Arhitektura i projektovanje softvera

DrawOut

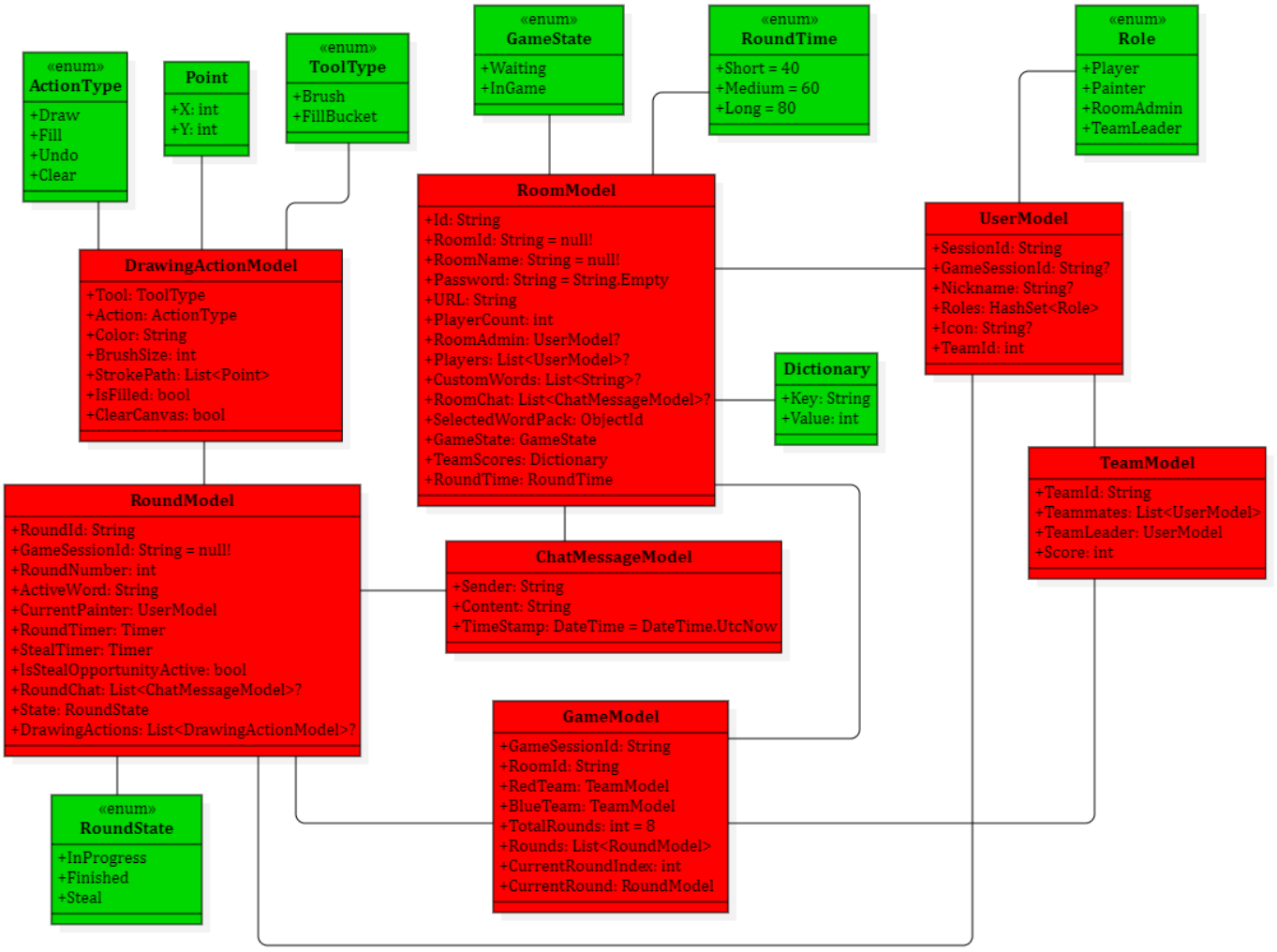
Faza II - Model podataka i model perzistencije

Pavle Antonijević 18077
Dušan Petrović 18329

Sadržaj

Model podataka

Model podataka u aplikaciji DrawOut je predstavljen sledećim klasnim dijagramom:



Slika 1. Dijagram modela podataka

- **RoomModel** - Klasa koja predstavlja model sobe gde se korisnici okupljaju i u kojoj se odvija igra. Svaka soba ima naziv, barem jednog administratora sobe, URL, trenutni broj korisnika u sobi, jedinstveni čet sobe (koji mogu da koriste svi korisnici), kao i odabrani paket reči. Pored toga, svaka soba ima stanje koje opisuje igru koja se trenutno odvija, kao i odabranu dužinu trajanja jedne runde. Takođe, administrator sobe može dodati svoje (custom) reči u odabrani paket reči.
- **UserModel** - Klasa koja predstavlja model korisnika u aplikaciji. Korisnik može postaviti svoj nadimak i izabrati jednu od ponuđenih ikonica. Ukoliko to ne učini, biće mu nasumice dodeljen nadimak i ikonica. Svaki korisnik se može priključiti

sobi, dok se unutar sobe može priključiti jednom od dva tima (crveni i plavi). U svakom trenutku korisnik može imati jednu ili više uloga, koje su definisane unutar ***Role*** enumeracije.

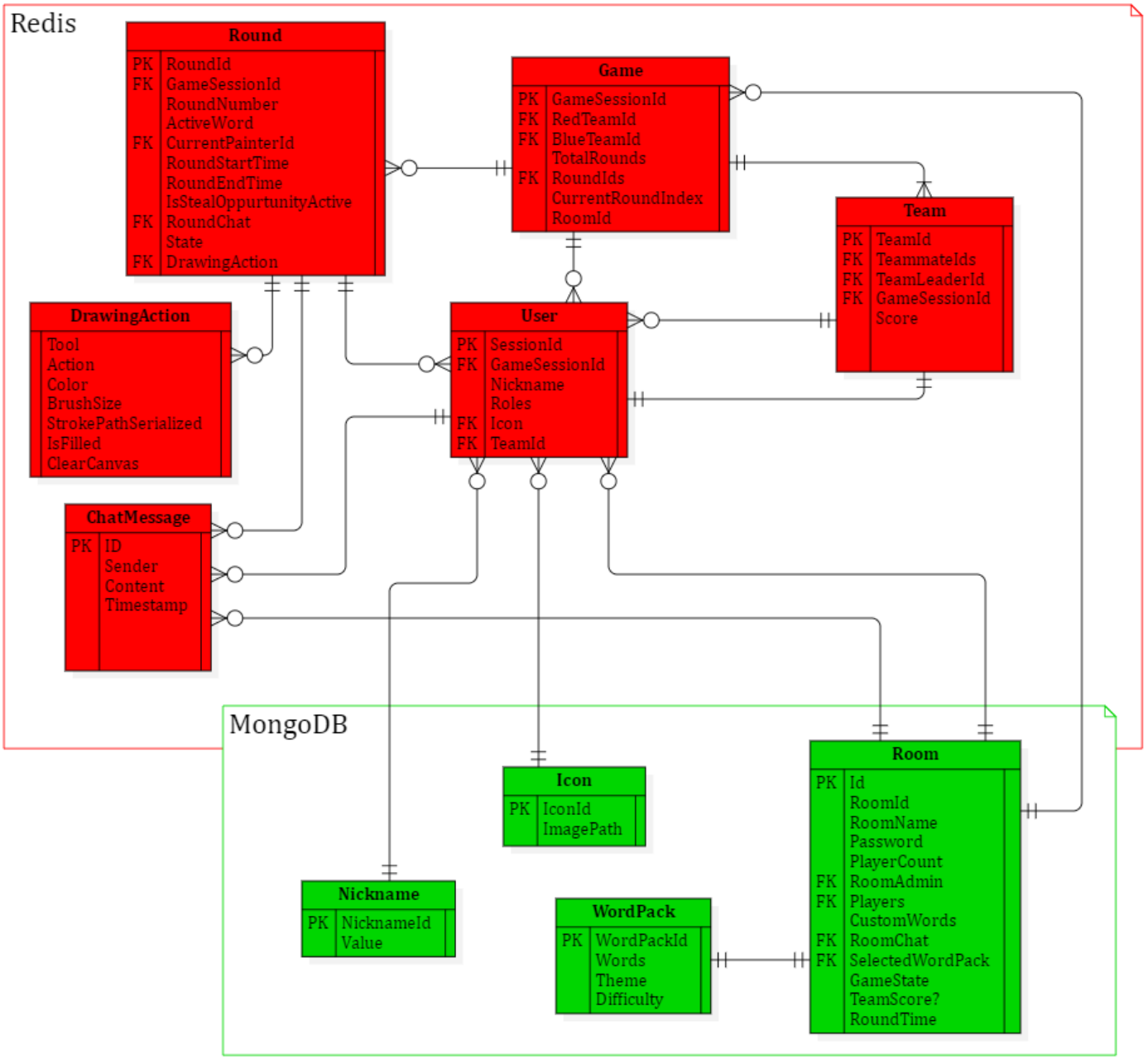
- **GameModel** - Klasa koja predstavlja glavni model za sesiju igre. Pored identifikatora sesije, poseduje i objekte za crveni i plavi tim (tip ***TeamModel***). Svaka igra se sastoji od tačno 8 partija, pa zbog toga poseduje listu čiji su elementi objekti tipa ***RoundModel***.
- **RoundModel** - Klasa koja opisuje model pojedinačne runde u igri. Svaka runda sadrži indikator broja runde, aktivnu reč koja se pogađa, slikara (korisnika) čiji je zadatak da nacrta aktivnu reč, njegov crtež, jedinstveni čet runde. Pored toga, svaka runda može biti u jednom od 3 stanja predstavljenih unutar enumeracije ***RoundState***. Takođe, postoje tajmeri za merenje vremena runde i vremena krađe.
- **TeamModel** - Klasa koja opisuje model tima unutar igre. Uključuje identifikator tima, listu članova tima (tipa **UserModel**) i ukupan broj poena koji je tim ostvario.
- **ChatMessageModel** - Klasa koja opisuje model poruke unutar čet sistema aplikacije. Svaka poruka sadrži informacije o pošiljaocu, sadržaju poruke, kao i vremenskoj oznaci koja pokazuje kada je poruka poslata.
- **DrawingActionModel** - Klasa koja služi kao reprezentacija jedne akcije crtanja. Sadrži informacije o izabranom alatu, vrsti akcije, izabranoj boji, veličini četke, putanji poteza, kao i dva bool
- **GameState** - Enumeracija koja definiše različita stanja u kojima se igra može nalaziti. Ova stanja uključuju: *Waiting* (čekanje) za period pre početka igre, *InGame* (u igri) kada je igra aktivna.
- **RoundState** - Enumeracija koja definiše skup mogućih stanja u kojima se može naći pojedinačna runda. Stanja su: *InProgress* (u toku), *Finished* (završena) i *Steal* (mogućnost krađe). Kada je runda u stanju *Steal*, to implicira da tim, koji je prvobitno imao zadatak da nacrta i pogodi aktivnu reč, nije uspeo u toj nameri. Već se sada pruža mogućnost suprotnom timu da dogovorom i za određeno vreme pokuša da “ukrade” poene.
- **RoundTime** - Enumeracija koja sadrži skup mogućih dužina trajanja runde u igri. Uključuje vrednosti *Short*, *Medium* i *Long*, koje definišu vreme trajanja runde u

