



BamBus

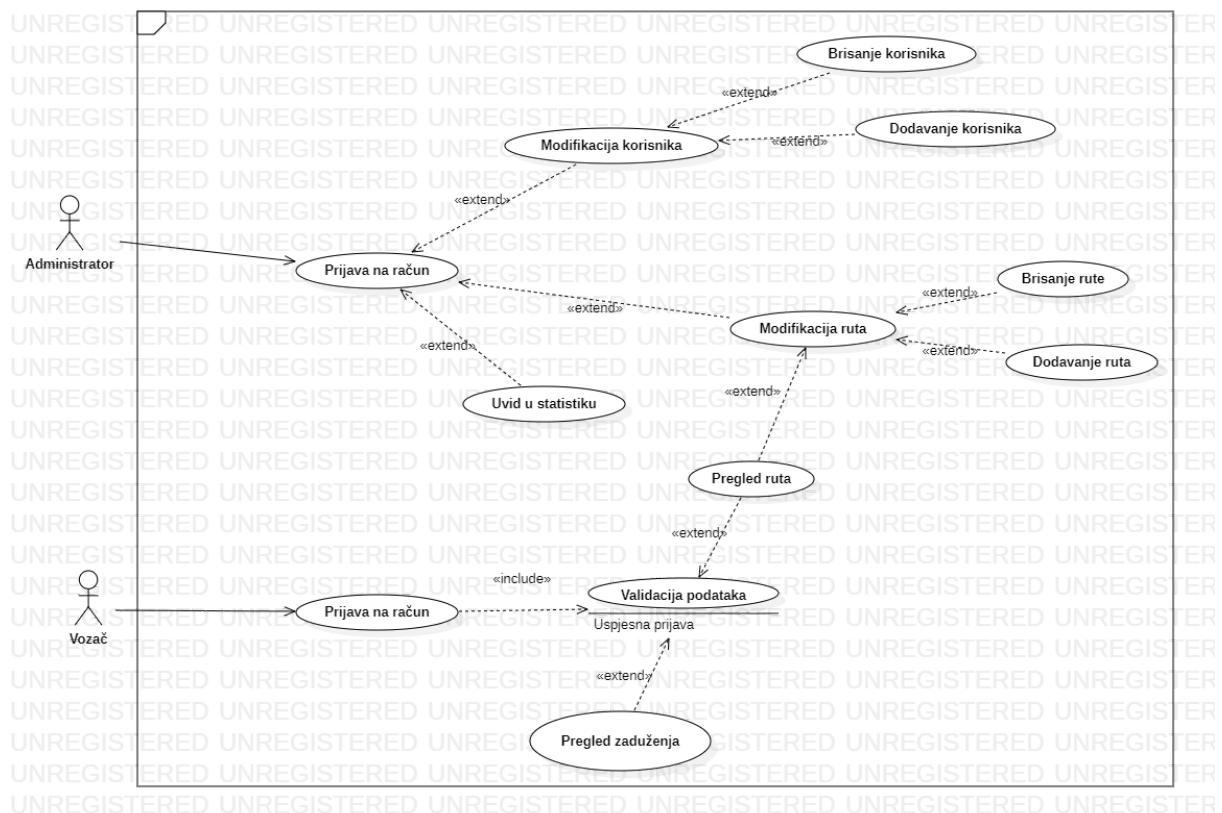
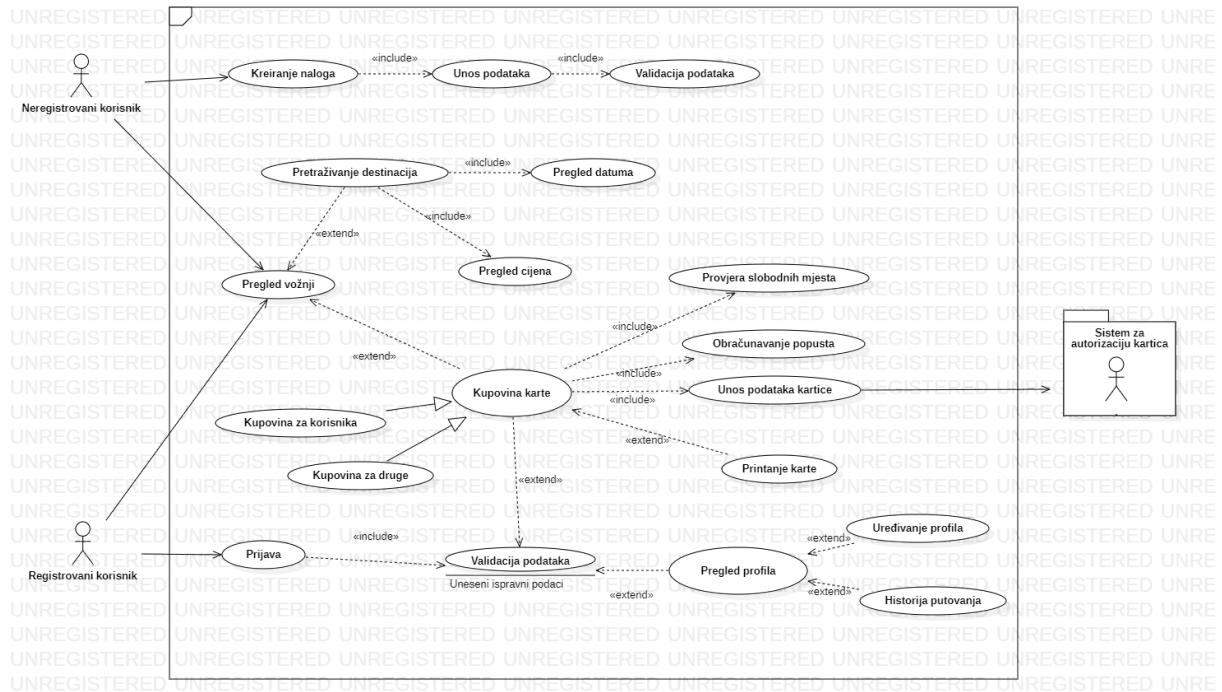
Determine your destination

