Riziko: Alea iacta est

Online društvena igra Riziko

Arhitekturni projekat

Verzija 1.0

Pregled izmena

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Verzija** | **Opis** | **Autor** |
| 22.04.2021. | 1.0 | Inicijalna verzija | Lazar Ignjatović |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Sadržaj

1. Cilj dokumenta 5

2. Opseg dokumenta 5

3. Reference 5

4. Predstavljanje arhitekture 5

5. Ciljevi i ograničenja arhitekture 5

6. Pogled na slučajeve korišćenja 5

6.1 Dijagrami slučajeva korišćenja 6

6.2 Kratak opis slučajeva korišćenja 7

6.2.1 Kreiranje Guest naloga 7

6.2.2 Prijavljivanje 7

6.2.3 Registracija 7

6.2.4 Igranje partije 8

6.2.5 Promena avatara 8

6.2.6 Pregled svog naloga(statistika) 8

6.2.7 Settings(Podešavanja igre) 8

6.2.8 Pregled leaderboard-a 8

6.2.9 Casual game 8

6.2.10 Ranked game 8

6.2.11 Play with friends 8

6.2.12 Izbor mape 8

6.2.13 Podešavanja partije 8

6.2.14 Ulaz u partiju 8

6.2.15 Tok partije 8

6.2.16 Završetak partije 8

6.2.17 Podešavanja Gameplay-a 9

6.2.18 Podešavanja zvuka 9

6.2.19 Help & support 9

7. Pogled na logičku arhitekturu sistema 9

7.1 Pregled arhitekture – organizacija paketa i podsistema u slojeve 9

7.1.1 Korisnički interfejs 10

7.1.2 Korisnička logika 10

7.1.3 Serverska logika 10

7.1.4 Pristup podacima 10

7.1.5 Unity 10

7.1.6 SignalR 10

7.1.7 .NET 10

7.1.8 EntityFramework 10

7.1.9 SQLExpress 10

8. Pogled na procese 10

8.1 Procesi 11

8.1.1 Aplikacija 11

8.1.2 Server 11

8.1.3 SQL Server 11

9. Pogled na raspoređivanje sistema 11

9.1 Klijent 11

9.2 Server 12

9.3 DBMS server 12

10. Pogled na implementaciju sistema 12

10.1 Model domena 12

10.2 Šema baze podataka 12

10.3 Komponente sistema 13

10.3.1 Korisnički interfejs 15

10.3.2 Korisnička logika 15

10.3.3 Pristup podacima 16

10.3.4 Serverska logika 16

11. Performanse 17

12. Kvalitet 17

Arhitekturni projekat

# Cilj dokumenta

Cilj ovog dokumenta je detaljni opis arhitekture Riziko: Alea iacta est igre.

# Opseg dokumenta

Dokument se odnosi na Riziko: Alea iacta est igru koji će biti razvijena od strane DeGaa tima. Namena igre je pre svega kvalitetna zabava i druženje korisnika.

# Reference

Spisak korišćene literature:

1. Riziko: Alea iacta est – Predlog projekta, DeGaa-Riziko: Alea iacta est-01, V1.0, 20212, DeGaa.
2. Riziko: Alea iacta est – Planirani raspored aktivnosti na projektu, V1.0, 2021, DeGaa.
3. Riziko: Alea iacta est – Plan realizacije projekta, V1.0, 2021, DeGaa.
4. Riziko: Alea iacta est – Vizija sistema, V1.0, 2021, DeGaa.
5. Riziko: Alea iacta est – Specifikacija zahteva, V1.0, 2021, DeGaa.

# 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. Riziko: Alea iacta est igra će biti implementirana kao Desktop i Android aplikacija zasnovana na UnityEngine-u, SignalR i .NET tehnologijama.
2. Klijentski deo Riziko: Alea iacta est igre će biti optimizovan za Desktop računare sa Windows operativim sistemom i Android mobilne uređaje.
3. 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 [5].