sekundama. Ovo omogućava korisnicima da prilagode trajanje rundi prema željenoj dinamici igre.

- **Role** - Enumeracija koja definiše skup mogućih uloga koje korisnik može imati unutar aplikacije. Uloge uključuju *Player* (igrač), *Painter* (slikar), *RoomAdmin* (administrator sobe) i *TeamLeader* (vođa tima). Svaka uloga ima određene privilegije i odgovornosti unutar aplikacije.
- **ActionType** - Enumeracija koja opisuje skup mogućih akcija koje korisnik može preduzeti prilikom crtanja. Ove akcije uključuju: *Draw* (crtanje), *Fill* (popunjavanje), *Undo* (poništavanje) i *Clear* (kompletno brisanje).
- **ToolType** - Enumeracija koja određuje vrste alata dostupne korisnicima za crtanje. Sadrži *Brush* (četku) za crtanje linija i *FillBucket* (kantu za boju) za popunjavanje oblasti bojom.
- **Point** - Pomoćna klasa koja se koristi na klasom dijagramu kako bi se prikazao skup koordinata, dok se u kodu zamenjuje linijom *List<int X, int Y>*.
- **Dictionary** - Pomoćna klasa koja se koristni na klasom dijagramu kako bi se prikazao rečnik, gde je ključ tipa String, dok je vrednost ključa tipa int.

Model perzistencije

Prethodno predstavljeni model podataka se na odgovarajući način perzistira u bazi podataka, predtavljen u vidu modela entiteta:



Slika 2. Dijagram modela perzistencije

Mehanizmi mapiranja

Za perzistenciju i rad sa bazom podataka korišćeni su MongoDB i Redis, dok je za mapiranje između objekata klasa iz modela podataka i entiteta baze podataka korišćen je tip OOM-a, Data Mapper Pattern. Princip koji se koristi za mapiranje je code-first. Prilikom mapiranja je primenjen DataLayer obrazac **Repository**.