Članovi tima:

- Agić Ahmed
- Begić Armin
- Briški Danijel
- Zukić Muhamed

Use case dijagram

Use case dijagram je raščlanjen na use case dijagram **Korisnika** i use case dijagram **Administratora** i **Vozača**



Scenariji

Scenarij - Pregled, dodavanje i brisanje uposlenika

Naziv:	Pregled, dodavanje i brisanje uposlenika
Opis slučaja upotrebe:	Admin ima mogućnost pregleda uposlenika i njihovih detalja (kao što su ime, prezime, plata) kao i mogućnost dodavanje novih i brisanje postojećih uposlenika
Vezani zahtjevi:	/
Preduslovi:	Osoba mora biti registrovana kao <u>admin</u>
Posljedica:	U sistemu su uneseni uposlenici sa svim njihovim vezanim detaljima.
Primarni akteri:	Admin (svaka osoba registravana kao admin)
Ostali akteri:	/
Glavni tok:	Administratoru se nakon log in-a otvara forma sa spiskom trenutnih zaposlenika kao i opcija da izmjeni postojeće (uključujući njihovo brisanje) kao i dodavanje novih.

Tok događaja

- Admin pristupa uposlenicima i modificira podatke o njima

Admin	Sistem
1. Administrator se prijavljuje unoseći pristupne podatke	2. Sistem vrši validaciju unesenih podataka
3. Administrator pristupa spisku uposlenika	4. Sistem učitava uposlenike iz baze podataka i prikazuje ih administratoru
5. Administrator odabire jednog korisnika	6. Administratoru se prikazuju detalji o uposleniku kao što su: ime, prezime, plata
7. Administrator u formu upisuje novo ime i/ili novo prezime i/ili novu platu i pritišće dugme za spašavanje promjena	8. Promjene se sačuvaju i nakon toga vrši se update baze podataka

Alternativni tok događaja br. 1

- U koraku 2 admin nije unio validne podatke**

Admin	Sistem
	1. Uneseni su nevalidni podaci i sistem terminira trenutnu aktivnost

Alternativni tok događaja br. 2

- U koraku 7 admin pristupa uposleniku i briše uposlenika**

Admin	Sistem
1. Administrator odabire opciju brisanja trenutno odabranog uposlenika	2. Sistem briše uposlenika iz baze podataka i korisnika vraća na listu svih uposlenika

Alternativni tok događaja br. 3

- U koraku 5 admin pristupa listi uposlenika i dodaje novog uposlenika

Admin	Sistem
1. Administrator odabire opciju dodavanja novog uposlenika	2. Sistem otvara formu za unos podataka o novom uposleniku
3. Administrator upisuje podatke i odabire opciju sačuvanja promjena	4. Vrši se verifikacija novog uposlenika i on se unosi u bazu podataka

Alternativni tok događaja br. 4

- U koraku 4 Alternativnog događaja br. 3 verifikacija nije uspješna.

Admin	Sistem
	1. Verifikacija nije uspješna pa sistem ne unosi novog uposlenika i terminira trenutnu aktivnost.

Scenarij: Modifikacija ruta

Opis scenarija:

Naziv:	Modifikacija, brisanje i dodavanje ruta
Opis:	Administrator sistema ima mogućnost pregleda i modifikacije ruta, što uključuje izmjenu i brisanje već postojećih, te dodavanje novih ruta
Vezani zahtjevi:	/
Preduvjeti:	Administrator mora biti prijavljen
Posljedice – uspješan završetak:	Izmijenjena/dodana/obrisana određena ruta
Posljedice – neuspješan završetak	Nema izmjena u sistemu, tj nisu modifikovane rute
Primarni akteri	Administrator
Ostali akteri	/
Glavni tok	Administrator pristupa sistemu i po potrebi ga modificira, tj. edituje postojeće rute (datum, vrijeme, put), dodaje nove ili briše stare rute.

Tok dogadjaja:

Administrator	Sistem
1. Pristup sistemu/prijava administratora	2. Validacija unesnih podataka
	3. Otvaranje interfejsa za administratora
4. Izbor prozora za modifikaciju ruta	5. Prikaz opcija (editovanje, brisanje, dodavanje ruta)
6. Izbor opcije editovanja ruta	7. Prikaz svih ruta
8. Pretraga i izbor određene rute	
9. Izmjena sadržaja rute (datum, vrijeme, kapacitet, cijena)	
10. Potvrđivanje izmjena	11. Spašavanje izmjena

Alternativni tok 1: Dodavanje ruta

Preduvjeti: u koraku 6 glavnog toka izabrana opcija dodavanje ruta

Administrator	Sistem
	1. Prikaz prozora za kreiranje nove rute
2. Unos svih podataka nove rute (datum, vrijeme, polazište, odredište, kapacitet, cijena)	
3. Validacija unesnih podataka	
4. Nastavak na korak 10 glavnog toka događaja	