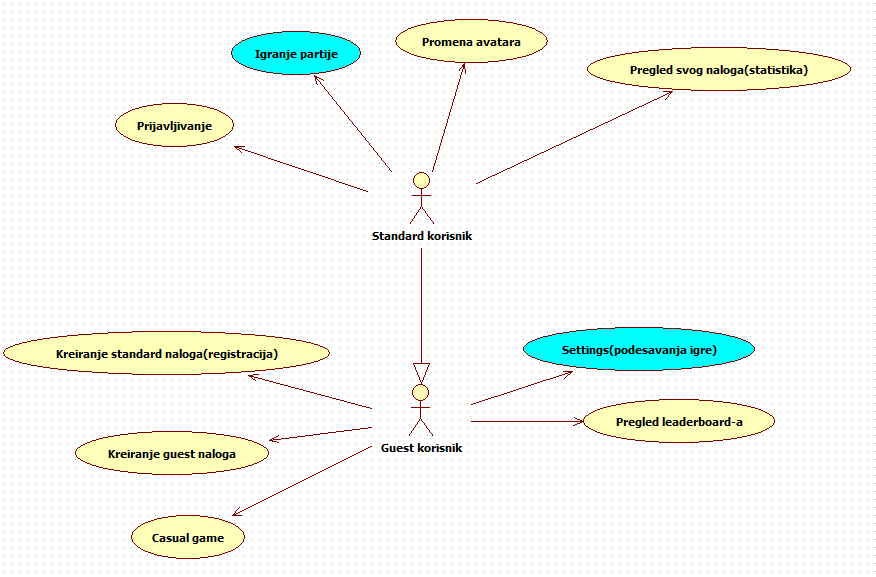
Slučajevi korišćenja Riziko: Alea iacta est igre su:

* Kreiranje Guest naloga
* Prijavljivanje
* Registracija
* Igranje partije
* Promena avatara
* Pregled svog naloga(statistika)
* Settings(Podešavanja igre)
* Pregled leaderboard-a
* Casual game
* Ranked game
* Play with friends
* Izbor mape
* Podešavanja partije
* Ulaz u partiju
* Tok partije
* Završetak partije
* Podešavanja Gameplay-a
* Podešavanja zvuka
* Help & support

Ove slučajevi korišćenja mogu da iniciraju Guest korisnik i Standard korisnik.

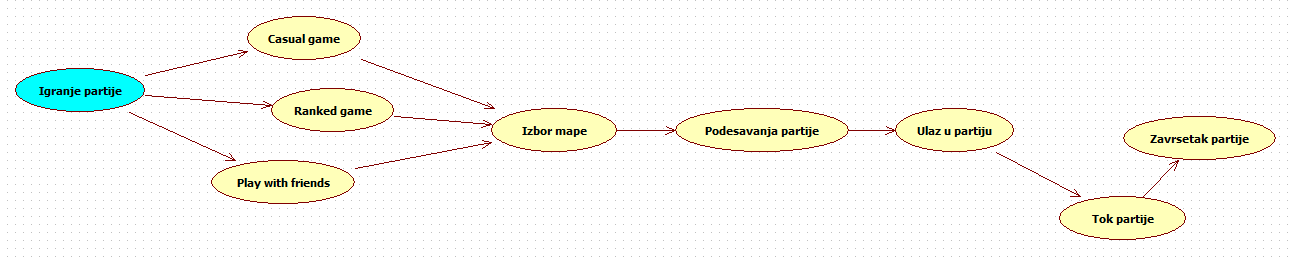
## Dijagrami slučajeva korišćenja

Osnovni UML dijagram koji prikazuje korisnike i slučajeve korišćenja Riziko: Alea iacta est igre prikazan je na sledećoj slici:

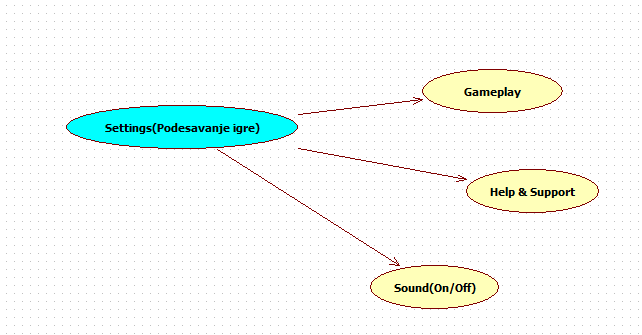


Slučajevi korišćenja *Igranje partije* i *Settings(Podesavanja igre)*obuhvataju složenije radnje koje se mogu razložiti dalje razložiti na pojedinačne slučajeve korišćenja.

Detaljni UML dijagram za slučaj korišćenja *Igranje partije* je prikazan na sledećoj slici:



Detaljni UML dijagram za slučaj korišćenja *Settings(Podesavanja igre)* je prikazan na sledećoj slici:



## Kratak opis slučajeva korišćenja

### Kreiranje Guest naloga

**Kratak opis:** Automatsko kreiranje korsiničkog Guest naloga.

**Akteri:** Guest korisnik

### Prijavljivanje

**Kratak opis:** Prijavljivanje korisnika u cilju pristupa svim funkcionalnostima igre.

**Akteri:** Standard korisnik

### Registracija

**Kratak opis:** Kreiranje novog Standard naloga.

**Akteri:** Guest korisnik

### Igranje partije

**Kratak opis:** Priprema za partiju, ulazak, igranje i završavanje iste. **Akteri:** Standard korisnik, Guest korisnik(samo Casual game omogućen)

### Promena avatara

**Kratak opis:** Promena avatara korisnika. **Akteri:** Standard korisnik, Guest korisnik.

### Pregled svog naloga(statistika)

**Kratak opis:** Prikaz forme sa statistikom korisnika(vreme utrošeno na igri, broj pobeda, broj poraza, ukupni broj partija...). **Akteri:** Standard korisnik.

