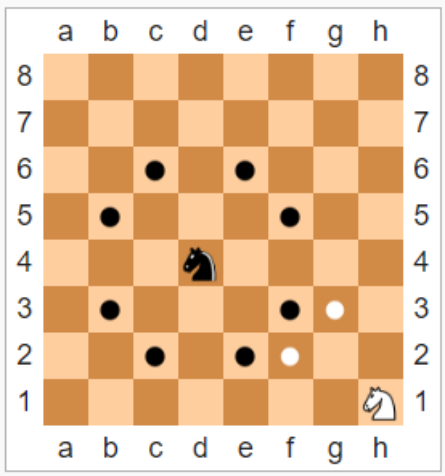


Slika 6. Mogući potezi lovcem
(**https://bs.wikipedia.org/wiki/Šah**)

2.2.4 Skakač

Skakač ili Konj (♘ , ♞) je pored lovca druga laka figura u šahu. Na početku partije svaki igrač ima dva skakača. Na šahovskoj se tabli nalaze između topa i lovca, te su pored pješaka, jedina figura sa kojom je moguće započeti šahovsko otvaranje. Skakač se kreće dva polja ravno, te jedno polje lijevo ili desno. Po definicije FIDE: Skakačem se može igrati na jedno od najbližih polja onom na kojem se nalazi, ali ne na istom redu, liniji ili dijagonali. Boja startnog i ciljnog polja, kod poteza skakačem uvijek je druge boje. Skakač može preskakati druge figure. Oznaka skakača u algebarskoj notaciji je S (međunarodna oznaka je N).

Dva skakača i kralj, protiv kralja u završnici, ne mogu natjerati protivničkog kralja u mat poziciju . Skakač nije pokretna figura, te mu je najpovoljniji položaju u sredini ploče, od kuda, ima mogućnosti premještanja na osam polja. Minimalan broj mogućih polja, samo dva, je u položaju skakača na rubnim poljima.

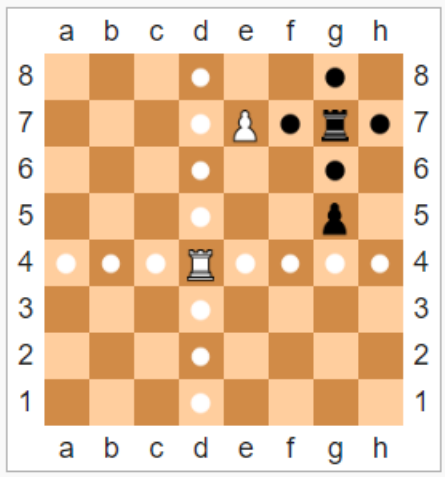


Slika 7. Mogući potezi skakačem
(https://bs.wikipedia.org/wiki/Šah)

2.2.5 Top

Top ili Kula (♖ , ♜) je druga po jačini figura u šahu. Svaki igrač na početku partije ima dva topa, jednog na lijevoj i jednog na desnoj strani ploče. Top se kreće okomito i vodoravno, neograničen broj polja pod uvjetom, da mu ni jedna figura ne stoji na liniji kretanja. Jedini izuzetak su rokade (mala 0-0 i ili velika 0-0-0, gdje top preskače preko kralja, na polje do njega. Oznaka topa u algebarskoj notaciji je T (međunarodna oznaka je R).

U šahovskoj završnici, top i kralj, protiv kralja, mogu uvijek matirati protivničkog kralja. U standardnim okolnostima, top je mnogo jači od skakača i lovca, mada u početku partije, zbog svog zatvorenog položaja igra podređenu ulogu, te igra topom na početku partije nije efektivna i vrlo se rijetko koristi. U zavisnosti od situacije, skakač i lovac, su jači od topa. Uzimanje topa, lovcem ili skakačem smatra se dobitkom materijala, dok se žrtva topa za laku figuru, može definirati kao kvalitetna žrtva, te se njome može zadobiti tempo, ili iznuditi dobru situaciju u završnici.



Slika 8. Mogući potezi topom

(**https://bs.wikipedia.org/wiki/Šah**)

2.2.6 Pješak

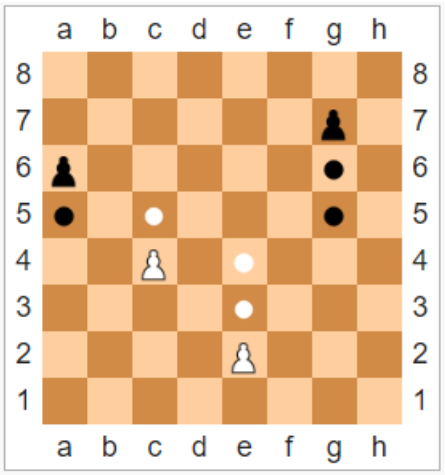
Pješak ili Pijun (♙ , ♟) je najmanje vrijedna figura u šahu. Oznaka pješaka u algebarskoj notaciji je P, ali se oznaka obično izostavlja, te se pješak obilježava slovom početnog položaja (a-h)

Pješak se u šahovskoj partiji kreće jedno polje naprijed, te je jedina šahovska figura koja se ne može pomaknuti unazad. Jedini izuzetak je, ako se pješak nalazi na početnom polju, može se premjestiti dva polja naprijed. Uzimanje pješakom vrši se dijagonalno naprijed, osim u slučaju, kada protivnički pješak preskoči branjeno polje, kada pješak uzima potezom zvanim *en passant.*

Kada pješak dosegne zadnji red, mora se promovirati u novu figuru, obično u damu, iako se u izuzetnim slučajevima može promovirati u topa, lovca ili skakača. Ne može se promovirati u drugog pješaka ili kralja.

Svaki igrač na početku partije ima na ploči osam pješaka, koji su poredani u drugom redu. U šahovskoj partiji pješaci pokazuju svoju snagu u slijedećim situacijama:

- ako su pokretni i nisu blokirani od protivničkih pješaka;
- ako su međusobno povezani u jaku liniju;
- ako su prodrli duboko u protivničko polje, te prijete promocijom na zadnjem redu.