Alternativni tok 2: Brisanje ruta

Preduvjeti: u koraku 6 glavnog toka izabrana opcija brisanje ruta

Administrator	Sistem
	1. Prikaz svih ruta
2. Pretraga i izbor određene rute	
3. Brisanje rute	
4. Nastavak na korak 10 glavnog toka događaja	

Alternativni tok 3: Neuspješna prijava

Preduvjeti: u koraku 2 glavnog toka validacija podataka nije bila uspješna

Posljedice: prijavljuje se greška

Administrator	Sistem
	1. Prijavljivanje greške
	2. Prekid trenutne aktivnosti

Opis scenarija

Naziv slučaja upotrebe:	Pregled zaduženja i odabir rute
Opis slučaja potrebe:	Vozač vrši pregled zaduženja za naredni period, te za specifičan tj. odabrani put, dobija interaktivnu mapu na kojoj bira rutu.
Vezani zahtjevi:	/
Preduslovi:	Mora biti registrovan/ulogovan
Posljedica:	Vozač posjeduje za željeno putovanje vlastitu rutu koju može koristiti za sljedeća putovanja.
Primarni akteri:	Vozač
Ostali akteri:	/
Glavni tok:	Vozaču se nakon registracije otvara pregled njegovih zaduženja u narednom periodu. Odabirom želenog puta dobija uvid u interaktivnu mapu za odabir rute i detalja puta.

Tok događaja

Google maps	Korisnik	Odgovor sistema
	1. Vozač se loguje	2. Sistem validira informacije o računu
	3. Vozač bira jedno od putovanja koja su ponuđena u narednom periodu	
4. Učitavaje mape	5. Na mapi vozač bira željenu rutu	6. Sistem ažurira rutu

Alternativni tok događaja

Google maps	Korisnik	Odgovor sistema
	1. Vozač se loguje	2. Sistem validira informacije o računu
		3. Sistem vraća poruku o neuspješnoj registraciji

Scenarij: Kupovina karte

Naziv:	Prijava na account i kupovina/narudžba karte
Opis:	Korisnik koji posjeduje account kupuje kartu
Vezani zahtjevi:	/
Preduvjeti:	Korisnik posjeduje account i bankovni račun
Posljedice - uspješan završetak:	Korisnik je kupio kartu
Posljedice - neuspješan završetak:	Korisnik nije kupio kartu
Primarni akteri:	Korisnik
Ostali akteri:	/
Glavni tok:	Korisnik koristi svoj account da bi mogao kupiti kartu za određenu destinaciju/destinacije

Tok događaja:

<u>Korisnik</u>	<u>Aplikacija</u>	<u>Sistem za kartično plaćanje</u>
1.Unos podataka za prijavu	2.Verifikacija account-a	
	3.Prikaz predloženih destinacija i prethodno ako postoje odabранe rute	
4.Odabir neke od ponuđenih ruta ili pretraga specifične rute	5.Prikaz željene destinacije sa informacijama	
5.Potvrda klikom na kupovinu		
		6.Obavljanje transakcije
	7.Ažuriranje podataka u sistemu	
	8.Vraćanje korisnika na predložene destinacije i	

	mogućnost pretrage	
--	--------------------	--

-> Alternativni tok događaja-kupovina karte sa popustom:

<u>Korisnik</u>	<u>Aplikacija</u>	<u>Sistem za kartično plaćanje</u>
1.Unos podataka za prijavu	2.Verifikacija account-a	
	3.Prikaz predloženih destinacija i prethodno ako postoje odabранe rute	
4.Odabir neke od ponuđenih ruta ili pretraga specifične rute	5.Prikaz željene destinacije sa informacijama	
5.Potvrda klikom na kupovinu		
	6.Obračun popusta na VIP korisnike ili slično	
		7.Obavljanje transakcije
	8.Ažuriranje podataka u sistemu	
	9.Vraćanje korisnika na predložene destinacije i mogućnost pretrage	

-> Alternativni tok događaja-željene destinacije nema

<u>Korisnik</u>	<u>Aplikacija</u>	<u>Sistem za kartično plaćanje</u>
-----------------	-------------------	------------------------------------

1.Unos podataka za prijavu	2.Verifikacija account-a	
	3.Prikaz predloženih destinacija i prethodno ako postoje odabранe rute	
4.Odabir neke od ponuđenih ruta ili pretraga specifične rute		
	5.Željene destinacije ili termina putovanja nema	
	6.Odjava sa account-a	

-> Alternativni tok događaja-kupovina karte sa popustom ili bez,ali kupovina neuspješna

<u>Korisnik</u>	<u>Aplikacija</u>	<u>Sistem za kartično plaćanje</u>
1.Unos podataka za prijavu	2.Verifikacija account-a	
	3.Prikaz predloženih destinacija i prethodno ako postoje odabранe rute	
4.Odabir neke od ponuđenih ruta ili pretraga specifične rute	5.Prikaz željene destinacije sa informacijama	
5.Potvrda klikom na kupovinu		
	6.Obračun popusta na VIP korisnike ili slično	
		7.Obavljanje transakcije
	8.Poruka o neuspjeloj transakciji	
	9.Vraćanje korisnika na predložene destinacije i mogućnost pretrage	

Scenarij: Otkazivanje karte

Naziv:	Prijava na account i otkazivanje karte
Opis:	Korisnik koji posjeduje account otkazuje kartu
Vezani zahtjevi:	/
Preduvjeti:	Korisnik posjeduje account i bankovni račun
Posljedice - uspješan završetak:	Korisnik je otkazao putovanje
Posljedice - neuspješan završetak:	/
Primarni akteri:	Korisnik
Ostali akteri:	/
Glavni tok:	Korisnik koristi svoj account da bi mogao otkazati kartu za određenu destinaciju/destinacije