### Settings(Podešavanja igre)

**Kratak opis:** Prikaz forme za podešavanje igre. **Akteri:** Standard korisnik, Guest korisnik.

### Pregled leaderboard-a

**Kratak opis:** Prikaz forme sa spiskom rangiranih korisnika u rastućem formatu. **Akteri:** Standard korisnik, Guest korisnik.

### Casual game

**Kratak opis:** Igranje obične partije. **Akteri:** Standard user, Guest korisnik.

### Ranked game

**Kratak opis:** Igranje partije koja utiče na leaderboard. **Akteri:** Standard korisnik.

### Play with friends

**Kratak opis:** Igranje partije sa prijateljima. **Akteri:** Standard korisnik.

### Izbor mape

**Kratak opis:** Biranje mape na kojoj će se partija odvijati. **Akteri:** Standard korisnik, Guest korisnik.

### Podešavanja partije

**Kratak opis:** Korisnik bira podešavanja za partiju.  
**Akteri:** Standard korisnik, Guest korisnik.

### Ulaz u partiju

**Kratak opis:** Svi učesnici partije imaju 10 sekundi da potvrde svoje prisustvo u istoj.  
**Akteri:** Standard korisnik, Guest korisnik.

### Tok partije

**Kratak opis:** Trajanje same partije.  
**Akteri:** Standard korisnik, Guest korisnik.

### Završetak partije

**Kratak opis:** Partija se završava kada neki od igrača ispuni svoj zadatak. **Akteri:** Standard korisnik, Guest korisnik.

### Podešavanja Gameplay-a

**Kratak opis:** Korisnik izmenjuje preferirana podešavanja igre  
**Akteri:** Standard korisnik, Guest korisnik

### Podešavanja zvuka

**Kratak opis:** Korisnik menja preferirana podešavanja zvuka  
**Akteri:** Standard korisnik, Guest korisnik

### Help & support

**Kratak opis:** Korisniku se pruža potrebna pomoć u vezi sa aplikacijom  
**Akteri:** Standard korisnik, Guest korisnik

# 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 Riziko: Alea iacta est igru obuhvata 4 glavna paketa: Korisnički interfejs, Korisnička logika, Serverska logika, Pristup podacima.

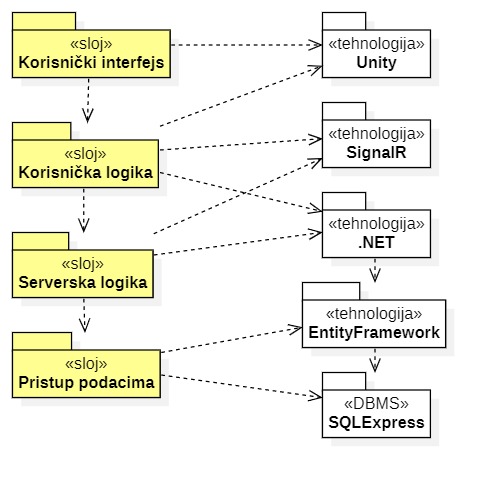
Paket *Korisnički interfejs* sadrži Unity C# skripte i multimedijalni sadržaj koji realizuju grafički dizajn i elemente preko kojih korisnici sistema komuniciraju sa sistemom.

Paket *Korisnička logika* sadrži Unity C# skripte koje omogućavaju komunikaciju sa serverom i obradu serverskih odgovora.

Paket *Serverska logika* predstavlja sloj sistema koji sadrži .NET program zadužen za realizaciju funkcionalnosti specifičnih za igru koji se razvija.

Paket *Pristup podacima* sadrži EntityFramework funkcije 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 igre. U njemu je sadržan sav multimedijalni sadržaja i Unity C# skripte koje kontrolišu grafiički interfejs preko kojeg korisnici komuniciraju sa sistemom.

Sloj korisničkog interfejsa zavisi od sloja korisničke logike, kao i paketa Unity.

### Korisnička logika

sloj

Ovaj sloj realizuje svu potrebnu logiku na korisničkoj strani. Ovaj sloj je zadužen za komunikaciju sa serverom i obradu rezultata te komunikacije. Sadži se od Unity C# skripti koje realizuju potrebne funkcionalnosti. Ovaj sloj zavisi od sloja Aplikacione logike, kao i paketa Unity, .NET i SignalR

### Serverska logika

sloj

Sloj aplikacione logike je treći sloj u četvoroslojnoj arhitekturi Riziko: Alea iacta est igre. Sadrži .NET program koji realizuje funkcionalnosti karakteristične za igru i uspostavlja vezu između korisničke logike i sloja za pristup podacima.

