Univerzitet u Beogradu
FAKULTET ORGANIZACIONIH NAUKA
Katedra za informacione sisteme

Specifikacija projektnog rada

Tema: Aplikacija za rezervaciju karata za „F1 Grand Prix de Monaco“

Mentor
Vladimir Belča

Student
Marija Grulović 0120/2019

Beograd,
2024.

Sadržaj

Verbalni opis sistema

Softverski sistem F1 Grand Prix de Monaco omogućava korisnicima da lako rezervišu karte za trke Formule 1, prateći istovremeno sve bitne informacije vezane za događaje, učesnike i svoje rezervacije.

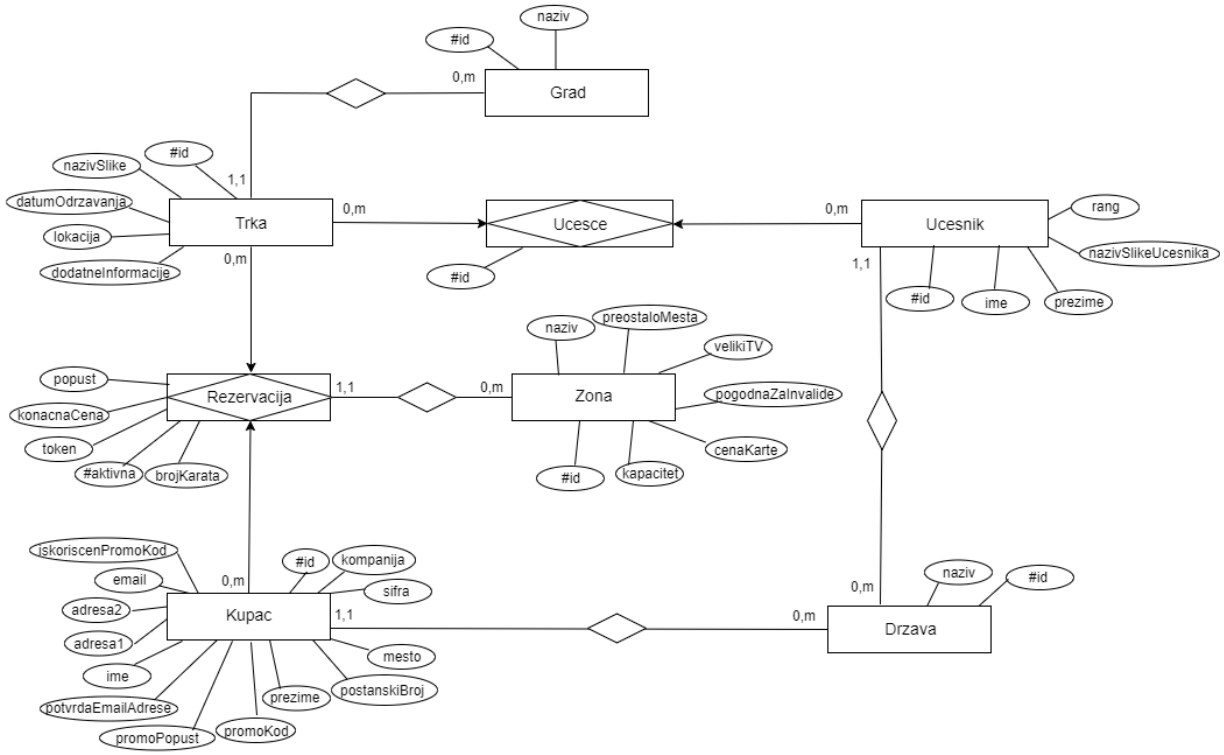
Korisnici softverskog sistema na naslovnoj stranici mogu videti koje se trke održavaju u narednom periodu, koji su trenutno najuspešniji trkači, u kojim trkama će ti trkači učestvovati i koje zone su kupcima na raspolaganju. Za rezervaciju karata i dalji rad na aplikaciji potrebno je kreirati korisnički nalog. Prilikom rezervacije karata korisnici mogu izabrati jednu ili više trka. Ukoliko izaberu samo jednu trku, ne dobijaju popust, dok se za svaki dodatni dan dobija dodatnih 10% popusta. Ukoliko kupci rezervišu karte do dve nedelje pre datuma održavanja trke dobijaju dodatni popust od 10% tzv. *early bird*. Za dodatnih 10% popusta korisnici mogu iskoristiti promo kod svog prijatelja. Korisnici mogu izmeniti ili otkazati rezervaciju prilikom čega je potrebno uneti token koji je generisan prilikom kreiranja rezervacije. Takođe je moguće odabrati različite zone za trku i videti sve pogodnosti koje različite zone nude.

Korišćene tehnologije

Softverski sistem je implementiran kao Single Page Web aplikacija, pri čemu su korišćene sledeće tehnologije:

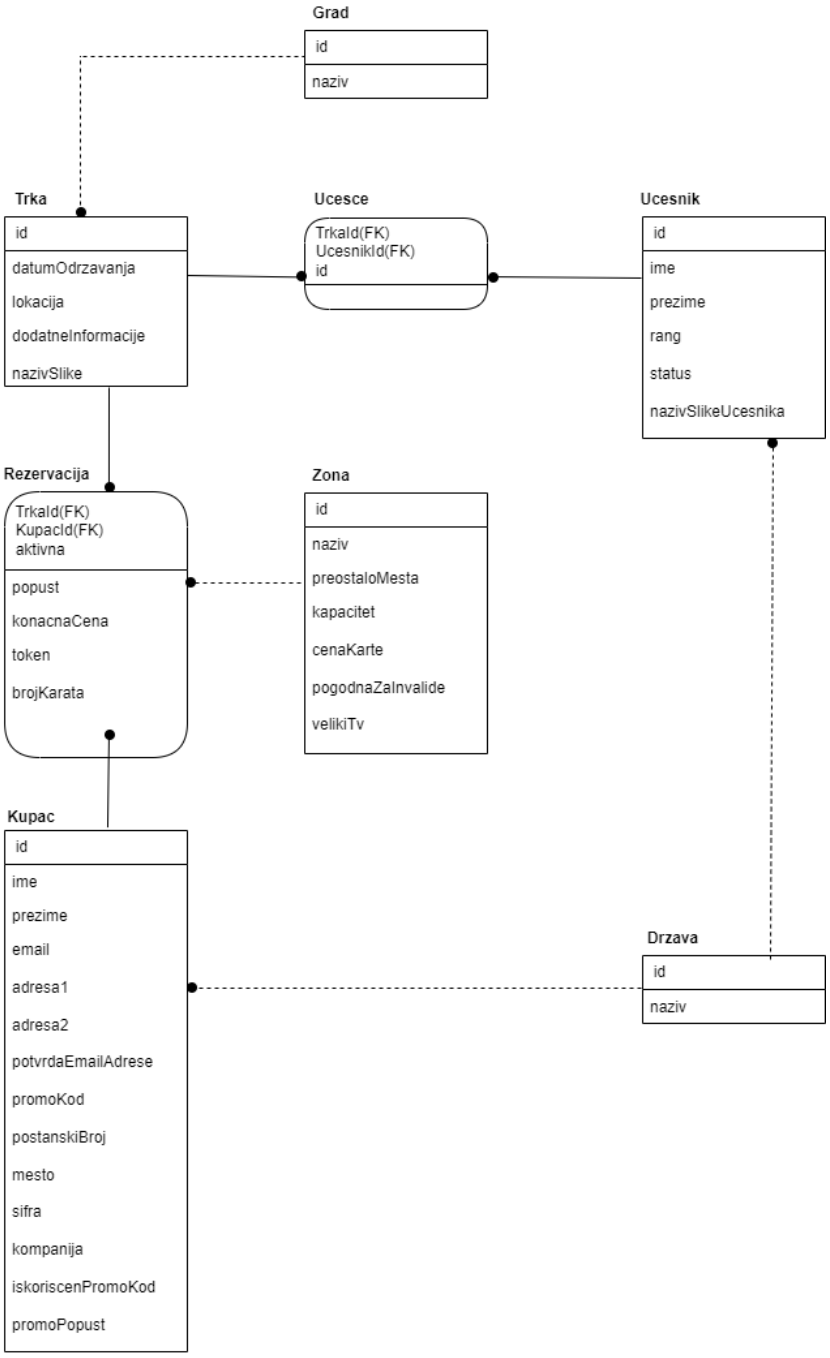
- ASP .NET Web API (back-end tehnologija)
- React (front-end tehnologija)
- Microsoft SQL Server (baza podataka)