Tok događaja:

<u>Korisnik</u>	<u>Aplikacija</u>	<u>Sistem za kartično plaćanje</u>
1.Unos podataka za prijavu	2.Verifikacija account-a	
	3.Prikaz predloženih destinacija i prethodno ako postoje odabранe rute	
4.Pronalazak rute za otkazivanje	5.Prikaz željene destinacije za otkazivanje sa informacijama	
5.Potvrda klikom na otkazivanje		
	6.Obračunavanje ako je bilo popusta	
		6.Obavljanje povrata novca
	7.Ažuriranje podataka u	

	sistemu	
	8.Vraćanje korisnika na predložene destinacije i mogućnost pretrage	

-> Alternativni tok događaja-korisnik se predomislio i ne želi otkazati putovanje

<u>Korisnik</u>	<u>Aplikacija</u>	<u>Sistem za kartično plaćanje</u>
1.Unos podataka za prijavu	2.Verifikacija account-a	
	3.Prikaz predloženih destinacija i prethodno ako postoje odabранe rute	
4.Pronalazak rute za otkazivanje		
	5.Odjava sa account-a	

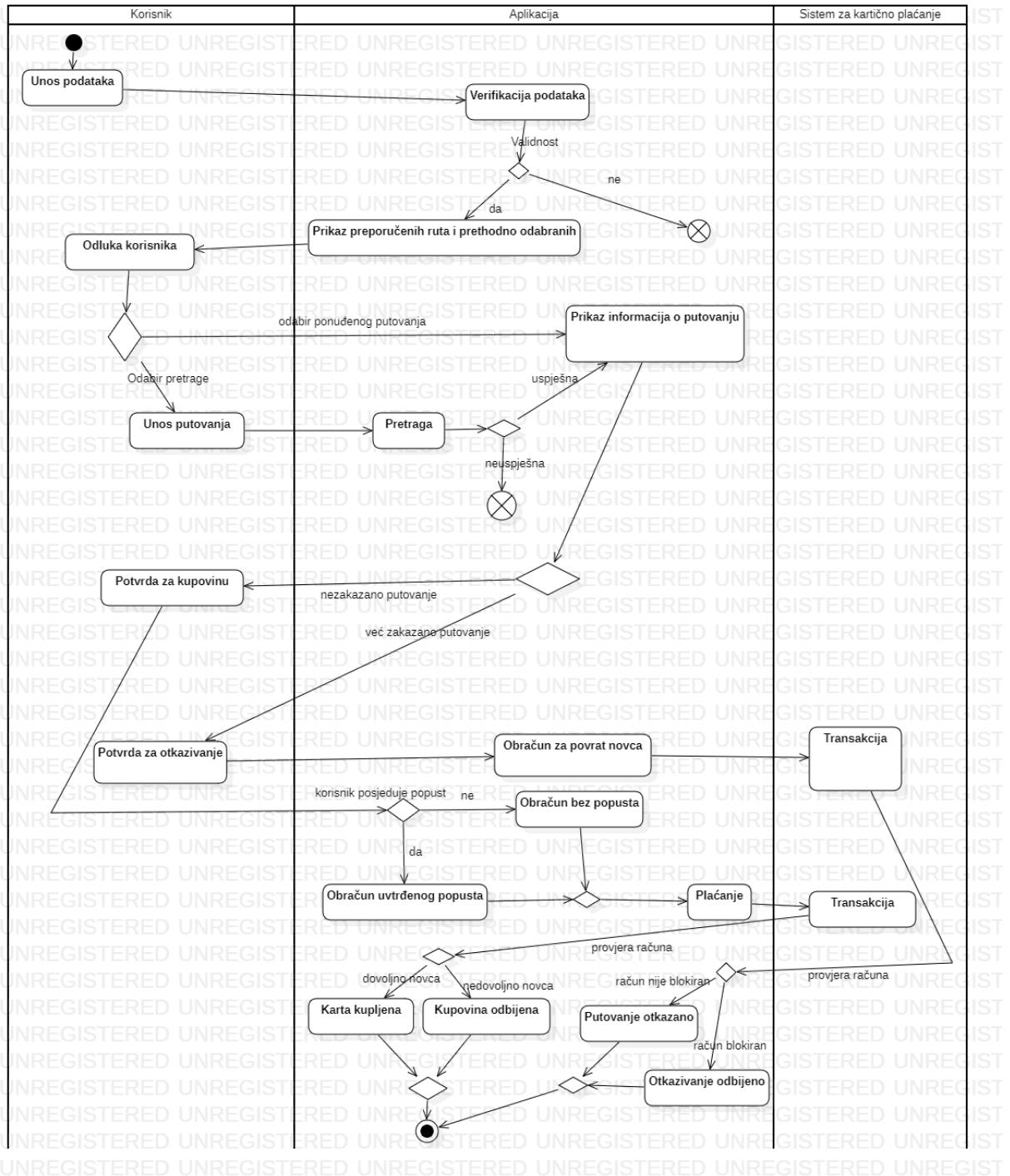
-> Alternativni tok događaja-otkazivanje karte sa popustom ili bez,ali neuspjela transakcija(račun blokiran ili slično)

<u>Korisnik</u>	<u>Aplikacija</u>	<u>Sistem za kartično plaćanje</u>
1.Unos podataka za prijavu	2.Verifikacija account-a	
	3.Prikaz predloženih destinacija i prethodno ako postoje odabранe rute	