Ovaj sloj zavisi od sloja za pristup podacima, i .NET paketa.

### Pristup podacima

sloj

Sloj za pristup podacima se nalazi na dnu arhitekture i sadrži EntityFramework funkcije zadužene za pribavljanje, dodavanje i ažuriranje podataka koji se čuvaju u SQLExpress bazi podataka.

Ovaj sloj ne zavisi od drugih slojeva, ali je zavisan od paketa EntityFramework i SQLExpress baza podataka.

### Unity

tehnologija

Tehnologija specifično izrađena za razvoj video igara. Sadrži korisnički interfejs i služi za kreiranje grafičkih elemenata igre i korisničke logike.

### SignalR

tehnologija

Tehnologija SignalR služi za klijent-server komunikaciju koja počiva na WebSocket-ima. Podržava i poruke od servera ka klijentu, grupe korisnika i još mnoge funkcionalnosti.

### .NET

tehnologija

Tehnologija bazirana na programskom jeziku C#. Služi za realizaciju komunikacije i serverske logike sistema.

### EntityFramework

tehnologija

EntityFramework je ORM (Object Relational Mapping) tehnologija sa primarnom namenom komunikacije sa i manipulacije baze podataka.

### SQLExpress

DBMS

SQLExpress predstavlja sistem za upravljanje bazama podataka koji će se koristiti za realizaciju Riziko: Alea iacta est igre.

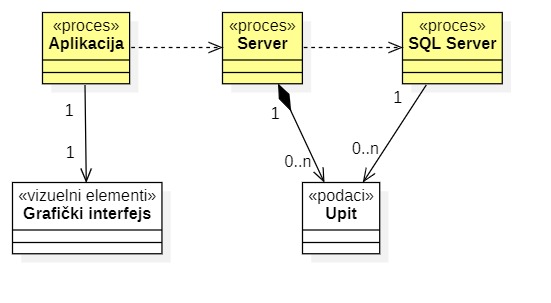
# Pogled na procese

U ovom odeljku je sadržan pogled na procesnu arhitekturu sistema. Ovaj opis treba da sadrži specifikaciju različitih zadataka (procesa i niti) uključenih u rad sistema. Takođe je potrebno dati dijagrame koji pokazuju njihovu interakciju i konfiguraciju. Dodela objekata i klasa na određene zadatke takođe spada u opis procesne arhitekture.

U nastavku je dat opis procesa uključenih u izvršenje Riziko: Alea iacta est igre.

## Procesi

Na sledećem UML dijagramu klasa prikazani su procesi koji učestvuju u izvršenju Riziko: Alea iacta est igre.



### Aplikacija

Aplikacija je proces koji izvršava funkcionalnosti kreiranja i kontrole grafičkog interfejsa. U najopštijem slučaju aplikacija u jednom trenutku može da prikazuje samo jedan grafički interfejs.

Aplikacija zavisi od Servera koji izvršava glavni deo logike igre.

### Server

Server je proces koji izvršava funkcionalnost opsluživanja zahteva prispelih sa više korisničkih aplikacija. Server takodje može da šalje određenim klijentima odgovarajuće poruke bez zahteva. Na serveru se izvršava glavni deo logike igre. Server takođe po potrebi šalje upite bazi podataka. Komunikacija između Serverskog procesa i SQL Servera se obavlja preko prosleđivanja upita i vraćanja rezultata.

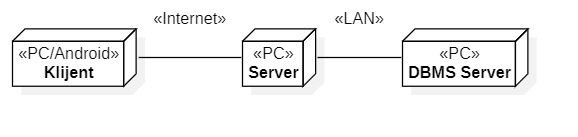
### SQL Server

SQL Server je proces koji izvršava funkcionalnost SQL sistema za upravljanje bazama podataka. Ovaj proces može konkurentno da prihvati određen broj upita, izvrši ih nad bazom podataka i vrati rezultate procesu koji je upite postavio.

# Pogled na raspoređivanje sistema

Pogled na raspoređivanje sistema prikazuje različite fizičke čvorove za najopštiju konfiguraciju sistema. Fizičkim čvorovima koji predstavljaju procesore vrši se dodeljivanje identifikovanih procesa.

Na sledećoj slici dat je UML dijagram raspoređivanja Riziko: Alea iacta est igre.



## Klijent

Pristup Riziko: Alea iacta est igra se obavlja preko klijentskih računara/mobilnih uređaja na kojima se izvršava Riziko: Alea iacta est aplikacija. Za povezivanje između klijenta i Servera koristi se Internet infrastruktura tako da nema ograničenja u pogledu lokacije klijenta.

## Server

Računar na kome se izvršava Server opslužuje više klijenata koji pristupaju preko Interneta. U najopštioj konfiguraciji DBMS se izvršava na posebnoj mašini koja je sa Serverom u lokalnoj mreži (LAN).