Prošireni model objekti veze (PMOV)



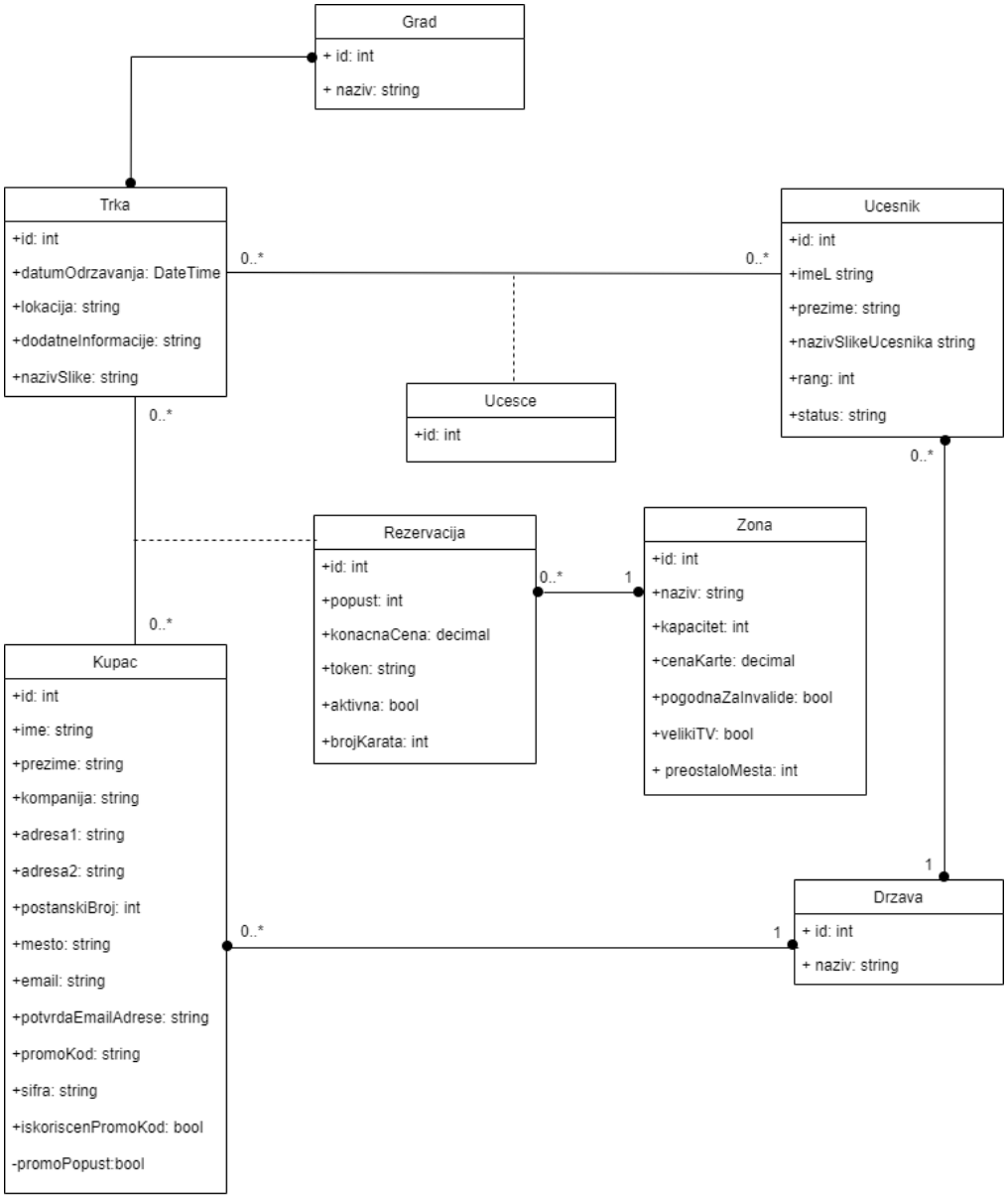
Na osnovu proširenog modela objekti veze, u posmatranom sistemu identifikujemo 8 ključnih objekata tj entiteta: Trka, Učesnik, Učešće, Kupac, Rezervacija, Zona, Grad i Država. Entitet Učešće predstavlja agregaciju izmedju entiteta Trka i Učesnik, usled činjenice da jedna trka ima više učesnika, a jedan učesnik može učestvovati u više trka. Sličan model javlja se i kod objekta Rezervacija koji takođe predstavlja agregaciju ali izmedju objekata Trka i Kupac, i pored id-eva navedenih agregiranih objekata čuvaju se podaci o konačnoj ceni rezervacije, popustu, broju karata, tokenu koji kupac dobija pri rezervaciji i status rezervacije (aktivna ili neaktivna). Za svaku Trku čuvaju se id, lokacija, datum održavanja, naziv slike trke i dodatne informacije o trci. Za Učesnika se pamte id, ime, prezime, rang i naziv slike učesnika, a za kupce pored id-a, imena i prezimena i naziv kompanije, adresa 1, adresa 2, email, potvrda email adrese, promo kod, postanski broj, mesto i šifra. Entitet Zona predstavlja zonu sedenja sa definisanim kapacitetom, cenom karte i brojem preostalih mesta za sedenje, nazivom i pogodnostima poput toga da li je zona prilagođena za invalide i da li ima veliki televizor.

