Atomac

-Šah na dve table za četiri igrača-

Arhitekturni Projekat

Dimitrije Cvetković, 15057
Nikola Lugić, 15193
Nemanja Malinović, 15199

Elektronski Fakultet Niš, Novembar 2017.

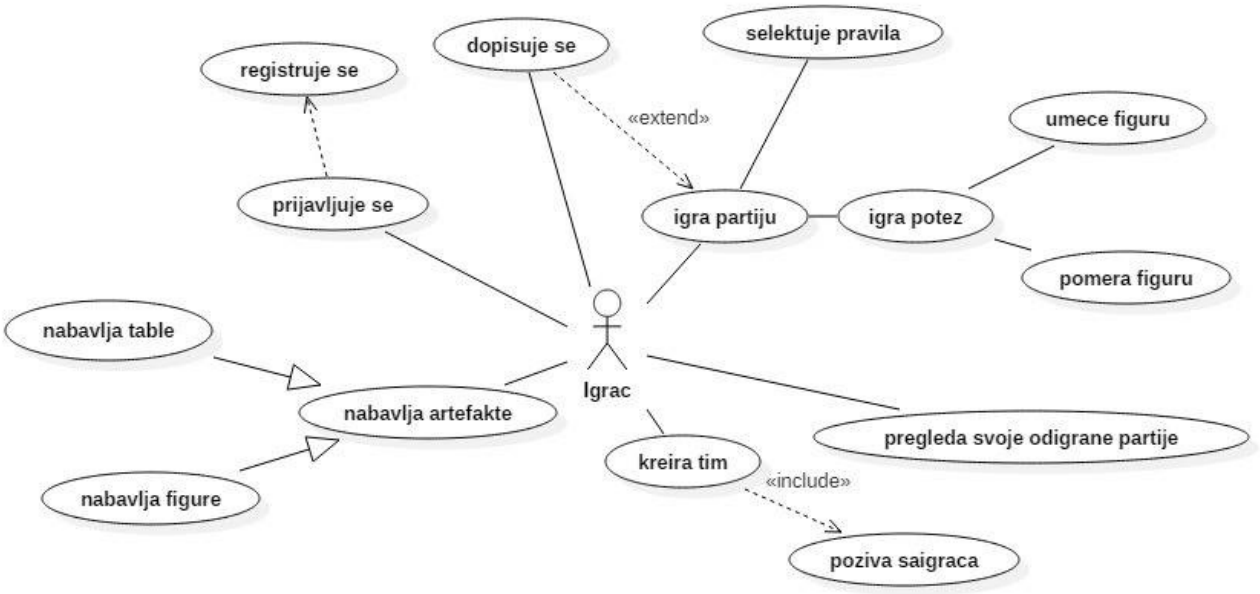
Opis sistema

Atomac je šah koji se igra na dve table, od strane četiri igrača (dva tima sa po dva igrača). Igraju se dve partije istovremeno i one su nezavisne, izuzev razmene osvojenih figura - igrač koji osvoji protivnikovu figuru daje figuru svom saigraču na susednoj tabli, tako da je ovaj može rasporediti na tablu (uz određena ograničenja) umesto odigravanja uobičajenog poteza. Pobeda na bilo kojoj od dve table označava pobedu tima. Pobedu tim moze ostvariti tako sto jedan član tima matira protivnika na svojoj tabli, ili tako što nekom od dvojice igrača iz protivničkog tima istekne vreme za igru. Tokom partije je dozvoljena konumikacija između članova istog tima. Iz navedenog je jasno da je glavni i jedini cilj sistema zabava za 4 osobe, pri čemu je potrebno poznavanje pravila šaha.

Zahtevi sistema

Funkcionalni zahtevi

Funkcionalni zahtevi sistema prikazani su dijagramom slučajeva korišćenja na slici ispod. Glavni i jedini akter sistema je igrač. Igrač mora biti registrovan da bi mogao da igra igru i koristi ostale funkcije sistema. Nakon prijave na sistem, igrač može formirati tim pozivanjem saigrača, dopisivati se sa ostalim aktivnim igračima, kupovati druge dizajne šahovske table i figura, gledati snimke partija koje je odigrao i igrati igru. Igrači se dogovaraju o pravilima igre pre njenog početka, a dok igra traje moguće je i dopisivanje sa drugim igračima.