Slika 9. Mogući potezi pješakom

(**https://bs.wikipedia.org/wiki/Šah**)

Svjetska šahovska federacija FIDE (franc, *Fédération Internationale des Échecs*) osnovana je 20. VII. 1924. u Parizu. FIDE je potaknula održavanje prve šahovske olimpijade 1927. u Londonu, od 1948. preuzima i svjetska prvenstva, a od 1950. dodjeljuje titule velemajstor i međunarodni majstor. FIDE je 1970. prihvatila sustav izračuna snage pojedinca (rating system) koji je razradio američki fizičar *Arpad Emrick Elo* (1903–92). Elo-rejting postaje glavni kriterij pri ispunjavanju normi za stjecanje titula, kategorija. Za jačinu kategorije određen je brojčani izraz – za početnike 1600 bodova, za igrače IV. kategorije 1700, III. kategorije 1800, II. kategorije 1900, I. kategorije 2000, za majstorske kandidate 2100, nacionalne majstore 2250 (ili 2200), majstore FIDE 2300, međunarodne majstore 2400 i velemajstore 2500.

3 Programska podrška

Programska podrška (eng. software) je termin koji se upotrebljava za sve programe koje koristi računalni sustav.

Programska se podrška dijeli u dvije osnovne grupe:

- sistemska programska podrška i
- aplikacijska programska podrška.

3.1 Sistemska programska podrška

Sistemsku programsku podršku proizvođač računalne opreme isporučuje korisniku zajedno sa sustavom sklopovske opreme, te je ona prilagođena njegovoj konfiguraciji (ulaznim i izlaznim jedinicama, te jedinicama masovne memorije).

Sistemsku programsku podršku sačinjavaju svi oni programski moduli, programi i programski paketi bez kojih se računalo uopće ne bi moglo aktivirati i bez kojih ono ne bi moglo izvršavati zadatke što ih korisnik pred njega postavlja.

Sistemska se programska podrška implementira već u fazi instalacije računala, a obuhvaća:

- operacijski sustav
- programe prevoditelje (jezičke procesore)
- pomoćne (servisne, uslužne) programe.

3.2 Aplikacijska programska podrška

Aplikacijska programska podrška obuhvaća programe koje korisnik upotrebljava za zadatke koje želi obaviti uz pomoć računala.

Svrha svog sklopovlja, operacijskog sustava sa svim uslužnim programima je stvaranje podloge za izvođenje aplikacijske (primjenske) programske podrške. Za neke specijalne primjene moramo sami osmisliti program, ali u većini slučajeva postoje gotovi programi ili skupine programa za uobičajene namjene kao što je npr. pisanje teksta. U ovu skupinu ubrajamo

programe za obradu teksta, programe za tablične kalkulacije, web preglednike, programe za izradu grafike, programski jezici, itd.

4 Programska podrška za Šah

U danima prije svanuća interneta relativno popularan je bio telešah (engl. telechess), tako da su između 1977. i 1990. organizirane tri olimpijade telešaha. To je šah na granici između dopisnog šaha i igranja šaha na internet serveru. Koristio se telex (metoda slanja tekstualnih poruka odvojena od telefonskog sustava, poruke su se mogle slati/primati diljem svijeta). Problem je što su partije bile vrlo spore i mogućnosti sprečavanja varanja bile su vrlo ograničene.

Računalni šah predstavlja računalnu arhitekturu koja koristi hardver i softver koji ima mogućnost igranja šaha samostalno bez kontrole čovjeka. Može se igrati samostalno (dopuštajući igračima da vježbaju poteze kad ne postoji dovoljno jak igrač protiv koga bi igrali), može se koristiti za analizu poteza, natjecanja u računalnom šahu.

Računalni šah je nedavno dostigao preokret i vrhunac s napretkom tehnologije i informatike. Pojavili su se mnogi šahovski programi, a interes je porastao za računalnim šahom nakon pobjede IBM-ovog programa *Deep Blue* nad tadašnjim svjetskim šahovskim prvakom, Gari Kasparovom. Današnji šahovski programi nepobjedivi za ljudske mogućnosti, jer ne prave grube previde, nego im je jedina razlika u pozicijskom programiranju. Trenutni prvak računalnog šaha je program *Stockfish*. Poznati programi koji funkcioniraju na sličan način su *Komodo, Houdini, Rybka, Critter* i drugi.

Danas je računalni šah dostupan za svakog korisnika. Još od sredine 1970-ih pa do danas šahovski programi bili su dostupni za kupovinu. Postoji mnogo šahovskih programa koji mogu besplatno skinuti s interneta.

Google je u prosincu 2017. godine objavio da je njihov program umjetne inteligencije *AlphaZero* koji se za razliku od dosadašnjih programa temelji na neuronskim mrežama, nakon 4 sata učenja igrajući protiv sebe, svladao aktualnog računalnog prvaka *Stockfisha* i to sa rezultatom od 28 pobjeda,72 remija i 0 poraza.

Kada govorimo o programskoj podršci za šah uglavnom se to odnosi na web stranice i mobilne aplikacije koje omogućuju korisnicima da igraju jedni protiv drugih, u raznim vremenskim formatima i oblicima, te im pomažu u daljnjem učenju i poboljšanju. Nadalje, one omogućuju korisnicima da pregledavaju svoje partije, analiziraju ih uz pomoć računala, igraju protiv računala, sudjeluju u raznim turnirima, itd.

5 Primjer programske podrške za Šah

U ovom primjeru napravljena je mobilna aplikacija za igranje šaha. Aplikacija je napravljena na način da se korisnici mogu logirati, igrati s drugim korisnicima i pregledavati svoje partije. Korisnici mogu kreirati svoje izazove koji pak drugi korisnici mogu prihvatiti. U ovom primjeru implementiran je i chat, pa korisnici mogu razgovarati tijekom partije.Za izradu aplikacije korišten je *React Native* koji je nastao iz *React* biblioteke.

5.1 React

React (poznat kao i React.js ili ReactJS) je JavaScript biblioteka za stvaranje korisničkih sučelja. React je održavan od strane Facebooka i nekolicine nezavisnih tvrtki i programera. React je stvorio *Jordan Walke*, programer u Facebooku, a prvi je put iskorišten na Facebooku 2011.,a na Instagramu 2012. godine. React je u svibnju 2015. godine postao otvorenog koda (eng. open-source).

Korisničko sučelje u Reactu je modularno i zasnovano je na komponentama. Svaka komponenta ima svoje stanje (eng. State) i to stanje možemo proslijediti u podkomponente.