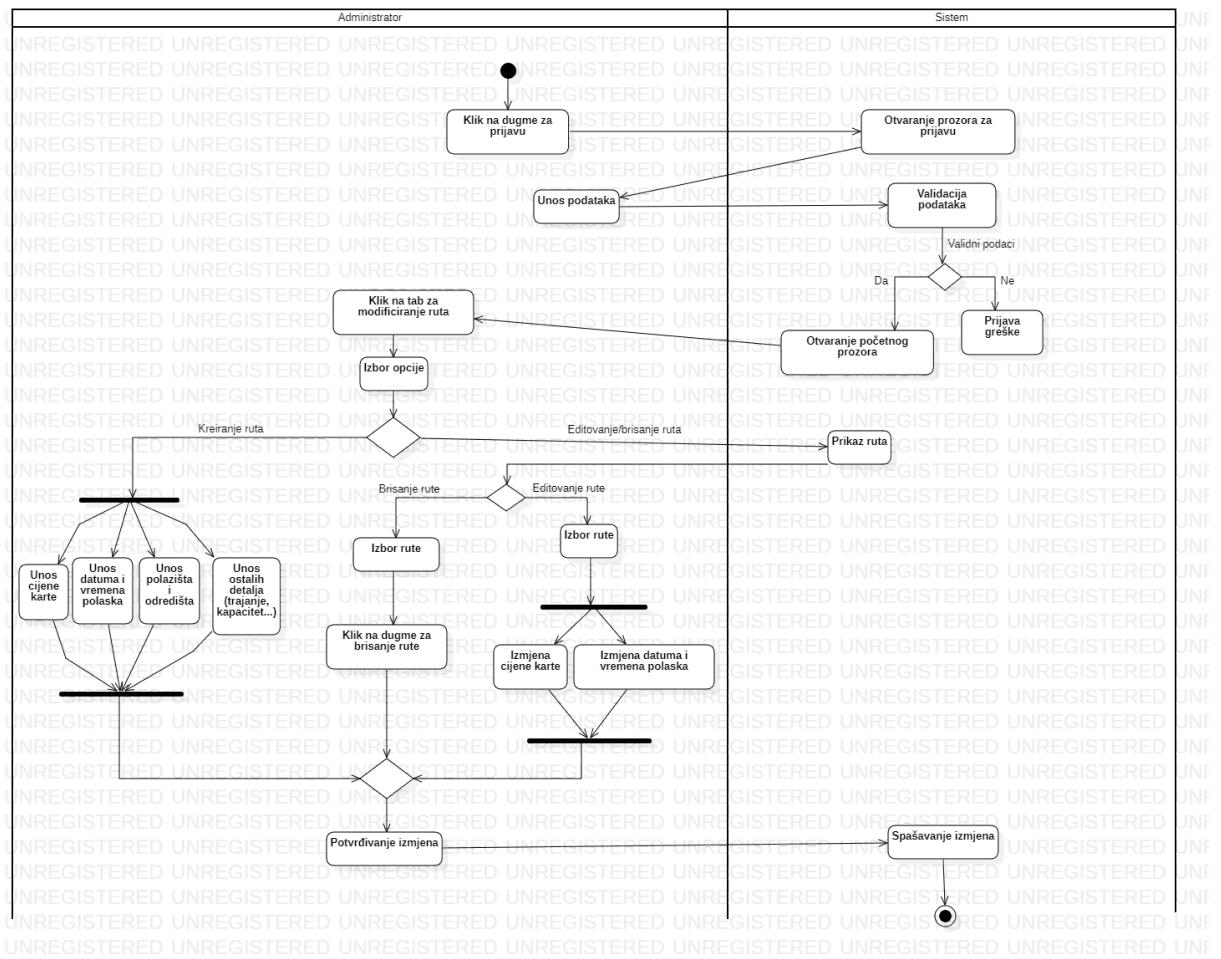
4.Odabir neke od ponuđenih ruta ili pretraga specifične rute	5.Prikaz željene destinacije sa informacijama	
5.Potvrda klikom na kupovinu		
	6.Obračun popusta na VIP korisnike ili slično	
		7.Obavljanje transakcije
	8.Poruka o neizvršenoj transakciji	
	9.Vraćanje korisnika na predložene destinacije i mogućnost pretrage	