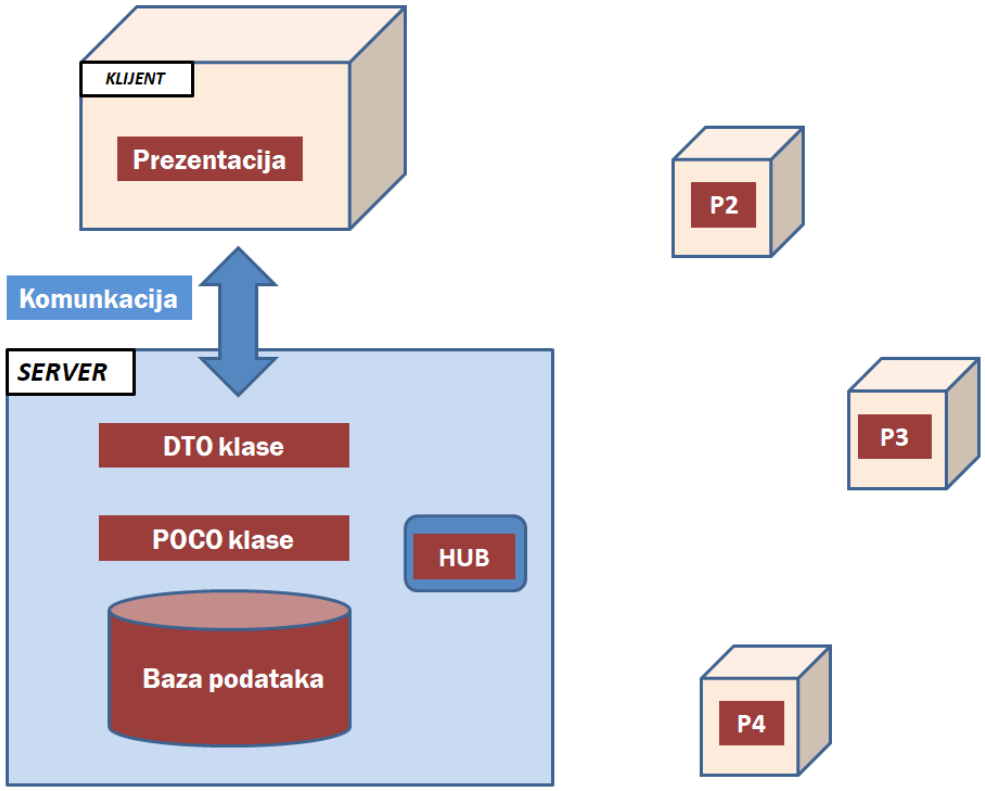
Nefunkcionalni zahtevi

S obzirom na real-time prirodu sistema, neophodna je pouzdana veza između igrača u toku trajanja partije. Ovo podrazumeva pouzdanu vezu sa serverom, njegovu dostupnost i dovoljnu brzinu obrade zahteva i događaja u igri. Odziv sistema mora biti manji od 3 sekunde, a svi zahtevi moraju biti izvršeni za manje od 10 sekundi, kako ne bi došlo do remećenja igre. Prikladan dizajn i upotrebljivost korisničkog interfejsa su neophodni, imajući u vidu njegovu složenost u toku partije - potrebno je prikazati tablu na kojoj igra igrač, drugu tablu na kojoj igra njegov saigrač, prozor sa porukama u toku igre i status igre, odnosno dodatne podatke.