React koristi i Virtualni DOM, koji služi za usporedbu sa stvarnim DOM-om i ako se ne podudaraju onda React samo izvrši promjene na onim dijelovima DOM-a koji se ne podudaraju. Ova osobina ga čini jako brzim i efikasnim. Svaki put kada komponenta promjeni svoje stanje React će osvježiti svoj Virtualni DOM, zatim ga uspoređuje sa stvarnim DOM-om.



Slika 10. Primjer koda u React-u i rezultat njegovog izvršavanja

(https://reactjs.org/)

5.2 *React Native*

React je u veljači 2015. godine na Facebookovoj React.js konferenciji objavio React Native, framework koji dopušta razvoj mobilnih aplikacija za iOS i Andorid.

React i React Native su u sintaksi jako slični. Jedna od razlika je u pisanju JSX-a. U React-u možemo koristit tagove kao što su *div* i *p*, dok u React Nativu koristimo *View* i *Text*. Razlog tome je to što React Native komponente pretvara u izvorne (eng. native) komponente. Na primjer React Native *ScrollView* komponenta koja nam omogućuje skrolanje po ekranu, na iOS uređajima koristi komponentu *UIScrollView*, a na android uređajima izvorni *ScrollView*. Ovo nam omogućuje izvornu brzinu uz React-ov dizajn.

React Native nam dopušta lako integriranje komponenata napisanih u programskim jezicima kao što su *Objective-C*, *Swift* ili *Java*. Ovaj pristup se često koristi ako želimo optimizirati neke dijelove aplikacije. Na primjer, Facebook-ova aplikacija funkcionira na ovaj način, tj. dio aplikacije je napisan u React Native-u a dio u izvornom kodu.

```
import React, { Component } from 'react';
import { Text, View } from 'react-native';

class WhyReactNativeIsSoGreat extends Component {
  render() {
    return (
      <View>
        <Text>
          If you like React on the web, you'll like React Native.
        </Text>
        <Text>
          You just use native components like 'View' and 'Text',
          instead of web components like 'div' and 'span'.
        </Text>
      </View>
    );
  }
}
```

Slika 11. Primjer React Native koda

(https://reactjs.org/)

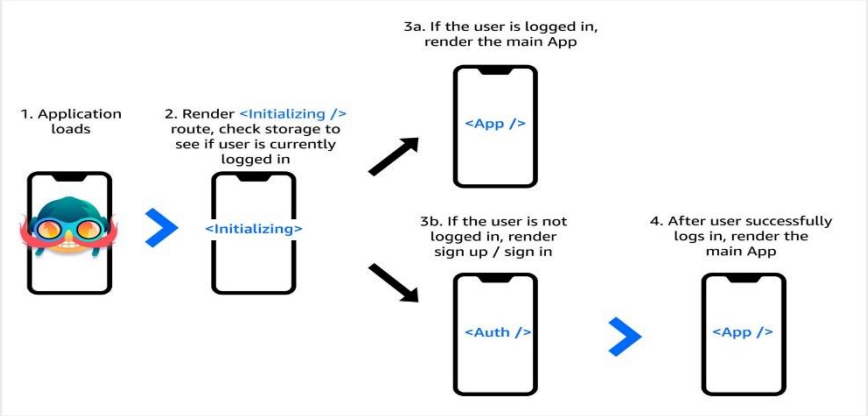
5.3 Šah i React Native

U ovom primjeru korišten je Node.js i Express za server, također je korišten i Firebase za logiranje korisnika, spremanje podataka o korisnicima i spremanje partija. Također korištene su i dodatne biblioteke:

- socket.io(https://github.com/socketio/socket.io)
- react-navigation- 2.0 (https://reactnavigation.org/blog/2018/05/07/react-navigation-2.0.html)
- react-native slider(https://github.com/jeanregisser/react-native-slider)
- react-native-vector-icons(https://github.com/oblador/react-native-vector-icons)
- react-native-background-timer (https://github.com/ocetnik/react-native-background-timer)
- firebase

5.3.1 Kreiranje korisnika

Ovaj projekt zamišljen je na način da se prilikom starta aplikacije prvo pojavi *Loading.js* komponenta u kojoj se provjerava da li je korisnik logiran. Ako je korisnik logiran aplikacija nas vodi na *Main.js* komponentu. U slučaju da korisnik nije logiran onda nas aplikacija vodi na *Login.js* ili *Singup.js* komponentu. Nakon što se prijavimo ili registriramo aplikacija nas automatski vodi na *Main.js* komponentu.



Slika 12. Shema aplikacije

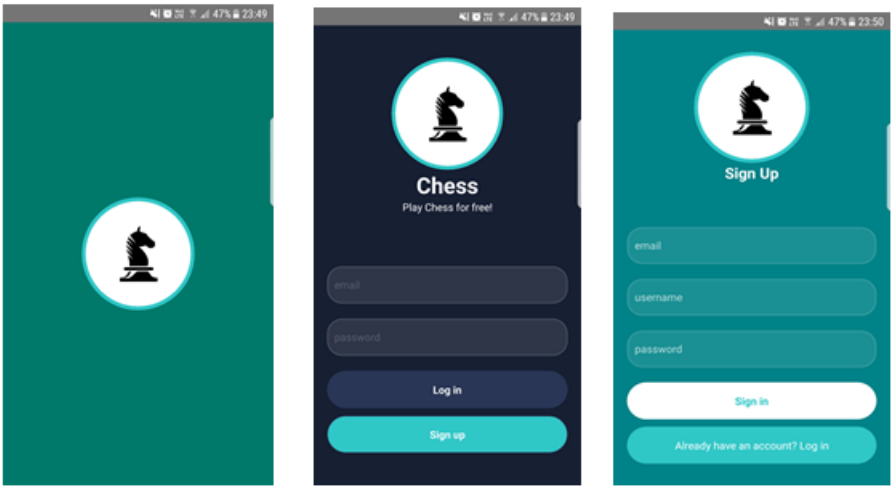
(https://medium.com/react-native-training/react-native-navigation-v2-by-wix-getting-started-7d647e944132)

U React-u postoje metode koje se pozovu nakon nekog događaja. Na primjer metoda *componentDidMount* se automatski poziva kada se komponenta prikaže (montira) na ekran. Komponenta *Loading.js* se sastoji od jednostavnog loga. U toj komponenti iskorištena je metoda *componentDidMount* tako da kada se ta metoda pozove provjerimo da li je korisnik logiran.

