Remy

Društvena kartaška igra

Arhitekturni projekat

Verzija 1.0

Pregled izmena

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Verzija** | **Opis** | **Autor** |
| 26.04.2018. | 1.0 | Inicijalna verzija | Nenad, Stefan |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Sadržaj

1. Cilj dokumenta 4

2. Opseg dokumenta 4

3. Reference 4

4. Predstavljanje arhitekture 4

5. Ciljevi i ograničenja arhitekture 4

6. Pogled na slučajeve korišćenja 4

6.1 Dijagrami slučajeva korišćenja 6

6.2 Kratak opis slučajeva korišćenja 8

6.2.1 Uređivanje profila 8

6.2.2 Pregled statistike 8

6.2.3 Chat servis 8

6.2.4 Q&A 8

6.2.5 Priključiti se igri 8

6.2.6 Napustiti igru 8

6.2.7 Chat(in-game) 8

6.2.8 Pozvati prijatelja 8

6.2.9 Kreirati igru 8

6.2.10 Prijavljivanje 8

7. Pogled na logičku arhitekturu sistema 8

7.1 Pregled arhitekture – organizacija paketa i podsistema u slojeve 9

7.1.1 Korisnički interfejs 9

7.1.2 Aplikaciona logika 9

7.1.3 Pristup podacima 9

7.1.4 C# 10

7.1.5 MySQL 10

8. Pogled na implementaciju sistema 10

8.1 Model domena 10

9. Performanse 10

10. Kvalitet 11

Arhitekturni projekat

# Cilj dokumenta

Cilj ovog dokumenta je detaljni opis arhitekture Remi softverskog proizvoda.

# Opseg dokumenta

Dokument se odnosi na softverski proizvod koji će biti razvijen od strane PineCard tima. Namena proizvoda je

zabava društvenog karaktera.

# Reference

Spisak korišćene literature:

1. Remy– – Predlog projekta, V1.0, 2018, PineCard.
2. Remy– – Vizija sistema, V1.0, 2018, PineCard.
3. Remy– – Plan realizacije projekta, V1.0, 2018, PineCard.
4. Remy– – Raspored aktivnosti, V1.0, 2018, PineCard.
5. Remy– – Specifikacija zahteva, V1.0, 2018, PineCard.

# Predstavljanje arhitekture

Arhitektura sistema u dokumentu je prikazana kao serija pogleda na sistem: pogled na slučajeve korišćenja, pogled na logičku arhitekturu sistema, pogled na procese, pogled na razmeštaj komponenti sistema i pogled na implementaciju. Ovi pogledi su predstavljeni odgovarajućim UML dijagramima.

# Ciljevi i ograničenja arhitekture

Ključni zahtevi i sistemska ograničenja koja imaju značajan uticaj na izbor arhitekture i projektovanje sistema su:

1. Remy sistem će biti implementiran kao Desktop aplikacija zasnovana na Unity engine-u, PHOTON serveru i bazi podataka.
2. Svi zahtevi u pogledu performansi dati u [5] moraju biti uzeti u obzir pri izboru arhitekture i razvoju sistema.

# Pogled na slučajeve korišćenja

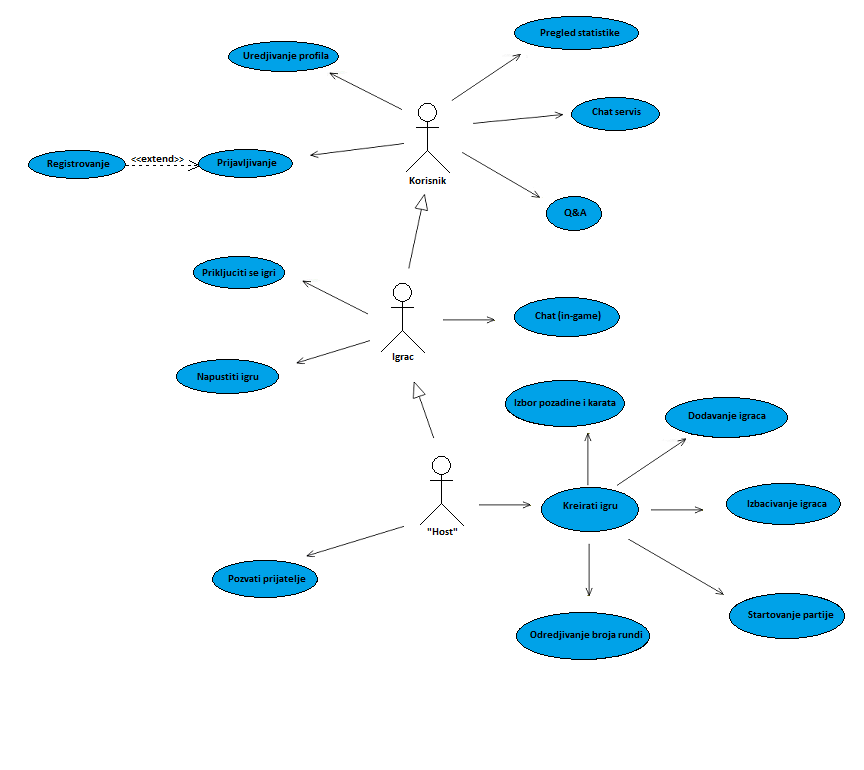
U ovom odeljku je dat pogled na slučajeve korišćenja definisane u specifikaciji zahteva.