Prikaz arhitekture sistema

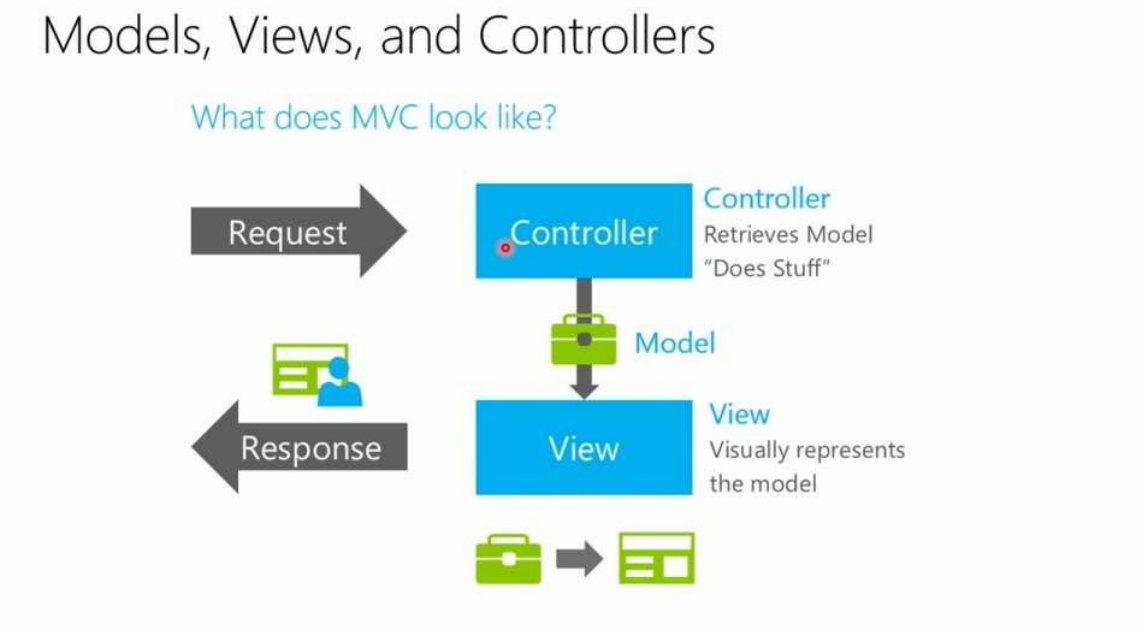
Uprošćen prikaz arhitekture sistema dat je na slici ispod. Osnovnu arhitekturu sistema čini Model-View-Controller arhitekturni obrazac, koga implementira ASP.NET MVC 5 framework koji je odabran za izradu aplikacije. Takođe, za potrebe real-time komunikacije u toku igre, koristi se i ASP.NET SignalR server (hub), koji implementira Publisher-Subscriber arhitekturni obrazac. Za sloj perzistencije koristi se Entity Framework ORM nad MSSQL bazom podataka kreiranom Code First pristupom, dok se za komunikaciju između servera i klijenta koriste DTO objekti (ViewModel klase u ASP.NET framework-u). Na klijentskoj strani se za prikaz stranica koristi Razor View Engine, kao podrazumevano rešenje za View u sklopu ASP.NET MVC aplikacije. Za prikaz šahovskih elemenata i logiku same igre koriste se chessboardJS i chess.js JavaScript biblioteke uz određene modifikacije.

Komunikacija se obavlja preko dva kanala: između servera i klijenta, korišćenjem ASP.NET kontrolera i između SignalR huba i njegovih klijenata (igrača u toku igre), korišćenjem asinhrone komunikacije (WebSocket/pozivi slični RPC-u).



ASP.NET MVC 5

Za izradu aplikacije odabran je ASP.NET MVC 5 framework za izradu Web aplikacija. Razlozi za ovaj izbor leže u odličnoj podršci i proširljivosti ovog framework-a, kako u pogledu perzistencije i komunikacije sa bazom, tako i u pogledu komunikacije kontrolera sa View sistemom i odabira samog View sistema - moguće je u budućnosti Razor View Engine zameniti nekim od front end framework-a, od kojih je Angular najpopularniji izbor. Takođe, zahvaljujući ASP.NET Identity sistemu, registracija i prijavljivanje korisnika su već obezbeđeni. što ostavlja više vremena za projektovanje i implementaciju same srži aplikacije. RTC podrška koju donosi SignalR rešava poslednji veliki problem arhitekturnog projektovanja - potrebu za real-time komunikacijom u toku igre i prikazivanje promena stanja igre kod svih igrača. Prikaz osnovne arhitekture ASP.NET MVC framework-a dat je na slici ispod.



SignalR

SignalR je framework koji, zajedno sa ASP.NET-om omogućava izgradnju višekorisničkih, interaktivnih i real-time web aplikacija. SignalR konekcija se kreira tehnologijom koja najbolje pristaje klijentu i serveru, i potpuno je transparentna u odnosu na programera (prva slika ispod). Sam SignalR, pored Publisher-Subscriber obrasca koji čini, implementira i Layered obrazac u svojoj arhitekturi, što je prikazano na drugoj slici ispod. Postoje dva API-ja za rad sa SignalR-om - API za perzistentne konekcije, API nižeg nivoa za komunikaciju sličnu komunikaciji preko soketa i API za rad sa tzv. Hub-ovima, koji implementira Publisher-Subscriber obrazac i radi po principu sličnom RPC-u. U aplikaciji se koristi drugi model, jer je potrebna upravo jedan-na-više komunikacija između servera (hub-a) i klijenata (igrača).