Komponenta *Login.js* se sastoji od loga, dva tekst inputa (komponenta *TextInput*) i dva *TouchableHighlight*-a. Komponenta *TouchableHighlight* osigurava da se pozove *onPress* događaj kad pritisnemo tu komponentu. Komponenta *TouchableHighlight* se sastoji od podkomponente *View* koja se sastoji od podkomponente *Text* u kojoj piše „Log in“ ili „Sign up“ ovisno o komponenti. Komponenta također sadrži i svoje stanje (eng. State). State ili stanje

komponente je JavaScript objekt koji u ovom slučaju sadrži dva svojstva: *email* i *password*. Kada korisnik unese tekst u komponentu TextInput vrijednost unosa se sprema u odgovarajuće svojstvo *state* objekta. Nakon što pritisnemo „Log in“, pošalje se zahtjev Firebase-u da logira korisnika. U slučaju da neko polje nije dobro uneseno ili korisnik ne postoji onda se korisniku ispiše odgovarajuća poruka. U slučaju da korisnik pritisne „Sign up“ ,onda ga aplikacija vodi na *Signup.js* komponentu. Cijelu komponentu okružuje još jedna komponenta a to je *KeyboardAvoidingView* koja nam osigurava da tipkovnica ne prekriva elemente dok pišemo na način da sve elemente pomakne vertikalno.

Komponenta *Signup.js* se sastoji od loga, tri tekst inputa (komponenta *TextInput*) i kao i komponenta *Login.js* dva *TouchableHighlight*-a. State komponente u ovom slučaju sadrži tri svojstva: *email*, *username* i *password*. Nakon što pritisnemo „Sign in“, pošalje se zahtjev Firebase-u da kreira korisnika. U slučaju da neko polje nije dobro uneseno ili da postoji korisnik s tim e-mailom onda se korisniku ispiše odgovarajuća poruka. U slučaju da korisnik pritisne „Already have an account?Log in“ ,onda ga aplikacija vodi na *Login.js* komponentu

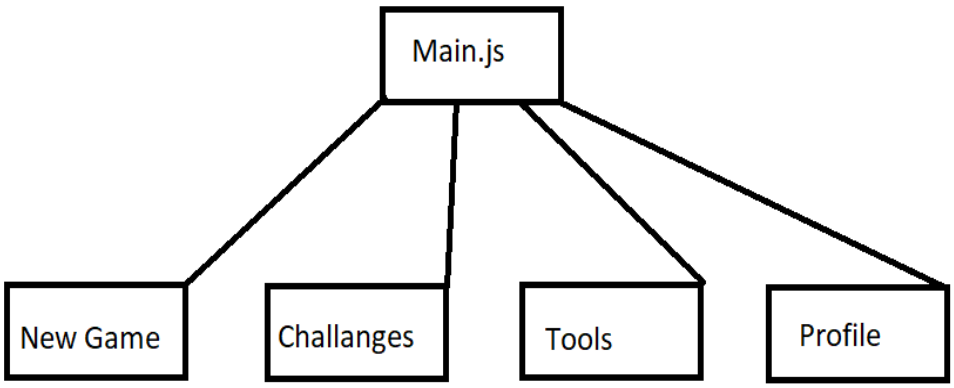


Slika 13. Loding.js,Login.js i SignUp.js

5.3.2 Glavni dio aplikacije

Komponenta *Main.js* se sa sastoji od komponente *bottomTabNavigaor* koju možemo koristit nakon što smo instalirali paket *react-navigation*, Ova vrsta navigacije nam osigurava

jednostavnu sučelje kao kod aplikacije Instagram ili YouTube android aplikacije. Komponenta *Main.js* se sastoji od 4 dijela, svaki dio se sastoji od *stackNavigaor* komponente.



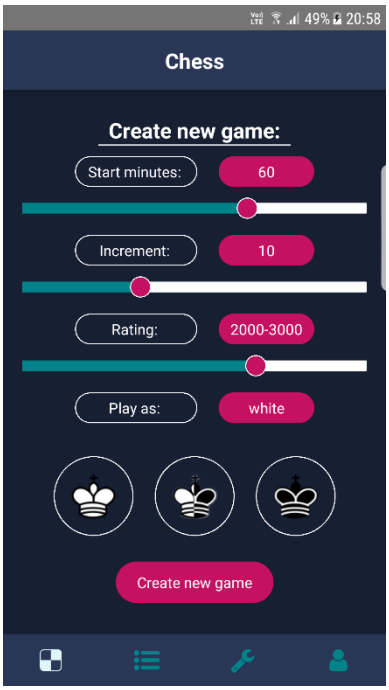
Slika 14 .Struktura komponente Main.js

Prilikom korištenja *bottomTabNavigaor* komponente potrebno je navesti koje sve komponente sadrži. Možemo još i navesti druge informacije, na primjer boju pozadine, boju aktivne kartice, boju neaktivne kartice, tekst, itd. U ovom primjeru *bottomTabNavigaor* sadrži četiri komponente od koji je svaka *stackNavigaor*, a svaki od njih se sastoji od barem dvije podkomponente. Možemo birati koju komponentu prvu da prikažemo korisniku, a u ovom slučaju to je *New Game*.

Komponenta *New Game* je *stackNavigaor* i sastoji se od dvije komponente. Prva komponenta je *createNewGame.js* koja se prva prikaže korisniku nakon prijave, a druga komponenta je PlayingScreen.js koja nam služi za igranje.