Slučajevi korišćenja Remy softverskog proizvoda su:

* *Prijavljivanje*
* *Uredjivanje profila*
  + Azuriranje podataka
  + Promena slike
  + Izbor karata
* *Pregled statistike*
  + Pobede igraca
  + Gubitci igraca
  + Odigranih partija
* *Chat servis*
  + Slati i primati poruke
  + Slati zahtev za prijateljstvo
* *Q&A*
* *Prikljuci se igri*
* *Napusti igru*
* *Chat(in-game)*
* *Pozvati prijatelja*
* *Kreirati igru*
  + Izbor pozadine karata
  + Dodavanje igraca
  + Izbacivanje igraca
  + Odredjivanje broja rundi
  + Startovanje Partije

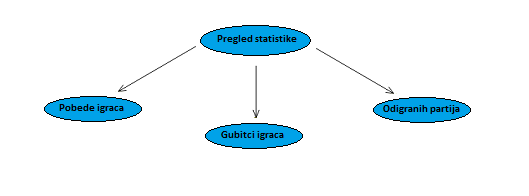
Ove slučajevi korišćenja mogu da iniciraju **igraci** i **hostovi**.

## Dijagrami slučajeva korišćenja

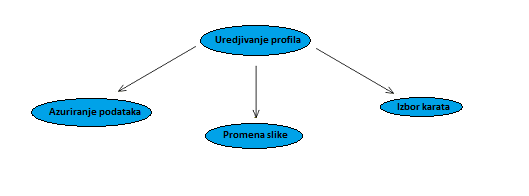
Osnovni UML dijagram koji prikazuje korisnike i slučajeve korišćenja softvera prikazan je na sledećoj slici:

Slučajevi korišćenja *pregled statistike, uredjivanje profila* i *chat servis-a* obuhvataju složenije radnje koje se mogu

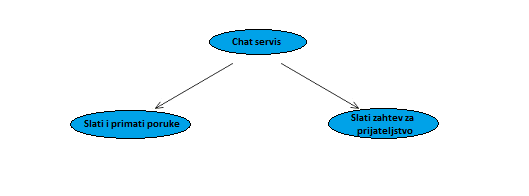
razložiti dalje razložiti na pojedinačne slučajeve korišćenja.

Detaljni UML dijagram za slučaj korišćenja *pregled statistike* je prikazan na sledećoj slici:

Detaljni UML dijagram za slučaj korišćenja *uredjivanje profila*  je prikazan na sledećoj slici:



Detaljni UML dijagram za slučaj korišćenja *chat servis-a* je prikazan na sledećoj slici:



## Kratak opis slučajeva korišćenja

### Uredjivanje profila

**Kratak opis:** Izbor slike, naziva i karata igraca.

**Akteri:** Igrac.

### Pregled statistike

**Kratak opis:** Prikaz prozora sa informacijama o odigranim partijama igraca.

**Akteri:** Igrac.

### Chat servis

**Kratak opis:** Medjusobna komunikacija korisnika.

**Akteri:** Igrac.

### Q&A

**Kratak opis:** Komunikacija sa proizvodjacem.

**Akteri:** Igrac.

### Prikljuciti se igri

**Kratak opis:** Ulazak u “lobby” za partiju.

**Akteri:** Igrac, Host.

### Napustiti igru

**Kratak opis:** Napustanje “lobby” za partiju

**Akteri:** Igrac, Host.

### Chat(in-game)

**Kratak opis:** Medjusobna komunikacija korisnika.

**Akteri:** Igrac, Host.

### Pozvati prijatelja

**Kratak opis:** Slanje poziva za igru prijatelju.

**Akteri:** Host.

### Kreirati igru

**Kratak opis:** Kreiranje “lobby” za partiju remija.

**Akteri:** Host.

### Prijavljivanje

**Kratak opis:** Prijavljivanje korisnika na serveru.

**Akteri:** Korisnici.

# Pogled na logičku arhitekturu sistema

U ovom odeljku je dat pregled logičke arhitekture sistema. Ovaj pogled sadrži opis najznačajnijih klasa, njihove organizacije u pakete i podsisteme, i organizacija podsistema u slojeve. U cilju opisivanja dinamičkih aspekata arhitekture, ovaj odeljak može da uključi opise realizacije najznačajnijih slučajeva korišćenja. Da bi se ilustrovala veza između arhitekturno značajnih klasa, podsistema, paketa ili slojeva moguće je uključiti i odgovarajuće dijagrame klasa.

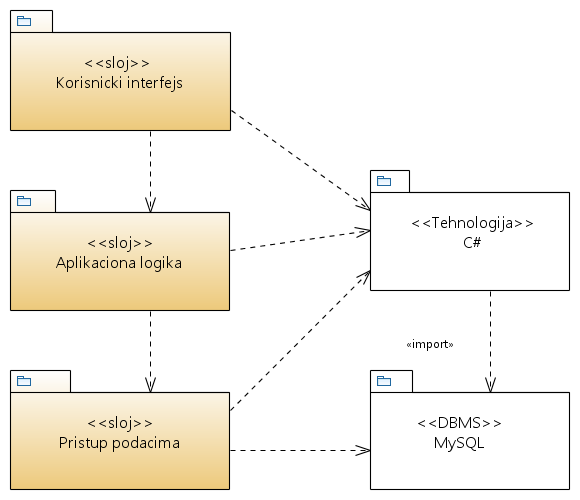
Logički pogled na Remy sistem obuhvata 3 glavna paketa: Korisnički interfejs, Aplikaciona logika, Pristup podacima.

Paket *Korisnički interfejs* sadrži C# skripte i multimedijalni sadržaj koji realizuju grafički dizajn i forme preko kojih korisnici sistema komuniciraju sa sistemom.

Paket *Aplikaciona logika* predstavlja srednji sloj sistema koji sadrži C# skripte zadužene za realizaciju funkcionalnosti specifičnih za domen sistema koji se razvija.

Paket *Pristup podacima* sadrži C# skripte koje predstavljaju interfejs za pristup, dodavanje i ažuriranje podataka koji se čuvaju u bazi podataka.

## Pregled arhitekture – organizacija paketa i podsistema u slojeve



### Korisnički interfejs

sloj

Ovaj sloj realizuje korisnički interfejs sistema. U njemu su sadržane sve C# skripte koje generišu prozore preko kojih korisnici komuniciraju sa sistemom.

Sloj korisničkog interfejsa zavisi od sloja aplikacione logike, kao i paketa tehnologija C#.

### Aplikaciona logika

sloj

Sloj aplikacione logike je srednji sloj u troslojnoj arhitekturi Remi softverskog proizvoda. Sadrži C# skripte koje realizuju funkcionalnost i uspostavljaju vezu između korisničkog interfejsa i sloja za pristup podacima.

Ovaj sloj zavisi od sloja za pristup podacima i C# paketa.

### Pristup podacima

sloj

Sloj za pristup podacima se nalazi na dnu troslojne arhitekture i sadrži C# skripte zadužene za pribavljanje, dodavanje i ažuriranje podataka koji se čuvaju u MySQL bazi podataka.

Ovaj sloj ne zavisi od drugih slojeva, ali je zavisan od paketa C# i MySQL baza podataka.

### PHOTON

tehnologija

Tehnologija PHOTONA-a obezbeđuje mehanizam za pisanje i izvršavanje skripti na strani servera. Ove skripte mogu da generišu kod koji realizuje korisnički interfejs i pristupaju bazi podataka u cilju pribavljanja, unosa i ažuriranja podataka.

### MySQL

DBMS

MySQL predstavlja sistem za upravljanje bazama podataka koji će se koristiti za realizaciju softvera.