Dijagrami aktivnosti

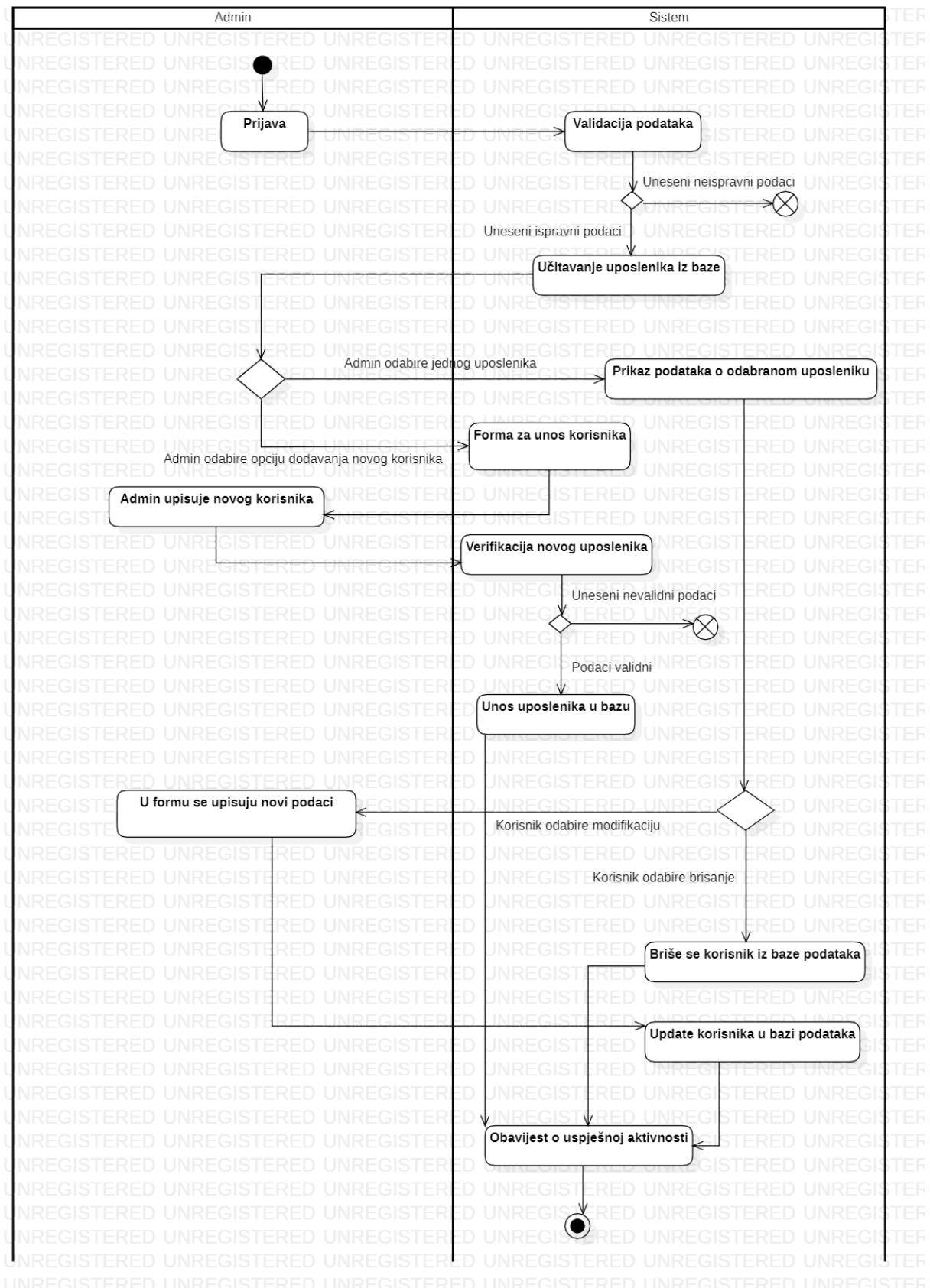
Dijagram kupovine i otkazivanja karte



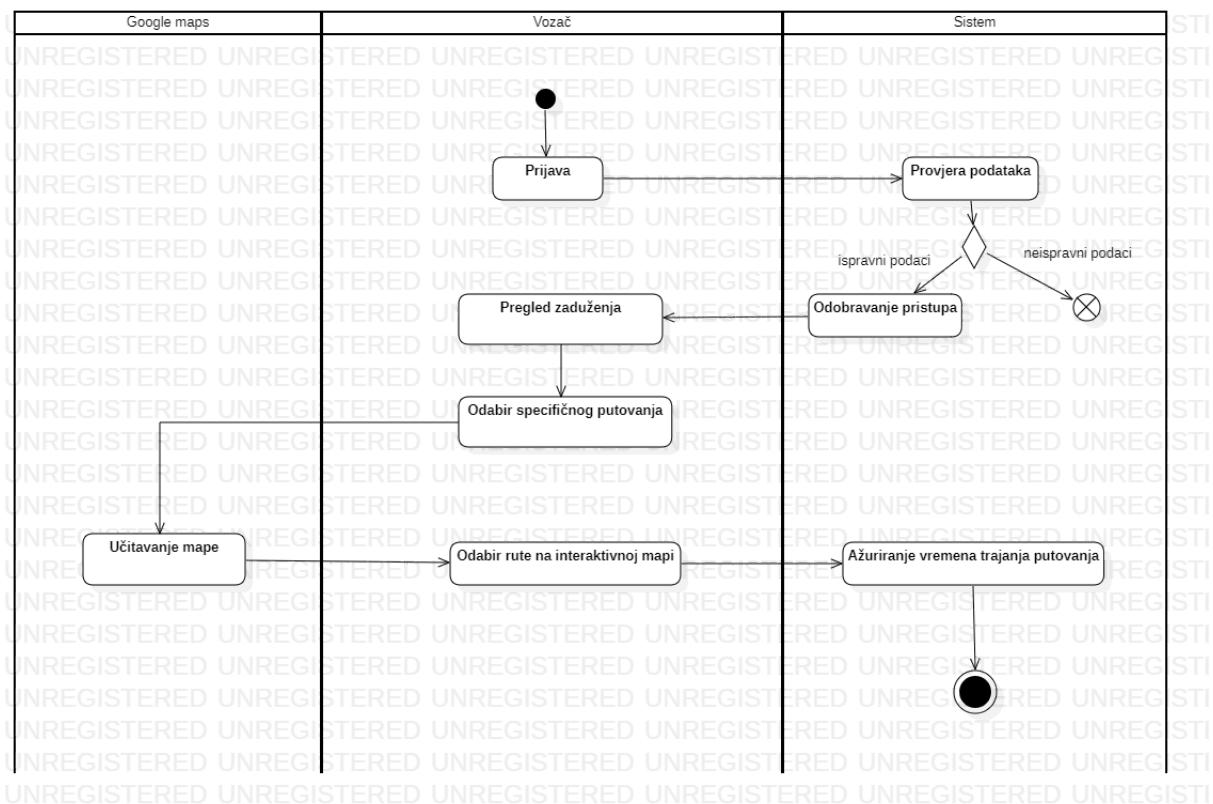
Dijagram modifikacije rute



Dijagram modifikacije uposlenika

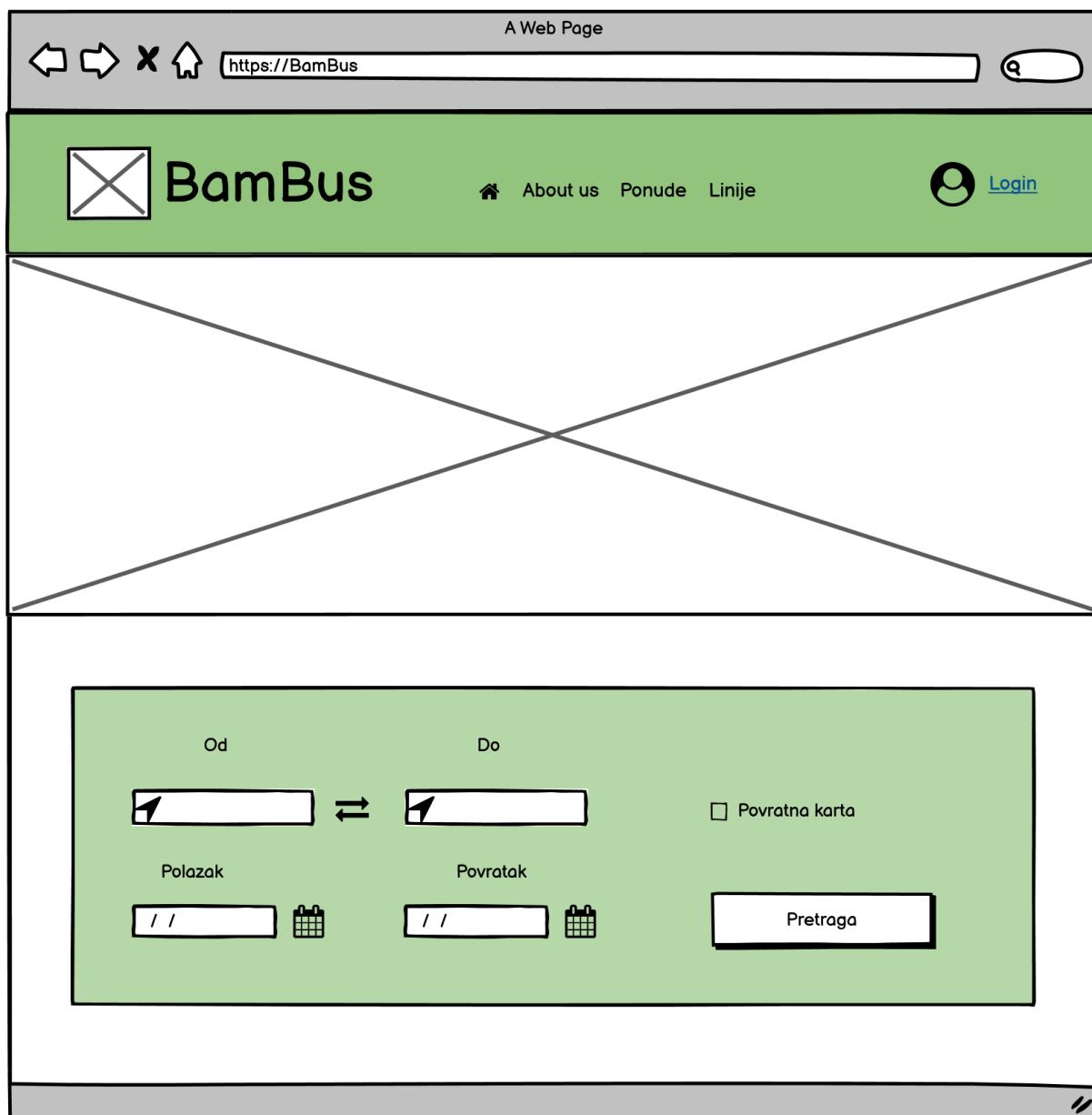


Dijagram odabira rute na mapi



Prototipi korisničkih interfejsa

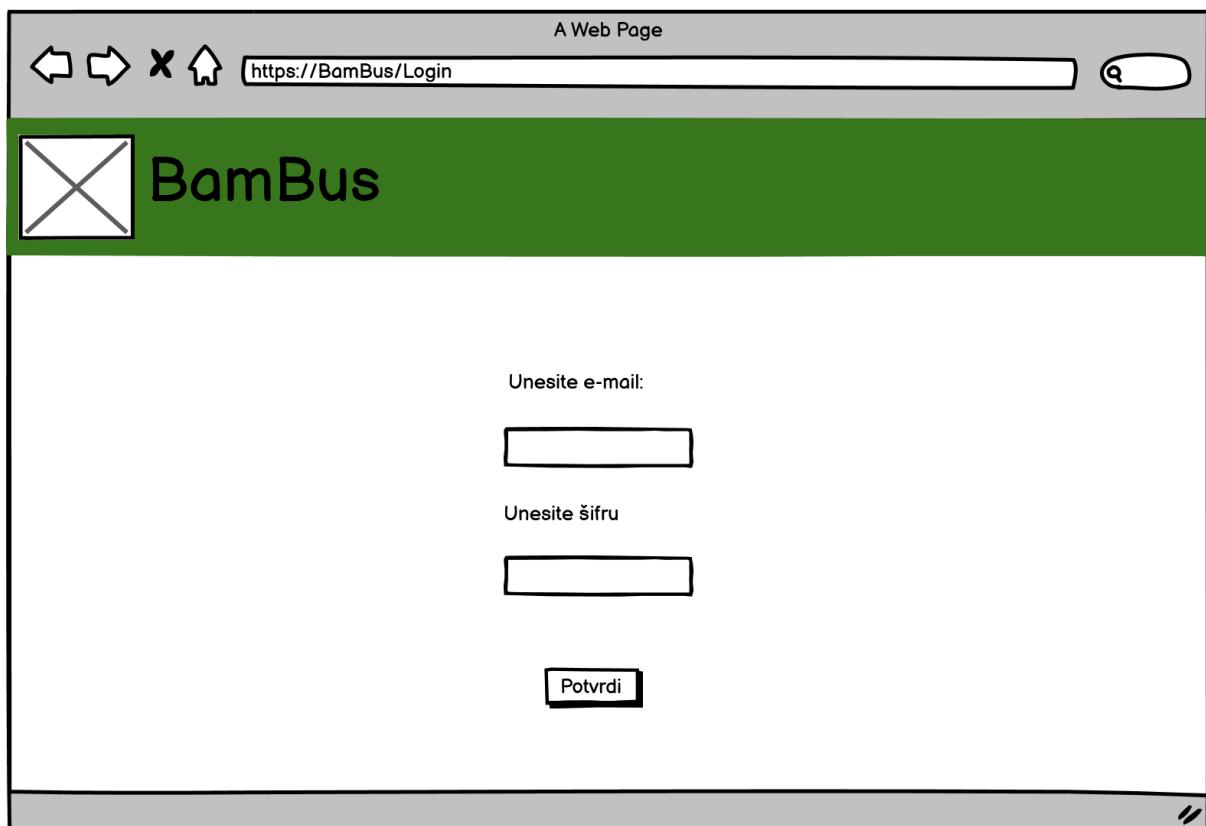
Prototip naslovne stranice



Prototip log-in stranice

A Web Page

https://BamBus/Login



BamBus

Unesite e-mail:

Unesite šifru

Potvrdi

The image shows a wireframe prototype of a web browser window titled "A Web Page". The address bar contains the URL "https://BamBus/Login". The main content area features a dark green header with the "BamBus" logo, which consists of a white square with a black 'X' inside. Below the header, there is a white form area. The first field is labeled "Unesite e-mail:" and has a rectangular input box. The second field is labeled "Unesite šifru" and also has a rectangular input box. Below these fields is a small rectangular button labeled "Potvrdi". At the bottom right corner of the form area, there is a small icon consisting of two diagonal lines.