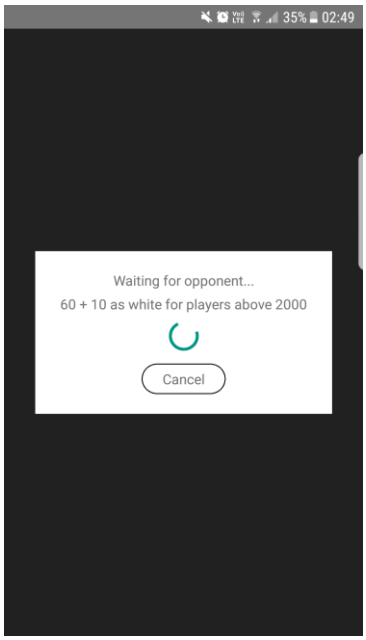
Komponenta *createNewGame.js* se sastoji od tri *slider-a* i četiri *TouchableHighlight-a,* u svom state-u sadrži četiri svojstva: početne minute, dodatak nakon svakog poteza u sekundama, rating i stranu za igranje. React Native dolazi sa ugrađenim sliderima ali u ovom primjeru je instaliran paket *react-native slider* koji nam za razliku od ugrađenih, nudi mogućnosti uređivanja. Tri *TouchableHighlight-a* su iskorištena za odabir strane. Svaki od njih sadrži komponentu *View* koja sadrži komponentu *Image* koja prikazuje sliku od šahovske figure kralja. Svaki put kada pritisnemo jednog od njih promjeni se svojstvo strane za igranje state objekta i to na način da ako smo pritisnuli *TouchableHighlight* sa bijelim kraljem onda

strana za igranje postaje bijela, ako smo pritisnuli *TouchableHighlight* sa crnim kraljem onda strana za igranje postaje crna, a ako smo pritisnuli *TouchableHighlight* sa crno-bijelim kraljem onda strana za igranje postaje slučajna.



Slika 15. createNewGame.js komponenta koja nam služi za kreiranje novog izazova

Posljednji *TouchableHighlight* je iskorišten za kreiranje nove igre. Kada ga pritisnemo aplikacija nas odvede na drugu komponentu *NewGame stackNavigator-a* koja se zove *PlayingScreen.js.* Zatim pošaljemo serveru obavijest da želimo kreirati novi izazov koristeći instalirani paket *socekt.io*, zatim čekamo da netko prihvati naš izazov.

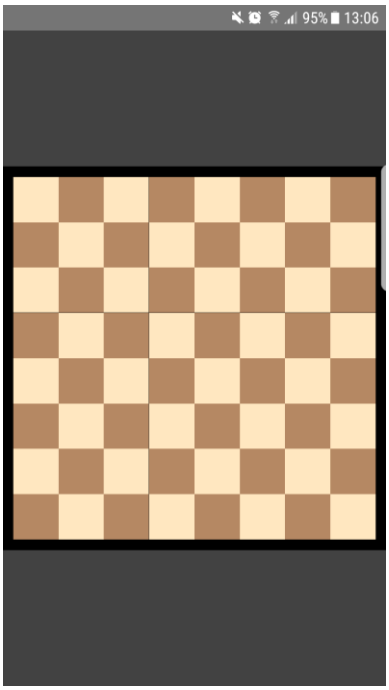


Slika 16. PlayingScreen.js dok čekamo da netko prihvati naš izazov

U slučaju pritiska na *Cancel* serveru se šalje obavijest da smo odustali od našeg izazova i on ga zatim briše te ga nitko više ne može prihvatit. Zatim nas aplikacija vraća na *createNewGame.js* komponentu. U slučaju da je netko prihvatio naš izazov potrebno je prikazati ploču i započeti igru.

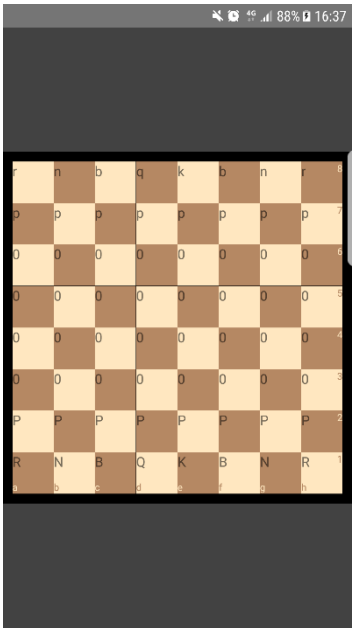
5.3.3 Izrada sučelja za igranje s protivnikom

Šahovska ploča i šahovske figure su osnovni dio svake šahovske aplikacije. U ovom primjeru korištenu su dvije komponente za prikazivanje ploče. Prva komponenta je *Chessboard.js* koja će sadržavati sva polja. Kako šahovska ploča ima 64 polja, iskoristit ćemo modularnost React Native-a, pa ćemo napravit samo jednu komponentu koju ćemo nazvati *Tile.js* koja će predstavljati jedno od 64 polja.. U React-u komponente mogu prosljeđivati podkomponentama svojstva ili *props.* U komponenti *Chessboard.js* stvorit ćemo 64 *Tile* komponente i svakoj od njih ćemo poslati neka svojstva. Za početak proslijedimo samo jedno svojstvo i to svojstvo boje. Rezultat je prikazana na slici 17.



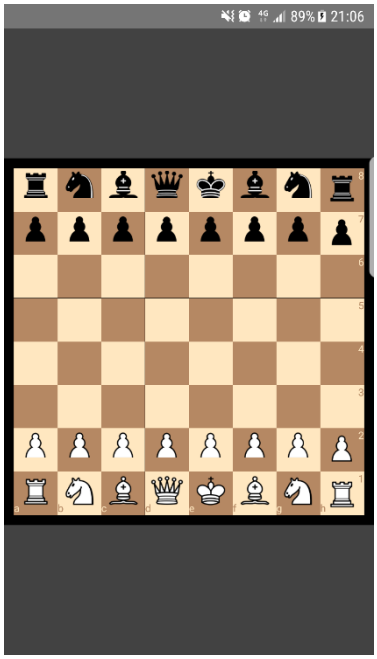
Slika 17. Chessboard.js sa 64 Tile.js komponente koje ne prikazuju ništa osim boje

Sljedeći korak je prikaz svih figura na ploči. Komponenta *Chessborad.js* sadrži jednostavan dvodimenzionalni niz koji sadrži informacije o svakom polju, tj. sadrži boju svakog polja, figuru koja je na tom polju i koordinatu tog polja. Sada je potrebno poslati svakoj *Tile* komponenti dva nova svojstva jer je boja polja već poslana. Rezultat je vidljiv na slici 18.