## DBMS server

DBMS server je računar na kome se izvršava SQL Server proces koji realizuje funkcionalnost sistema za upravljanje bazama podataka. Zbog sigurnosti podataka koji se na ovom računaru čuvaju pristup bazi je ograničen samo na računare iz lokalne mreže (LAN).

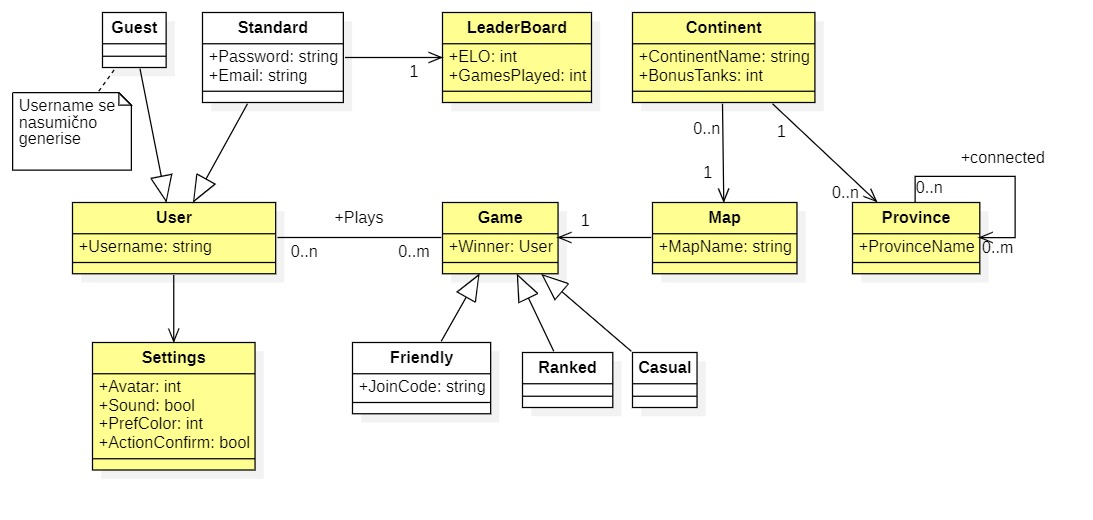
# Pogled na implementaciju sistema

Pogled na implementaciju prikazuje različite aspekte bitne za implementaciju sistema. U slučaju Riziko: Alea iacta est igre ovaj odeljak sadrži model domena, šemu baze podataka i prikaz komponenti sistema razvrstanih u ranije identifikovane pakete.

## Model domena

Model domena za koji se Riziko: Alea iacta est igra 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 koje će biti implementirane.