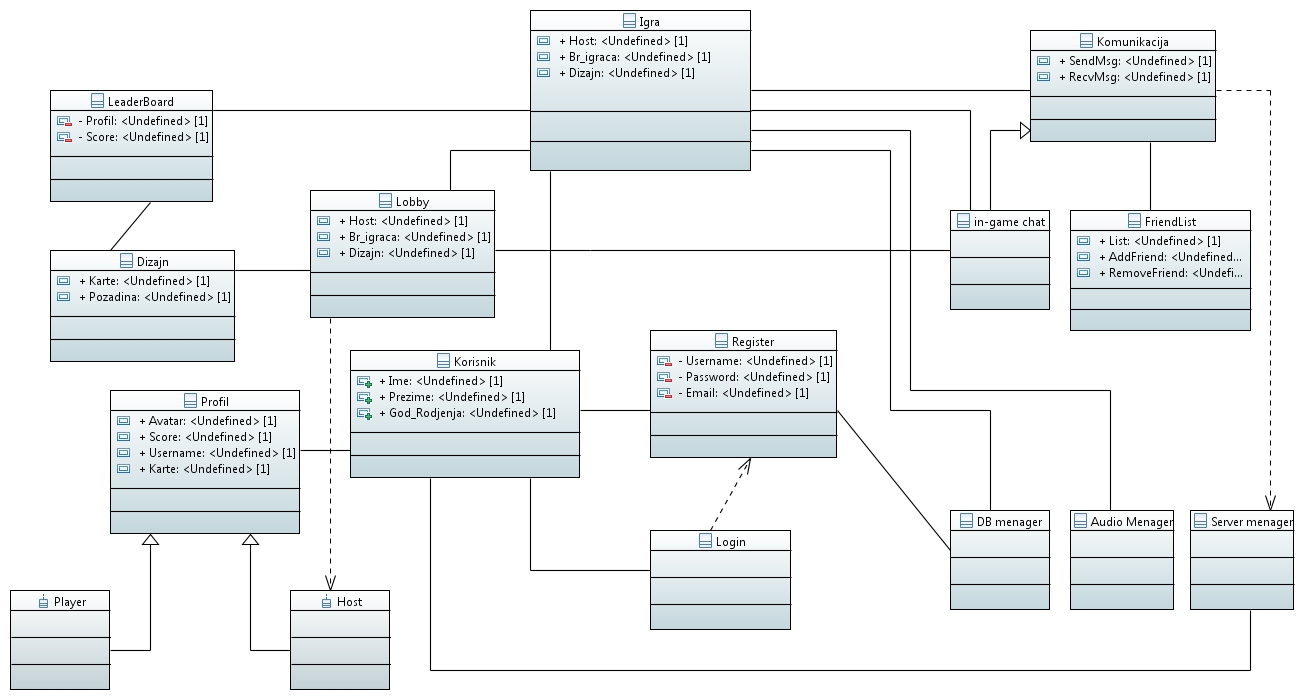
# Pogled na implementaciju sistema

Pogled na implementaciju prikazuje različite aspekte bitne za implementaciju sistema. U slučaju Remy sistema ovaj odeljak sadrži model domena.

## Model domena

Model domena za koji se softver projektuje je ilustrovan UML dijagramom klasa. U njemu su prikazane domenske klase, neki od njihovih atributa, kao i veze koje se mogu identifikovati između njih.

Model domena predstavlja osnovu za projektovanje baze podataka, ali i identifikaciju nekih od komponenti (C# skripti) koje će biti implementirane.



# Performanse

Izabrana arhitektura softvera podržava zahteve u pogledu broja korisnika koji mogu simultano pristupati sistemu i vremena odziva za pristup bazi podataka specificirane u zahtevima u pogledu performansi:

1. Sistem će da podrži do 100 simultanih pristupa korisnika serveru.
2. Vreme potrebno za pristupanje bazi podataka u cilju izvršenje nekog upita ne sme da bude veće od 5 sekundi.

Zahtevane performanse su zadovoljene izborom tehnologija na kojima će sistem biti razvijen i definisane hardverske platforme.

# Kvalitet

Izabrana arhitektura softvera podržava zahteve u pogledu dostupnosti i srednjeg vremena između otkaza specificirane u zahtevima u pogledu pouzdanosti [5]:

1. Server će biti dostupan 24 časa dnevno, 7 dana u nedelji. Vreme kada server nije dostupan ne sme da pređe 10%.
2. Srednje vreme između dva sukcesivna otkaza ne sme da padne ispod 120 sati.