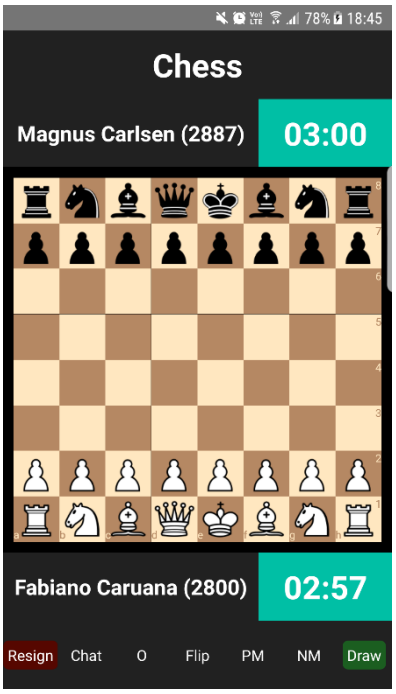
Slika 18. *Chessboard.js* komponenta sa podacima o boji, koordinati i figuri za svako polje.
Bijele figure su označene sa velikim slovima, crne s malim, a 0 predstavljaju polja bez figura.
U prvom redu i zadnjem stupcu su vidljive koordinate za svaki redak i stupac.

Sljedeći korak je učitavanje slika svih figura i njihove zamjene sa slovima. Slike svih figura su učitane u *Tile* komponenti, a prikazat će se samo jedna i to ona koja je proslijeđena iz komponente *Chessboard*.js putem svojstava. Na primjer, ako je poslano svojstvo „K“, onda će se prikazati slika bijelog kralja. Rezultat je vidljiv na slici 19.



Slika 19. *Chessboad.js* komponenta sa učitanim figurama

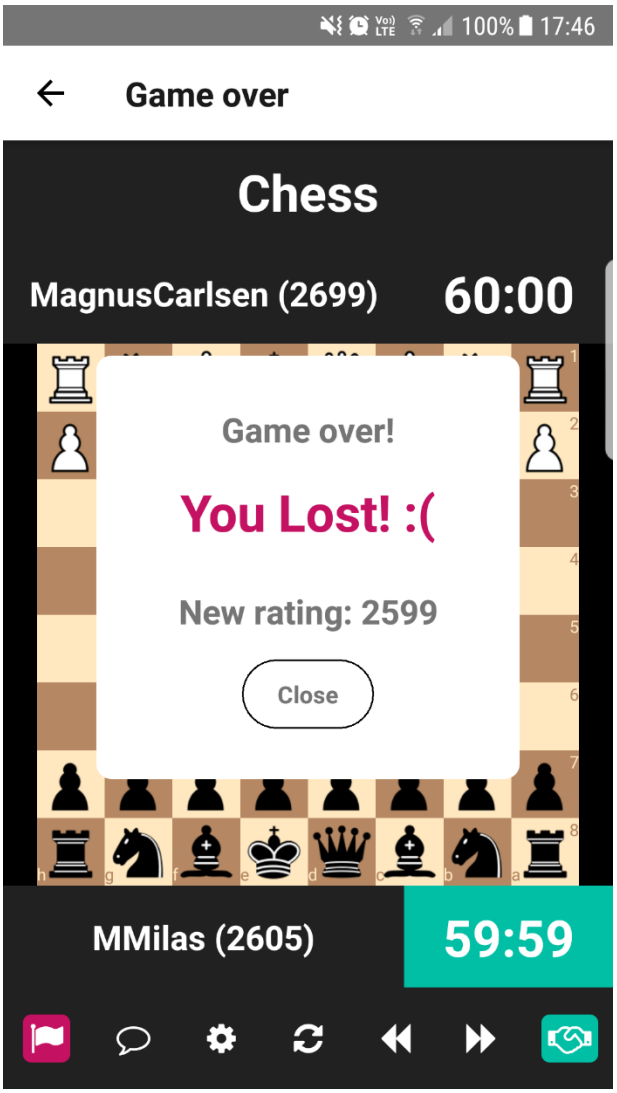
Nakon prikazivanja figura, potrebno je prikazati imena igrača i vrijeme svakog igrača. Također je bilo potrebno centrirati i bolje urediti slike svih figura. Nakon toga u Chessborad.js komponentu dodana su još i šest *TouchableHighlight* komponenti koje se nalaze u zadnjem redu. Rezultat je vidljiv na slici 20.



Slika 20. Chessboard.js nakon dodavanja imena, vremena i ostalih dijelova

Prva komponenta *TouchableHighlight-a* služi za predavanje partije. Kada pritisnemo „Resign“ otvara nam se prozorčić za potvrdu, i ako potvrdimo gubimo partiju. Kraj njega se nalazi *TouchableHighlight* koji otvara prozor za razgovor u kojem se korisnici mogu dopisivati. Zatim slijede komponenta *opcije* u kojem igrač može birati promociju pješaka. Nakon toga slijedi komponenta za okretanje ploče koja samo okrene ploču i ništa drugo. Ta akcija neće učiniti da igrači zamjene strane. Zatim slijede dvije komponente *TouchableHighlight-a* koje služe za pregledavanje prošlih poteza. Zadnja komponenta služi za ponudu remija drogom igraču. Kada pritisnemo tu komponentu šalje se obavijest drugom igraču koji onda može a i ne mora prihvatit remi. U slučaju da prihvati, igra je gotova i korisnicima je nudi opcija za vraćanje. Tijekom igranja ne možemo mijenjati kartice.

U slučaju da jedan od igrača napusti partiju bez predaje, onda drugi igrač automatski pobjeđuje. Nakon završetka partije igračima se računaju bodovi, koji se nakon toga prikažu korisniku na ekranu (slika 21), a zatim spremanju na *Firebase* skupa se cijelom partijom koja se kasnije može pregledavati.