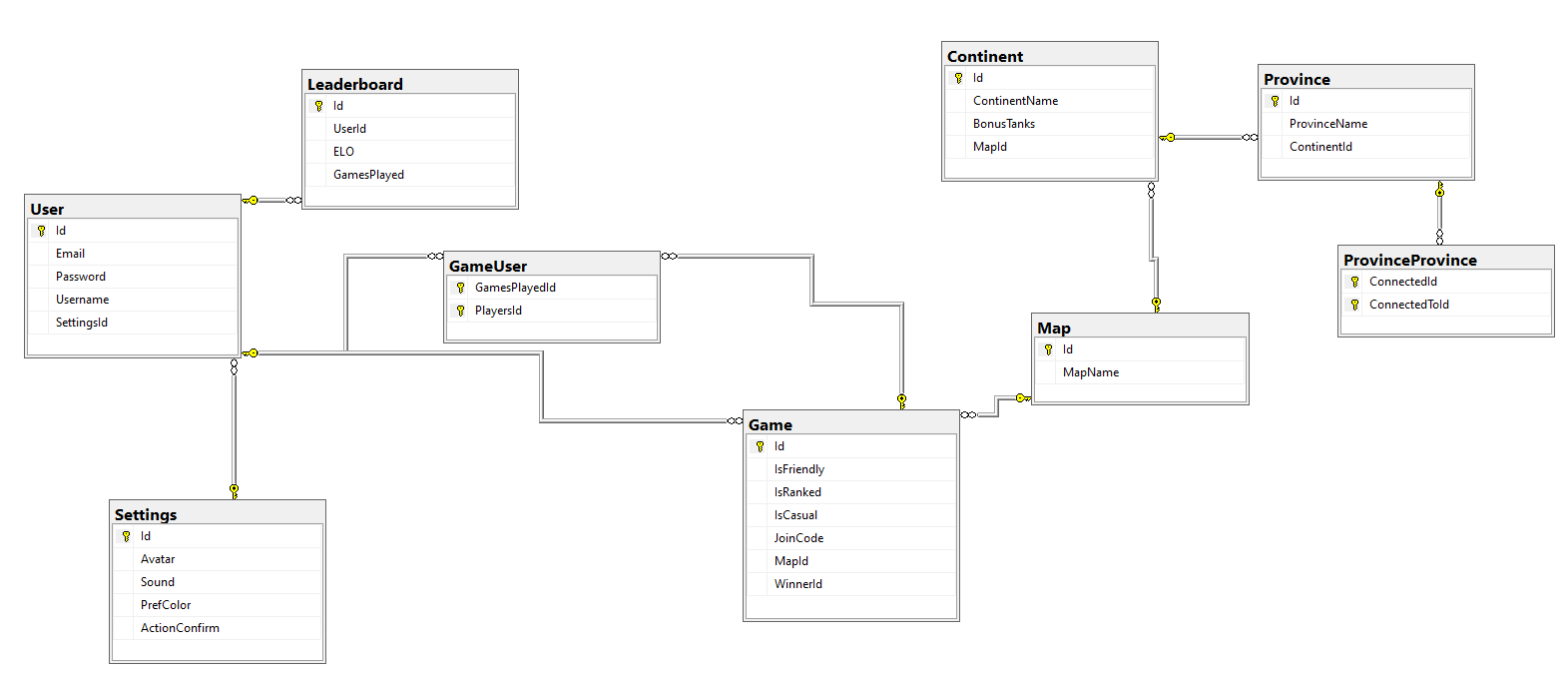


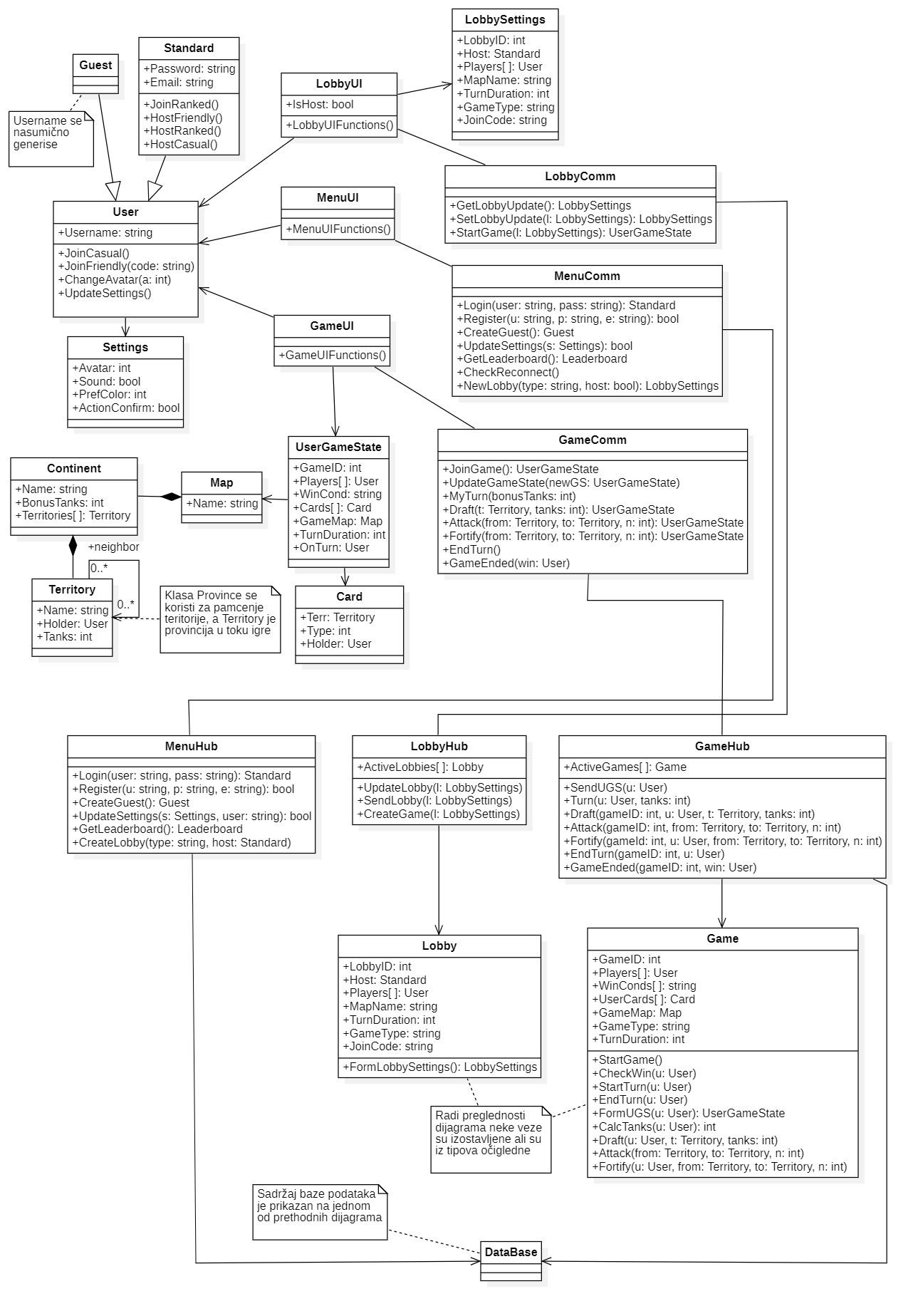
## Šema baze podataka

Detaljna šema baze podataka je prikazana na sledećem dijagramu. Baza podataka je kreirana Code First pristupom u Entity Framework Core-u i dijagram je kreiran korišćenjem Microsoft SQL Management Studio-m.

## Komponente sistema

Komponente sistema Riziko: Alea iacta est igre su Unity i C# programi čiji će pregled biti dat po arhitekturnim slojevima. Za ilustraciju će biti korišćeni UML dijagrami klasa. U nastavku je dat dijagram klasa celokupnog sistema.





### Korisnički interfejs

Dizajn korisničkog interfejsa je obuvaćen klasama MenuUi, LobbyUI i GameUI. Zbog obimnosti ovih klasa i broja funkcija koje one izvršavaju, kao i nemogućnosti da predvidimo precizno njihov celokupan sadžaj, date su u skraćenom obliku. U ovim klasama se obavlja sva logika koja je vezana za korisnički interfejs.

### Korisnička logika

Korisničku logiku realizuju sve klase, van gore navedenih tri, u gornjem delu dijagrama. Detaljnije objašnjenje klasa i funkcija:

#### User

Predstavlja generalizaciju korisničkih klasa. Implementira funkcije koje su dostupne svim korisnicima. Sve funkcije pozivaju odgovarajuće funkcije komunikacionog menadžera da bi se promene reflektovale i na serveru.

* JoinCasual() – poziva odgovarajuću funkciju komunikacionog menadžera za pridruživanje Casual partiji
* JoinFriendly(code: string) – poziva odgovarajuću funkciju komunikacionog menadžera za pridruživanje Friendly partiji pomoću prosledjenog koda
* ChangeAvatar() – promena avatara
* UpdateSettings() – ažuriranje podešavanja

#### Settings

Sadrži podatke o korisničkim podešavanjima.

#### Standard

Specijalizacija korisničke klase. Implementira funkcije karakteristične za Sandard korisnika. Objašnjavanje sadržaja ove klase je suvišno.

#### Guest

Specijalizacija korisničke klase. Predstavlja Guest korisnika.