IDEF1X model



Model podataka ostaje nepromenjen u odnosu na prethodni opis PMOV modela. Jedinu razliku u odnosu na PMOV model predstavlja sintaksa kojom se model opisuje, i zbog toga je moguće preslikavanje PMOV modela u model IDEF1X i obratno.

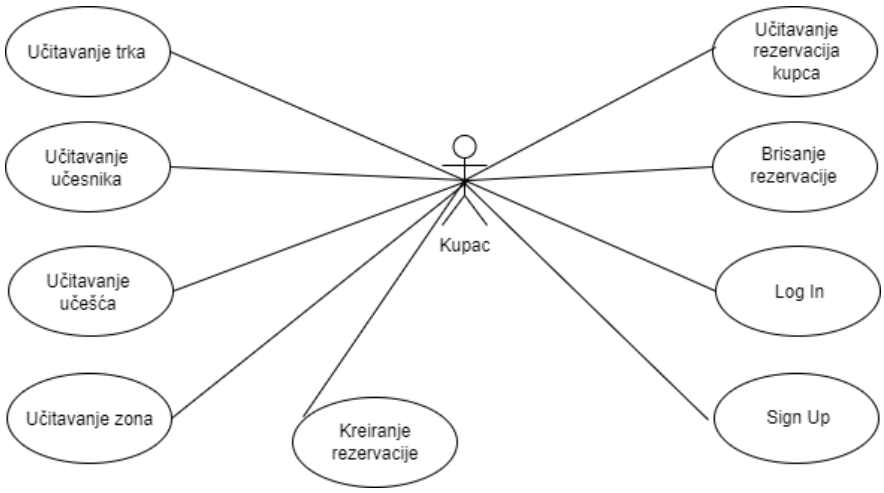
Konacni dijagram klasa



Konačni dijagram klasa predstavlja detaljni model koji se definiše u kasnijim fazama projektovanja sistema, pružajući jasan uvid u način na koji klase u sistemu treba da budu implementirane i povezane međusobno. Dijagram klasa u objedinjenom jeziku za modelovanje (UML) predstavlja tip dijagrama koji opisuje strukturu sistema prikazivanjem klasa sistema, njihovih atributa, operacija (metoda) i odnosa između objekata, pri čemu se prikazuju i tipovi vidljivosti atributa ili metoda.

Softverski zahtevi (features)

1. Učitavanje trka
2. Učitavanje učesnika
3. Učitavanje učešća
4. Učitavanje zona
5. Učitavanje rezervacija kupca
6. LogIn
7. SignUp
8. Kreiranje rezervacije
9. Brisanje rezervacije



1. Učitavanje trka

Akter: Kupac

Preduslovi: /

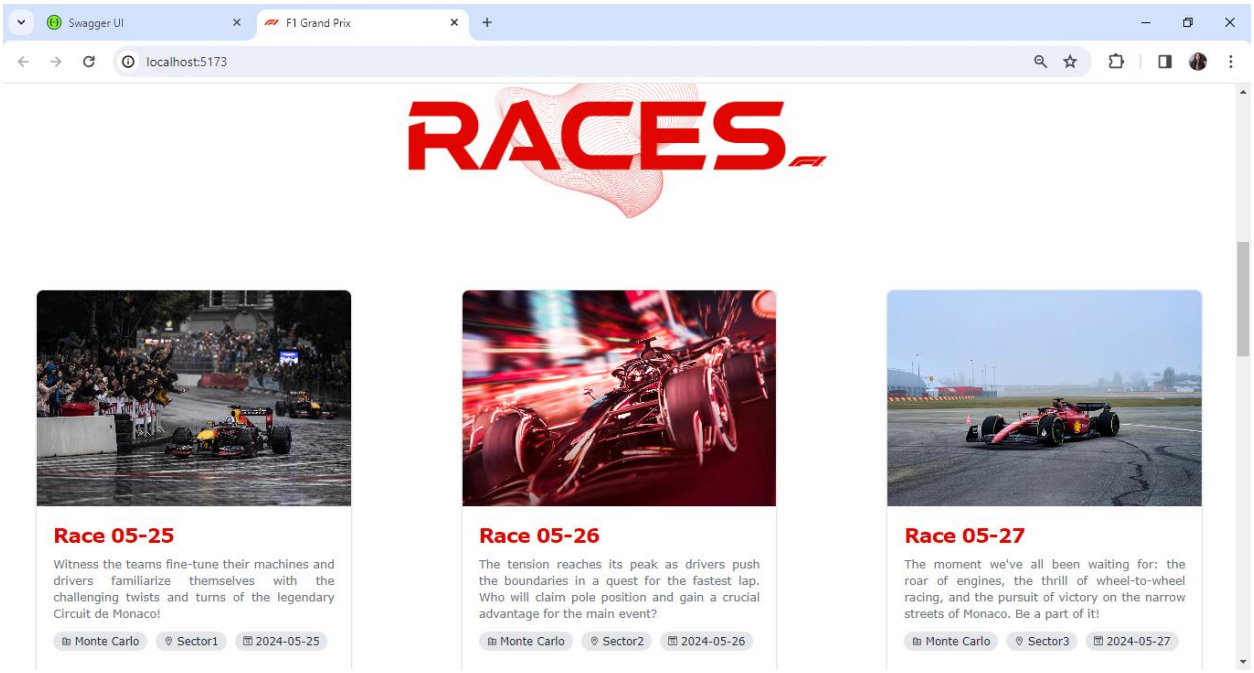
Postuslovi: Prikazana je lista trka

Osnovni scenario:

1. Kupac otvara naslovnu stranu sistema i poziva sistem da učita listu trka
2. Sistem pronalazi i ispisuje listu trka

Alternativni scenario:

2.1 Sistem ne uspeva da učita listu trka



2. Učitavanje učesnika

Akter: Kupac

Preduslovi: /

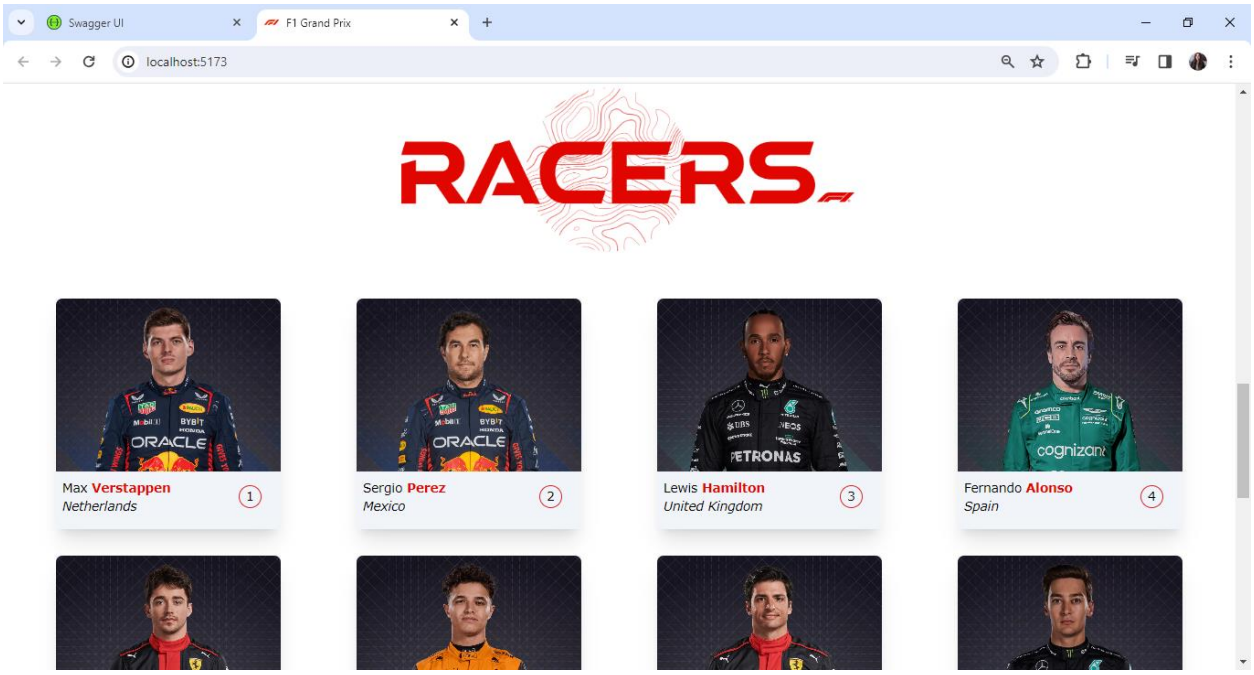
Postuslovi: Prikazana je lista učesnika

Osnovni scenario:

1. Kupac otvara naslovnu stranu sistema i poziva sistem da učita listu učesnika
2. Sistem pronalazi i ispisuje listu učesnika

Alternativni scenario:

2.1 Sistem ne uspeva da učita listu učesnika



3. Učitavanje učešća

Akter: Kupac

Preduslovi: Lista trka je uspešno učitana

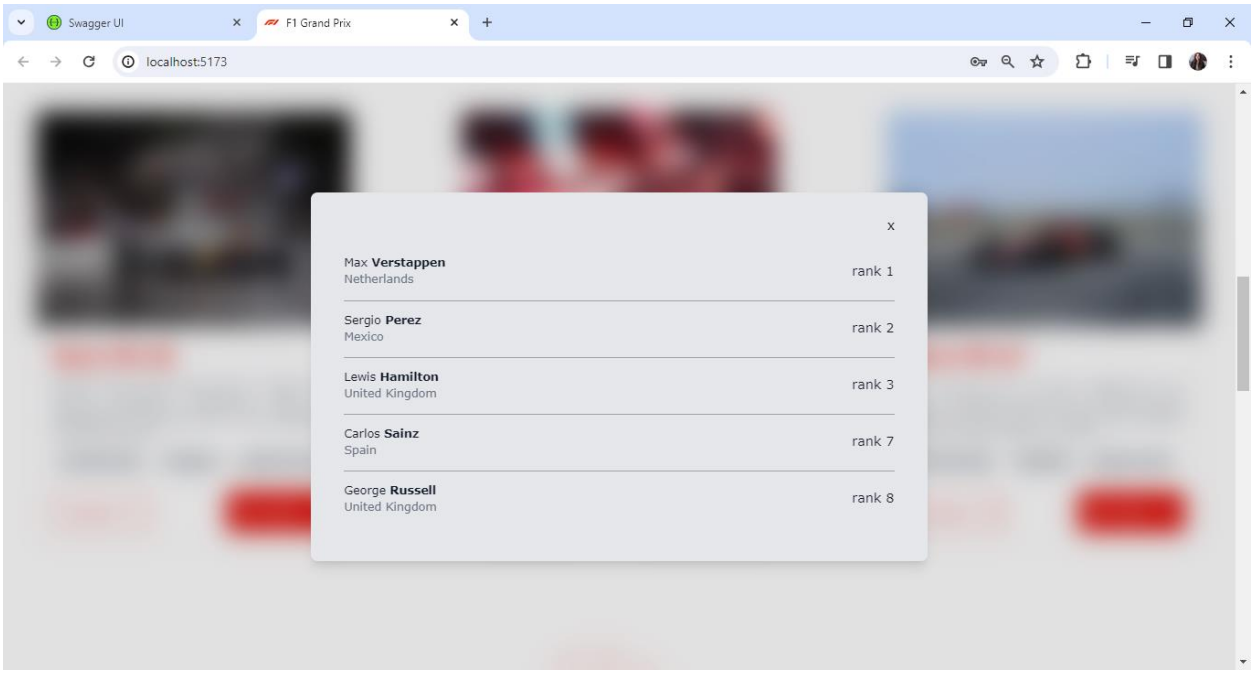
Postuslovi: Prikazana je lista učešća

Osnovni scenario:

1. Kupac klikom na dugme Racers na kartici trke poziva sistem da učita listu učešća
2. Sistem pronalazi i ispisuje listu učešća

Alternativni scenario:

2.1 Sistem ne uspeva da učita listu učešća



4. Učitavanje zona

Akter: Kupac

Preduslovi: /

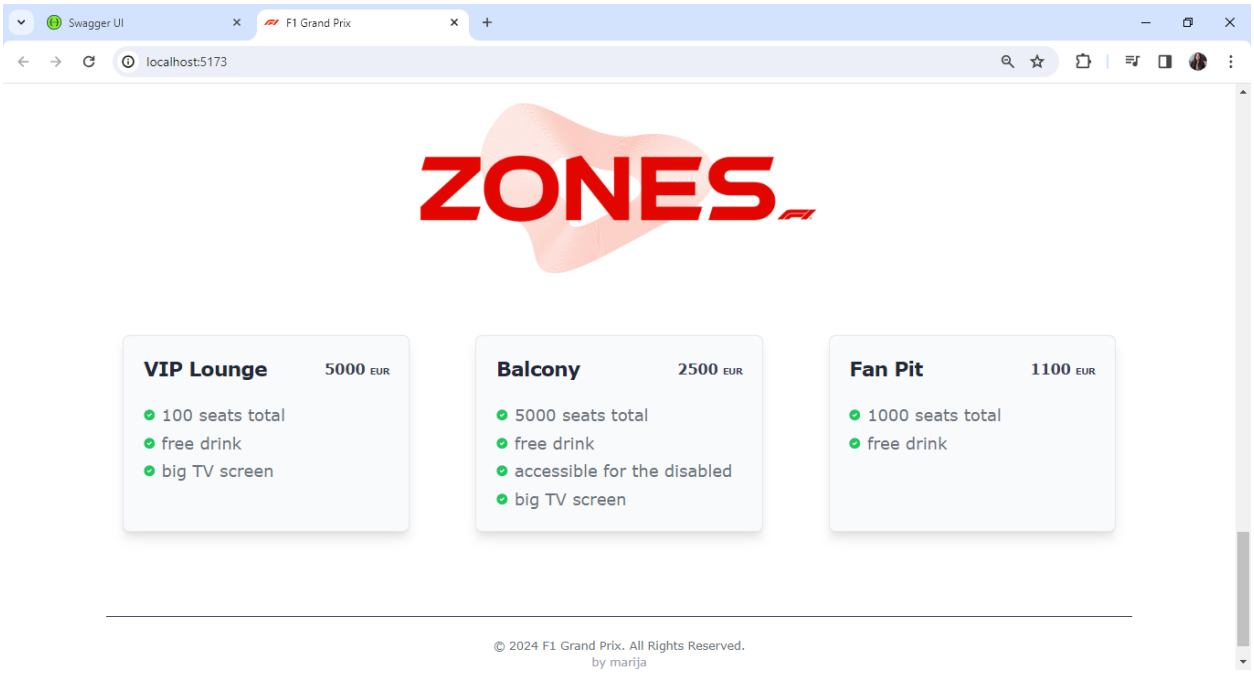
Postuslovi: Prikazana je lista zona

Osnovni scenario:

1. Kupac otvara naslovnu stranu sistema i poziva sistem da učita listu zona
2. Sistem pronalazi i ispisuje listu zona

Alternativni scenario:

2.1 Sistem ne uspeva da učita listu zona



5. Učitavanje rezervacija kupca

Akter: Kupac

Preduslovi: Kupac je ulogovan u sistem

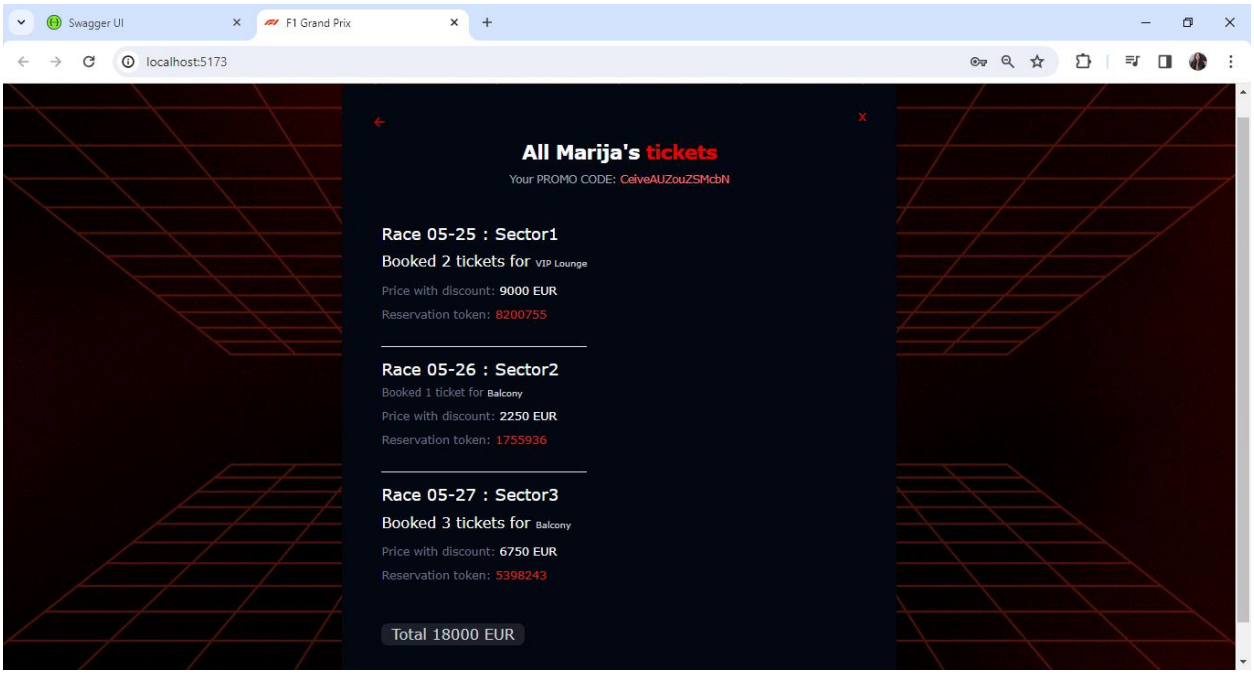
Postuslovi: Prikazana je lista rezervacija

Osnovni scenario:

1. Kupac klikom na dugme My Reservations na stranici za rezervaciju poziva sistem da učita listu korisnikovih rezervacija
2. Sistem pronalazi i ispisuje listu rezervacija kupca

Alternativni scenario:

2.1 Sistem ne uspeva da učita listu korisnikovih rezervacija



6. Login

Akter: Kupac

Preduslovi: Kupac ima kreiran nalog

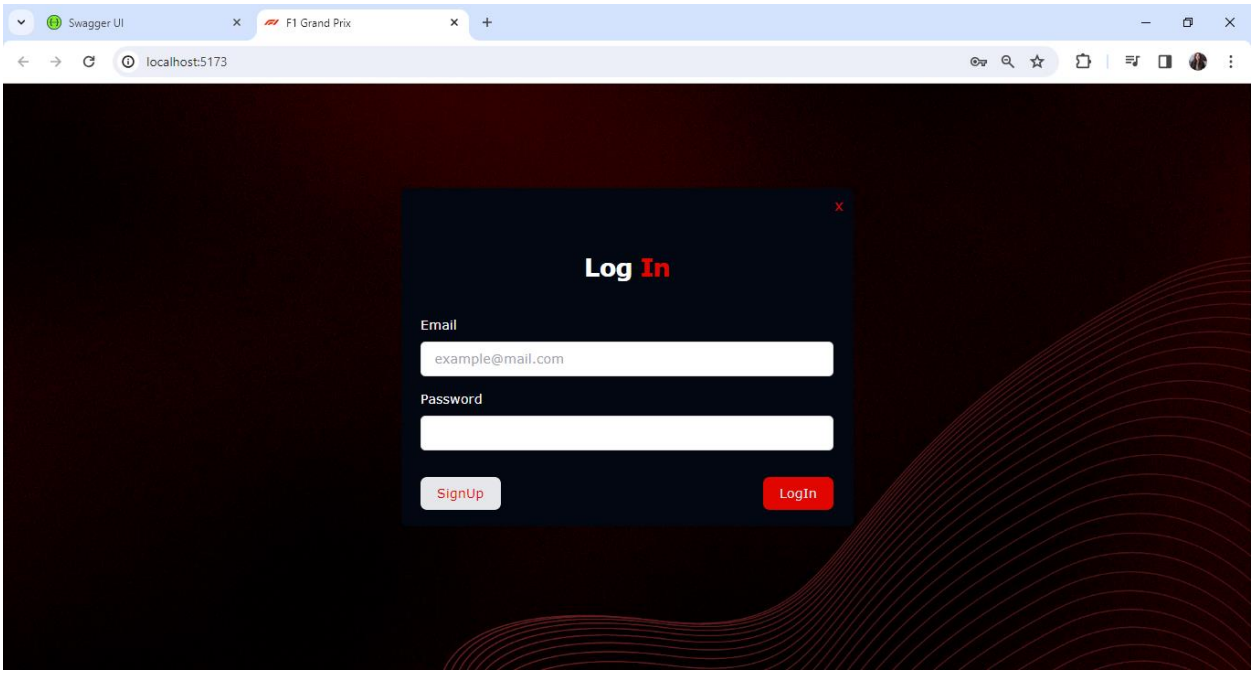
Postuslovi: Kupac je ulogovan u sistem

Osnovni scenario:

1. Kupac pokreće LogIn formu
2. Sistem prikazuje formu
3. Kupac unosi email i lozinku i pritiska dugme LogIn
4. Sistem pronalazi kupa i otvara formu za rezervaciju karata

Alternativni scenario

4.1 Sistem ne pronalazi kupca sa unetim kredencijalima



7. SignUp

Akter: Kupac

Preduslovi: Učitana je lista država

Postuslovi: Kreiran je nalog kupca

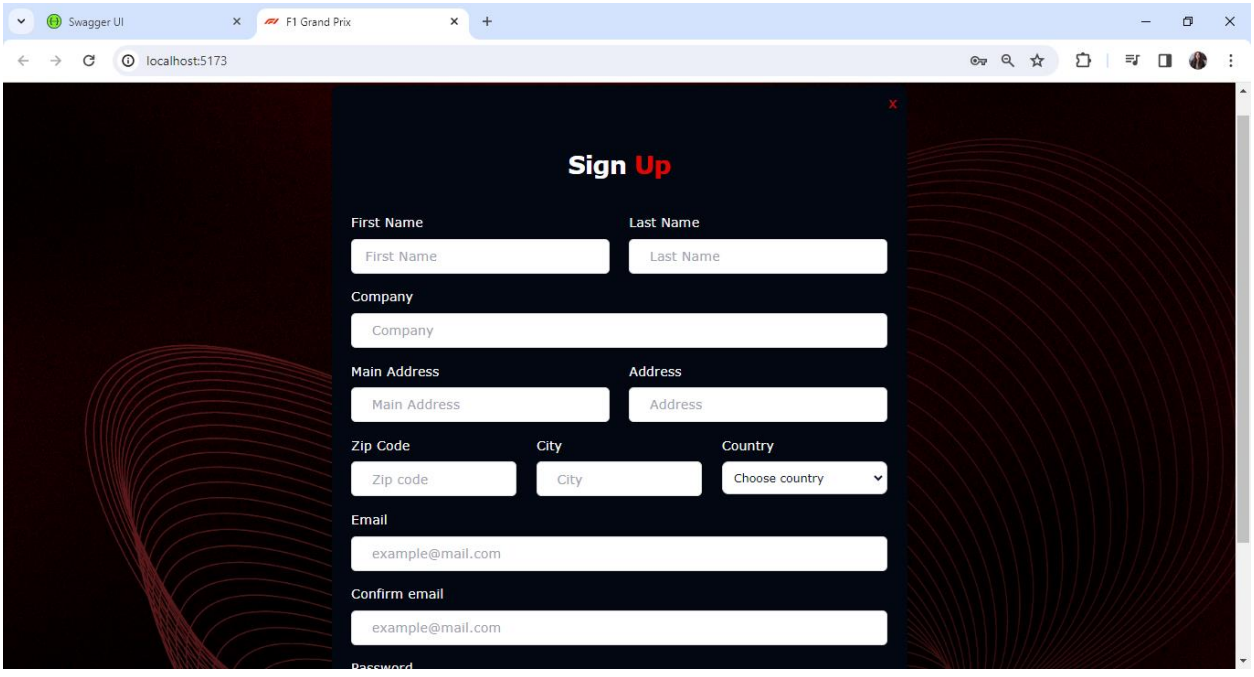
Osnovni scenario:

1. Kupac pritiska SignUp dugme na LogIn stranici
2. Sistem prikazuje formu za registraciju korisnika
3. Kupac unosi svoje podatke i pritiska dugme SignUp
4. Sistem kreira kupca i otvara formu za rezervaciju karata

Alternativni scenario:

3.1 Kupac nije pravilno uneo podatke i sistem ispisuje grešku

4.1 Kupac već postoji i sistem ispisuje grešku



8. Kreiranje rezervacije

Akter: Kupac

Preduslovi: Kupac je ulogovan u sistem, učitana je lista trka i lista zona

Postuslov: Rezervacija je kreirana

Osnovni scenario:

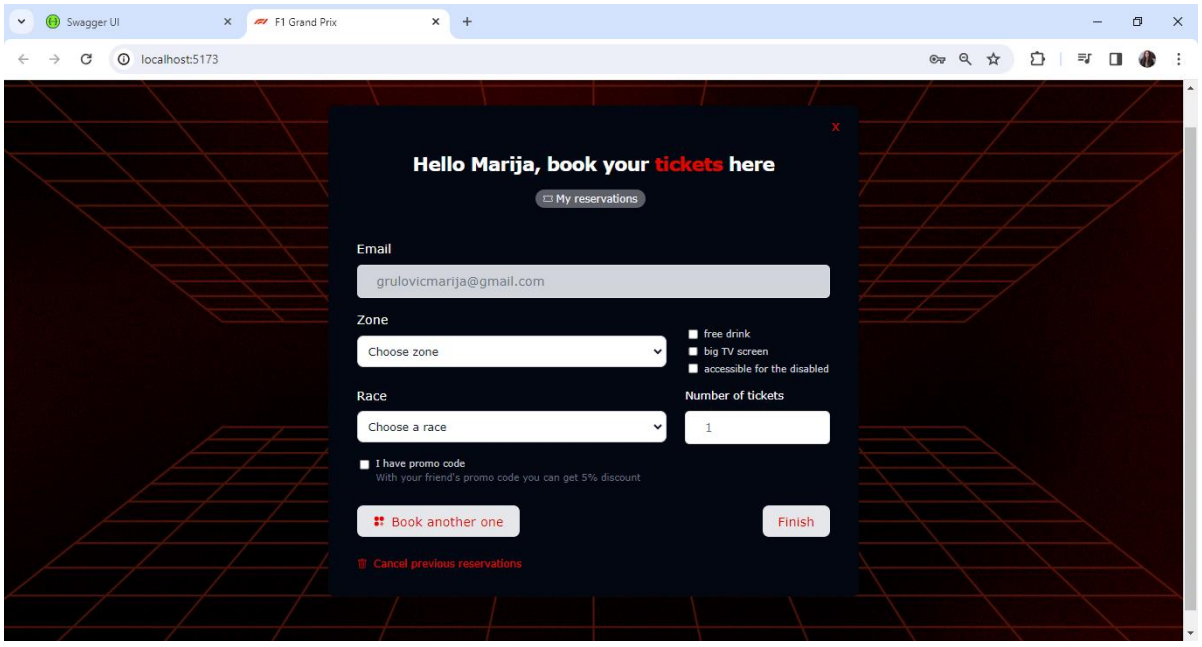
1. Kupac otvara formu za rezervaciju karata
2. Kupac bira trku i zonu i unosi broj karata
3. Ukoliko je završio sa rezervacijom, kupac pritiska dugme Finish. Ukoliko kupac želi da rezerviše kartu za još trka pritiska dugme Book another one
4. Sistem preusmerava korisnika na stranicu za potvrdu rezervacije
5. Kupac pritiska dugme Confirm
6. Sistem kreira rezervacije i prikazuje spisak rezervacija ulogovanog kupca

Alternativni scenario:

3.1 Kupac nije popunio sva polja i sistem izbacuje grešku

5.1 Kupac pritiska dugme ‘x’ i sistem ga vraća na formu za rezervaciju karata čime se prelazi na 2. Korak osnovnog scenarija

6.1 Kupac pokušava da kreira rezervaciju koja već postoji, sistem izbacuje poruku u greški



9. Brisanje rezervacije

Akter: Kupac

Preduslovi: Kupac je ulogovan u sistem, rezervacija je kreirana

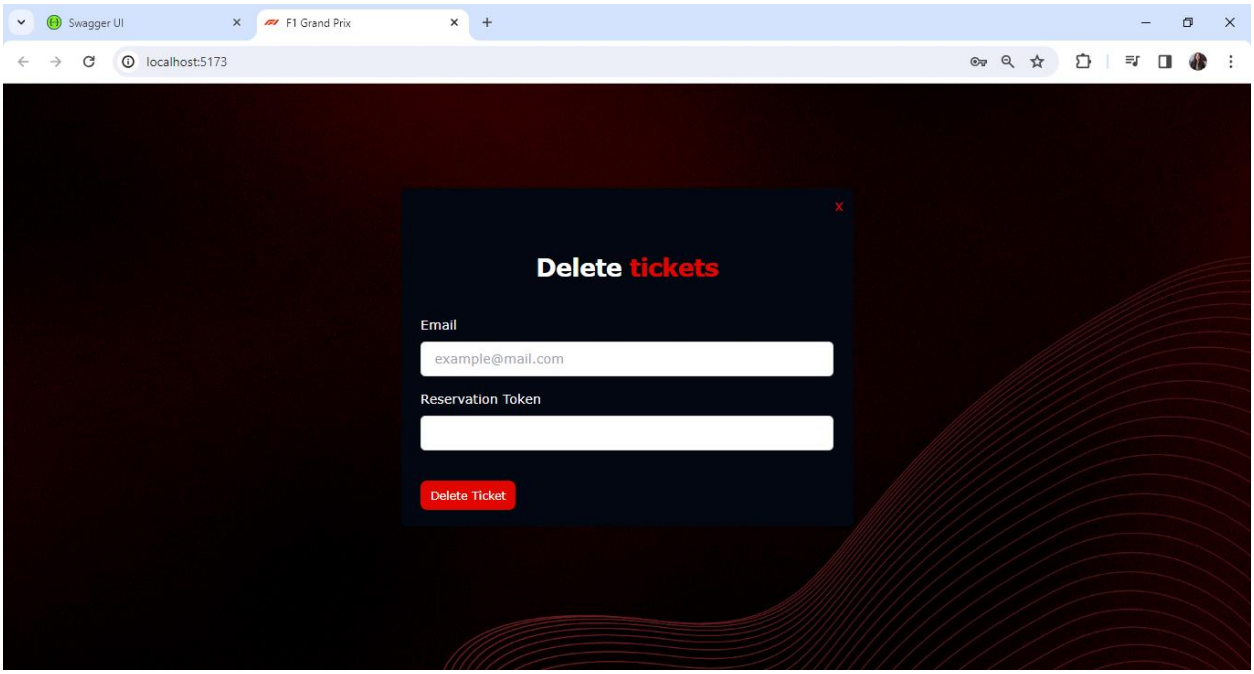
Postuslov: Rezervacija je obrisana

Osnovni scenario:

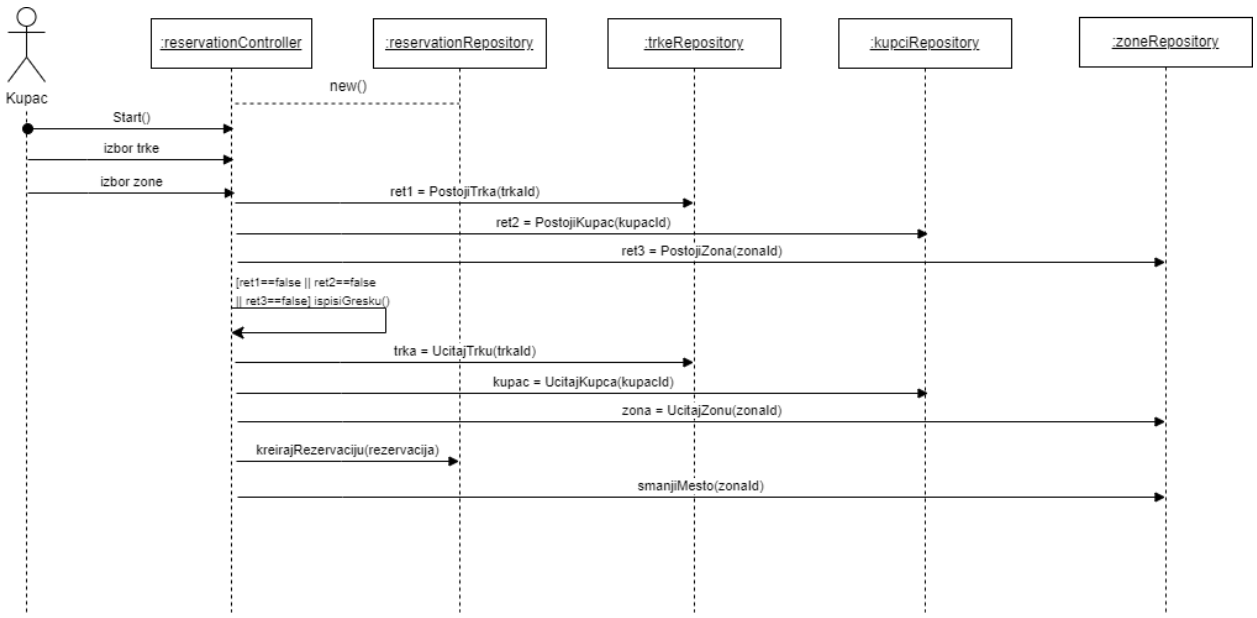
1. Kupac pritiska dugme Cancel previous reservations na formi za rezervaciju karata
2. Sistem preusmerava kupca na stranicu za brisanje rezervacija
3. Kupac unosi email i token rezervacije i pritiska dugme Delete ticket
4. Sistem pronalazi rezervaciju, briše je i otvara stranicu sa rezervacijama kupca

Alternativni scenario:

4.1 Email ili token nisu ispravni, sistem ne pronalazi rezervaciju i ispisuje poruku o greški



Rezervacija karata – Dijagram sekvenci

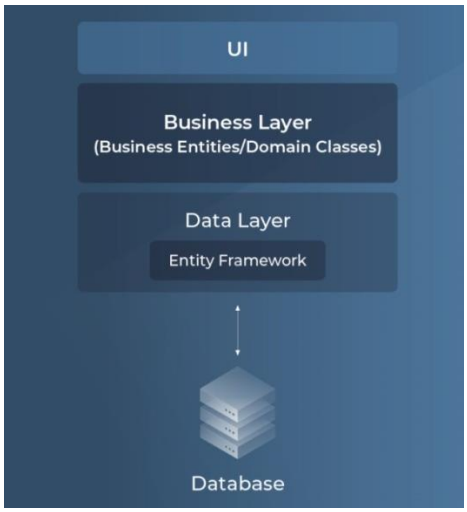


ASP .NET Web API

ASP .NET je back-end tehnologija korišćena u radu, koja predstavlja framework otvorenog koda, kreiran od strane Microsoft-a primarno za izgradnju web aplikacija i servisa. ASP .NET proširuje .NET platformu alatima i bibliotekama specifičnim za potrebe izgradnje web aplikacija.[1]

Entity Framework

Entity Framework preuzima na sebe odgovornost komunikacije sa bazom podataka koristeći jednostavnu, čitljivu i lako razumljivu sintaksu. Entity Framework transformiše naredbe objektno-orijentisanog jezika u SQL upite koji se dalje prosleđuju do baze podataka i na taj način predstavlja posrednika između .NET objekata i baze podataka.[2]



Objektno-relacioni maperi služe da, prepoznavanjem šablona, obezbede transformaciju koncepata objektno-orijentisane paradigme, kao što su klase, atributi i relacije između klasa, u odgovarajuće koncepte relacionog prostora, poput tabela, kolona i spoljnih ključeva. Na osnovu navedenih atributa klase, Entity Framework kreira kolone u relacionoj bazi podataka koja je definisana nad domenom koji je analogan tipu podatka, dok će celokupna klasa u relacionom svetu biti predstavljena kao relacija (tabela). Neka od tipičnih preslikavanja atributa u kolone su preslikavanje tipa podatka string u domen varchar(max) ili bool u bit.

[1] https://dotnet.microsoft.com/en-us/learn/aspnet/what-is-aspnet
[2] https://learn.microsoft.com/en-us/aspnet/entity-framework

Klasa DbContext predstavlja sesiju sa bazom podataka i omogućava izvršavanje osnovnih operacija za manipulaciju podacima u bazi. DbContext se obično koristi sa izvedenim tipom (u ovom radu - klasa DataContext) koji sadrži DbSet<TEntity> svojstva koja reprezentuju konekciju sa svakom tabelom u bazi. Ovi setovi se automatski inicijalizuju kada se instanca izvedene klase kreira.[3]

Language-Integrated Query (LINQ) je skup tehnologija koji omogućava integraciju i realizaciju upita direktno u C# programskom jeziku. Na ovaj način jednostvanom sintaksom objektno-orijentisanog programskog jezika možemo kreirati upite bez korišćenja jezika koji su za to specijalizovano namenjeni (kao što je SQL).

Klase koje reprezentuju repozitorijum predstavljaju izdvojeni sloj u komunikaciji sa bazom podataka, i zadužene su za izvršavanje osnovnih (CRUD) operacija nad bazom podataka. Ove klase sadrže DataContext kao svoje polje, preko kog se vrši celokupna komunikacija sa bazom.

```
namespace F1GrandPrixApi.Repository
{
    2 references
    public class RezervacijaRepository : IRezervacijaRepository
    {

        DataContext context;

        0 references
        public RezervacijaRepository(DataContext context)
        {
            this.context = context;
        }

        2 references
        public bool KreirajRezervaciju(Rezervacija rezervacija)
        {
            context.Add(rezervacija);
            return Sacuvaj();
        }

        2 references
        public bool OtkaziRezervaciju(Rezervacija rezervacija)
        {
            rezervacija.token = "x";
            rezervacija.aktivna = false;
            context.Update(rezervacija);
            return Sacuvaj();
        }

        1 reference
        public Rezervacija UcitajRezervaciju(int kupacId, int trkaId)
        {
            return context.rezervacije.Where(r => r.KupacId == kupacId && r.TrkaId == trkaId && r.aktivna == true).FirstOrDefault();
        }
```

Metoda SaveChanges() je zadužena za rad sa transakcijama. Sve promene napravljene do poziva ove metode biće smeštene u jednu transakciju, a povratna vrednost ove metode je tipa bool koja reprezentuje Commit odnosno Rollback kreiranih promena.

```
public bool Sacuvaj()
{
    int sacuvano = context.SaveChanges();
    return sacuvano > 0;
}
```

[3] https://learn.microsoft.com/en-us/dotnet/api/system.data.entity.dbcontext?view=entity-framework-6.2.0