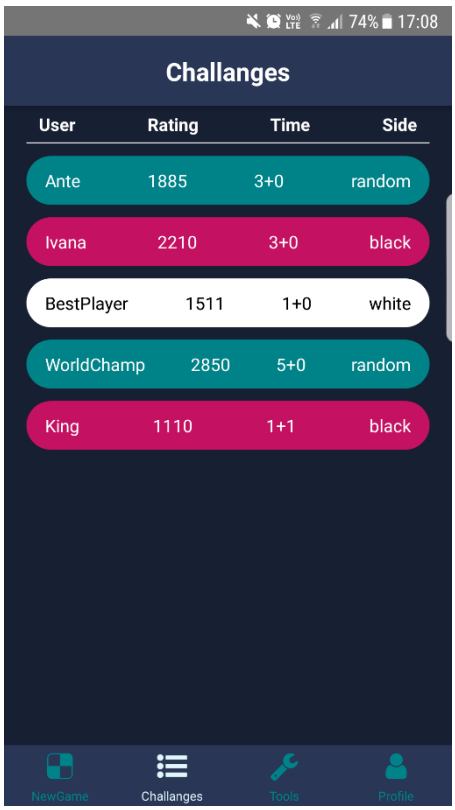


Slika 21. Chessboard.js sa ikonama i obavijesti o rezultatu partije

5.3.4 Pregled otvorenih izazova

Druga komponenta *bottomTabNavigator-a* je novi *StackNavigator* koji se sastoji od dvije komponente: *Challanges.js* i *PlayingScreen.js*. U komponenti *Challanges.js* možemo vidjeti sve otvorene izazove, i onda pritiskom na jednom od njih šaljemo obavijest serveru da smo prihvatili taj izazov i onda nas aplikacija vodi na komponentu *PlayingScreen.js*, te započinjemo igru s tim protivnikom.

U komponenti *Challanges.js* nalazi se komponenta *FlatList* koja je uključena u React Native, te nije potrebno instalirati nikakve dodatne pakete za njeno korištenje. Za njeno ispravno korištenje potrebno je navesti par svojstava od kojih je najvažnije svojstvo izvora podataka. U ovom slučaju izvor je niz *openChallanges* koji je svojstvo state objekta komponente Challlanges.js. U početku je niz prazan, pa u metodi *componentDidMount* šaljemo zahtjev serveru da nam pošalje popis svih izazova. Nakon što dobijemo popis, spremimo ga u *openChallanges* i zatim se lista izazova prikaže na ekranu kao na slici 22.



Slika 22. Popis otvorenih izazova

Svi izazovi u kojima korisnici koji su ih kreirali žele igrati sa bijelim figurama imaju za pozadinu bijelu boju, u slučaju da korisnici žele igrati s crnim onda je boja pozadine crvena, a ako su postavili slučaju boju onda je boja pozadine zelena. Boja slova je prilagođena boji pozadine kod svih izazova.

5.3.5 Implementacija različitih alata

Sljedeća kartica bottomTabNavigator-a je novi *StackNavigator* koji se u ovom slučaju sastoji od pet novih komponenti. Prva komponenta je Tools.js koja nam služi za odabir jednog od četiri dodatna načina igre. Svaki od tih načina je jedna komponenta.

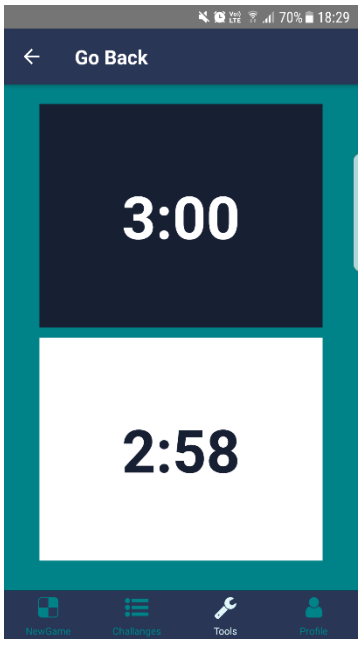
U *Tools.js* možemo birati da li želimo igrati protiv prijatelja na uređaju, protiv računala, analizirat partije pomoću računala ili koristit sat ako želimo igrati na pravoj ploči ali nemamo pravi šahovski sat.

Izgled *Tools.js* komponente je prikazan na slici 23.



Slika 23. Izgled *Tools.js* komponente

U slučaju da smo odabrali „Offline clock“ otvara nam se komponenta *Clock.js*. U toj komponenti se nalaze dva *slider*-a pomoću koji biramo startne minute i dodatak nakon svakog poteza. Nakon sto smo podesili vremenski format otvara se sat kao na slici 24.



Slika 24. Izgled *Clock.js* komponente

Kada odigramo potez na stvarnoj ploči, onda samo pritisnemo svoje vrijeme koje onda stane i krene teći protivniku sve dok on ne odigra potez pa pritisne svoje vrijeme.

Ako smo umjesto sata odabrali opciju da igramo protiv prijatelja izvan mreže u komponenti *Tools.js* onda se otvori komponenta *OfflineChessboard.js* (slika 25) koja za razliku od *Chessboard.js* komponente ne podržava razgovor, te ne sadrži opcije za predaju i ponudu remija



Slika 25. Izgled *OfflineChessboard.js*

Komponenta *OfflineChessboard.js* sadrži tri komponente od kojih dvije služe za pregledavanje prošlih poteza, dok treća komponenta služi da izbor promocije pješaka. U ovom načinu igrači igraju bez vremena.

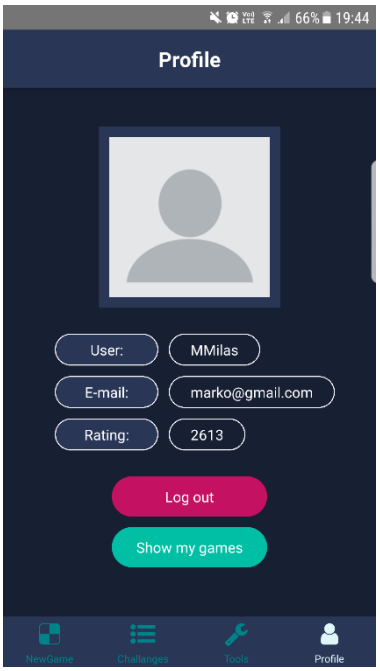
Posljednja komponenta *bottomStackNavigator-a* je *StackNavigator* koji se sastoji od dvije komponente: *Profile.js* i *ReviewChessBoard.js*.

5.3.6 Korisnikov profil

Komponenta *Profile.js* služi za prikazivanje podataka o korisnikovom profilu. Podaci koje možemo vidjeti su: korisničko ime, e-mail i bodovi. Također, postoji *Image* komponenta za korisnikovu sliku i dva *TouchableHighlight-a* od kojih jedan služi za odjavu a drugi za prikaz vlastitih partija.

Rezultat je vidljiv na slici 26.