#### LobbySettings

Sadrži podatke o trenutnom Lobiju u kom je korisnik: id lobija, ko je host, povezani igrači, mapa, trajanje poteza, tip igre i kod za pristup ako se radi o Friendly partiji.

#### UserGameState

Sadrži podatke o partiji u kojoj se korisnik nalazi. Podaci su prilagođeni svakom korisniku. Obuhvata: id partije, listu igrača, uslov za pobedu(kartica za pobedu u običnoj igri uživo), listu kartica koje može razmeniti za tenkove, mapu modelovanu klasom Map(ovjašnjavanje ove klase detaljno je takođe suvišno), trajanje poteza i ko je trenutno na potezu. Ažurira se pri svakoj promeni u toku partije i na osnovu nje se prilagođava korisnički interfejs.

#### MenuComm

Komunikacioni menadžer menija. Odgovoran je za komunikaciju sa serverom. Funkcije:

* Login(in user:string, in pass:string): Standard – šale login zahtev i kao povratnu vrednost daje Standard nalog ulogovanog korisnika.
* Register(in u:string, in p:string, in e:string): bool – zahtev za registraciju, vraća true ako je uspešna
* CreateGuest(): Guest – šalje zahtev za korisničkim Guest nalogom
* UpdateSettings(in s:Settings): bool – ažuriranje korisničkih podešavanja
* GetLeaderboard(): Leaderboard – preuzimanje leaderborda
* CheckReconnect() – provera da li je potrebna rekonekcija u neku partiju
* NewLobby(in type:string, in host:bool): LobbySettings – zahtev za određenim lobijem

#### LobbyComm

Komunikacioni menadžer lobija. Odgovoran je za komunikaciju sa serverom. Funkcije:

* GetLobbyUpdate(): LobbySettings – callback funkcija koja obrađuje ažuriranje lobija
* SetLobbyUpdate(in l:LobbySettings)– host šalje ažuriran loby
* StartGame(in l:LobbySettings): UserGameState – pokretanje partije od strane host-a i dobijanje UserGameState-a od servera

#### GameComm

Komunikacioni menadžer partije. Odgovoran je za komunikaciju sa serverom. Funkcije:

* JoinGame(): UserGameState – callback funkcija koju poziva server i traži od korisnika da se pridruži igri
* UpdateGameState(in newGS:UserGameState) – callback funkcija koju poziva server da ažurira stanje partije.
* MyTurn(in bonusTanks:int) – callback, početak poteza
* Draft(in t:Territory, in tanks:int): UserGameState – šalje zahtev za raspoređivanje tenkova
* Attack(in from:Territory, in to:Territory, in n:int): UserGameState – napad, UserGameState sadrži ishod.
* Fortify(in from:Territory, in to:Territory, in n:int): UserGameState – Transfer tenkova
* EndTurn() – kraj poteza
* GameEnded(in win:User) – callback, kraj partije

### Pristup podacima

Pristup podacima se na serverskoj strani vrši iz klasa MenuHub i GameHub. Dijagram baze podataka je prikazan na prethodnim dijagramima u ovom dokumentu, te da ne bi opterećivali dijagram, objedinjen je u klasu DataBase. Pristup podacima se vrši preko Entity Framework-a.

### Serverska logika

Serversku logiku predstavljaju sve ostale klase koje su predstavljene u donjem delu dijagrama. Veza sa klijentom je na dijagramu označena vezom asocijacije, a zapravo se obavlja preko Interneta.

#### MenuHub

Predstavlja centar za komunikaciju usredsređenu na Meni. Funkcije:

* Login(in user:string, in pass:string): Standard – vrši autentifikaciju i vraća Standard nalog
* Register(in u:string, in p:string, in e:string): bool – vrši registraciju
* CreateGuest(): Guest – kreira Guest nalog
* UpdateSettings(in s:Settings, in user:string): bool – ažurira korisnički nalog
* GetLeaderboard(): Leaderboard – vraća leaderboard
* CreateLobby(in type:string, in host:Standard) – kreira odgovarajući lobi u zavisnosti od parametara

#### LobbyHub

Centar za komunikaciju u lobiju. Sadrži listu aktivnih lobija. Funkcije:

* UpdateLobby(in l:LobbySettings) – ažurira lobi
* SendLobby(in l:LobbySettings) – šalje lobi korisnicima
* CreateGame(in l:LobbySettings) – pokreće partiju sa odgovarajućim parametrima

#### Lobby

Serverska klassa koja predstavlja lobi. Sadrži id lobija, informacije o hostu, povezane igrače, ime mape, trajenje poteza, tip partije, i kod za pristup ako je tip partije Friendly.

Od funkcija sadrži FormLobbySettings(): LobbySettings koji formira objekat koji se prosleđuje korisniku.

#### GameHub

Centar za komunikaciju sa serverom u toku partije. Sadrži listu aktivnih partija. Funkcije:

* SendUGS(in u:User) – šalje UserGameState korisniku
* Turn(in u:User, in tanks:int) – počinje korisnikov potez i dodljuje mu tenkove
* Draft(in gameID:int, in u:User, in t:Territory, in tanks:int) – prihvata zahtev za raspoređivanje tenkova
* Attack(in gameID:int, in from:Territory, in to:Territory, in n:int) – prihvata zahtev za napad
* Fortify(in gameId:int, in u:User, in from:Territory, in to:Territory, in n:int) – prihvata zahtev za transfer tenkova
* EndTurn(in gameID:int, in u:User) – završava korisnikov potez
* GameEnded(in gameID:int, in win:User) – kraj partije

#### Game

Klasa koja izvršava logiku same partije. Sadrži sve potrebne informacije o partiji i status mape. Funkcije:

* StartGame() – pokreće partiju, dodeljuje teritorije i uslove za pobedu
* CheckWin(in u:User) – proverava da li je korisnikov uslov za pobedu ispunjen
* StartTurn(in u:User) – pokreće korisnikov potez
* EndTurn(in u:User) – završava korisnikov potez
* FormUGS(in u:User): UserGameState – formira UserGameState koji se šalje korisniku
* CalcTanks(in u:User): int – izračunava broj tenkova koje korisnik dobija na početku poteza
* Draft(in u:User, in t:Territory, in tanks:int) – izvršava raspoređivanje tenkova
* Attack(in from:Territory, in to:Territory, in n:int) – izvršava napad na teritoriju
* Fortify(in u:User, in from:Territory, in to:Territory, in n:int) – izvršava transfer tenkova

# 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 [5]:

1. Sistem će da podrži do 100 simultanih pristupa korisnika igrau.
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 [5].

# Kvalitet

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

1. Riziko: Alea iacta est igra će biti dostupan 24 časa dnevno, 7 dana u nedelji. Vreme kada igra nije dostupna ne sme da pređe 10%.
2. Srednje vreme između dva sukcesivna otkaza ne sme da padne ispod 120 sati.