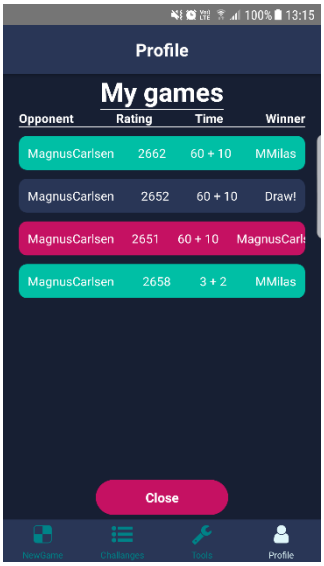
Pritiskom na „Log out“ korisnik se odjavi, zatim ga aplikacija vodi na *Login.js* komponentu.



Slika 26. *Profile.js*

Pritiskom na „Show my games“ otvara se popis partija koje je korisnik igrao s drugim igračima. Korištena je komponenta *FlatList*, a u metodi *componentDidMount* šalje se zahtjev na Firebase za prikaz korisnikovih partija, a odgovor se koristi kao izvor *FlatList-e*.

Svaki element liste je *TouchableHighlight* koji u sebi sadrži osnovne podatke o svakoj partiji. Pozadina elementa je zelene boje ako je korisnik pobijedio, crvene ako je izgubio a u slučaju remija plave boje (slika 27). Pritiskom na neki od elementa otvara se nova komponenta *ReviwChessBoard.js* kao na slici 28.



Slika 27. Popis partija korisnika

Komponenta *ReviwChessBoard.js* se sastoji od ploče, korisničkih imena, vremenskog formata i tri *TouchableHighlight* komponente od kojih dvije služe za pregledavanje prošlih poteza, a treća služi za rotiranje ploče (slika 28).



Slika 28. *ReviewChessBoard.js* koja služi za pregled završene partije

U slučaju da je rotacija ploče uključena, onda pozadina *TouchableHighlight* komponente za rotaciju promijeni boju u plavu. Povratak nas vodi na popis korisnikovih partija.

Na kraju svake partije, pobjednik postavlja završenu partiju na Firebase, a u slučaju da je partija završila remijem onda igrač sa bijelim figurama postavlja partiju. Ovo je jednostavan način pomoću kojeg smo izbjegli duplikate.

Svaki igrač nakon završene partiju računa svoje nove bodove, zatim ih postavlja na Firebase. U slučaju da jedan igrač napusti partiju, pobjednik je njegov protivnik i u tom slučaju igrač koji je pobijedio računa nove bodove od oba igrača i zatim ih postavlja na Firebase.

S time smo završili primjer programske podrške za aplikaciju Šah.

6 Zaključak

Šah je igra za dva igrača koja se igra na ploči od 64 crno-bijela polja sa 32 figure, gdje je cilj svakom igraču zadati *mat* protivniku, tj. zarobiti protivnikovog kralja, a pri tome sačuvati vlastitog.

Rana verzija je nastala oko 600. godine u Indiji, zatim se kroz idućih par stoljeća širila po svijetu. Europljani su prvi standardizirali pravila i od nje napravili igru kakvu danas poznajemo.

S razvojem računala i interneta razvila se i programska podrška za šah. Pojavili su se mnogi programi, mnogo snažniji od najjačih ljudskih igrača. Razvile su se mnoge web stranice na kojima igrači mogu igrati međusobno.

Pojava pametnih uređaja dodatno je pomogla širenju šaha, jer sada igrači u svome džepu imaju pristup najsnažnijim programima, najvećim bazama podatka, lekcijama od raznih velemajstora, te samo u par koraka mogu igrat uživo sa protivnikom iz bilo koje države.

Pojavili su se mnogi turniri, u kojima mogu sudjelovati svi sa pristupom internetu, a u kojima pobjednici dobivaju novčane nagrade. Možda najveći problem sadašnje šahovske programske podrške je u tome što igrači mogu varati. Na primjer, na turnirima u kojima se dodjeljuje novčana nagrada korisnik na svome laptopu može igrati protiv protivnika, a na svom mobitelu upali šahovski program pa u njega unosi protivnikove poteze. Sadašnja računala i mobiteli su toliko snažni da samo u par sekundi pronađu najbolje poteze koje onda igrač može iskoristit protiv svog protivnika, i kao rezultat toga osvoji turnir bez problema.

Postoje mnoge metode koje pomažu uhvatit takve igrače, ali nikad ne mogu biti u potpunosti točne jer svaki igrač ponekad može odigrati odličnu partiju.

Ovaj rad sadrži primjer aplikacijske programske podrške za igru šah. Primjer je implementiran na način da korisnici mogu igrati s bilo kim u svijetu, birati kakvu igru žele, igrati bez interneta, pregledavati svoje partije i koristiti alate kao što su šahovski sat.

7 Popis literature

Knjige:

1. Banks Alex, Porcello Eve (2017) .*Learning React: Functional Web Development with React and Redux.* O'Reilly Media.

2. Boduch Adam (2017).*React and React Native*. Packt Publishing

3. Haverbeke Marijn (2011). *Eloquent JavaScript: A Modern Introduction to Programming*. No Starch Press.

4. Horton Adam, Vice Ryan (2016). *Mastering React*. Packt Publishing

5. Zakas Nicholas C. (2005). *Professional JavaScript for Web Developers.* John Wiley & Sons

Mrežne stranice:

1. Annonymus, Šah
   URL: http://www.enciklopedija.hr/Natuknica.aspx?ID=59291

2. Annonymus, Šah
   URL: https://hr.wikipedia.org/wiki/Šah

3. Annonymus, Šah
   URL: https://bs.wikipedia.org/wiki/Šah

4. Annonymus, Kralj (šah)
   URL: https://bs.wikipedia.org/wiki/Kralj_(šah)

5. Facebook Inc., React
   URL: https://reactjs.org/

6. Facebook Inc., React Native
   URL: https://facebook.github.io/react-native/

7. Annonymus, Programska podrška
   URL: https://hr.wikipedia.org/wiki/Programska_podrška

8. Annonymus, Programska podrška
   URL: http://mapmf.pmfst.unist.hr/~lada/oi/pog6-2.pdf

9. Annonymus, Programska podrška
   URL: http://www.fpz.unizg.hr/hgold/es/de/programska%20p.htm