Prototip kreiranja vožnje

Brisanje/Editovanje Dodavanje

?

Polazište: Izaberi... ▾

Veze: Izaberi... ▾
 Izaberi... ▾
 Izaberi... ▾

Add

Odredište: Izaberi... ▾

Ukupna cijena: xy KM (zavisi od veza, tj. rute)

Kapacitet: ↕

Datum: specificiraj datum
 specificiraj dane

Vrijeme polaska: ...:...

Add

Prototip modifikacije vožnji

Brisanje/Editovanje		Dodavanje	
Polazište - Odredište	Datum	Vrijeme	Detalji
			<input type="button" value="Edituj"/> <input type="button" value="Obriši"/>
Polazište - Odredište	Datum	Vrijeme	Detalji
			<input type="button" value="Edituj"/> <input type="button" value="Obriši"/>
Polazište - Odredište	Datum	Vrijeme	Detalji
			<input type="button" value="Edituj"/> <input type="button" value="Obriši"/>
Polazište - Odredište	Datum	Vrijeme	Detalji
			<input type="button" value="Edituj"/> <input type="button" value="Obriši"/>

Filtriranje

Od Do

Stajalište

Datum

Cijena

Prototip modifikacije uposlenika

Početna uposlenici Admin

Ime i prezime

Godište: XX

Modifikuj

Slika

Uposlenici: Pretraži uposlenika

Dodaj novog uposlenika

APRIL 2021

S	M	T	W	T	F	S
28	29	30	31	1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	1
2	3	4	5	6	7	8

Prototip zaslona nakon pretrage korisnika

A Web Page
https://BamBus/Login/Korisnik

BamBus

Ime i prezime

Trenutno najpopularnije destinacije

Putovanje	Polazak	Slobodnih mje	Povratna	Cijena (KM)
Sarajevo - Salzburg	12.5.2021 8:00	4	<input type="checkbox"/>	100
Klagenfurt - Madrid	6.6.2021 10:00	10	<input checked="" type="checkbox"/>	150

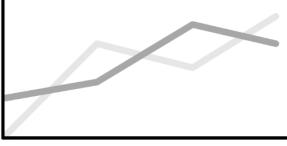
Pretraga prema mjestima: Period:

Pretraga

Vaša putovanja

Putovanje	Polazak	Cijena	Povratna	Otkazivanje
Sarajevo - Barcelona	12.5.2021 8:00	250	<input type="checkbox"/>	<input type="button" value="Kupi"/>
Klagenfurt - Beč	6.6.2021 10:00	260	<input checked="" type="checkbox"/>	<input type="button" value="Otkazi"/>

Vaša aktivnost putovanja

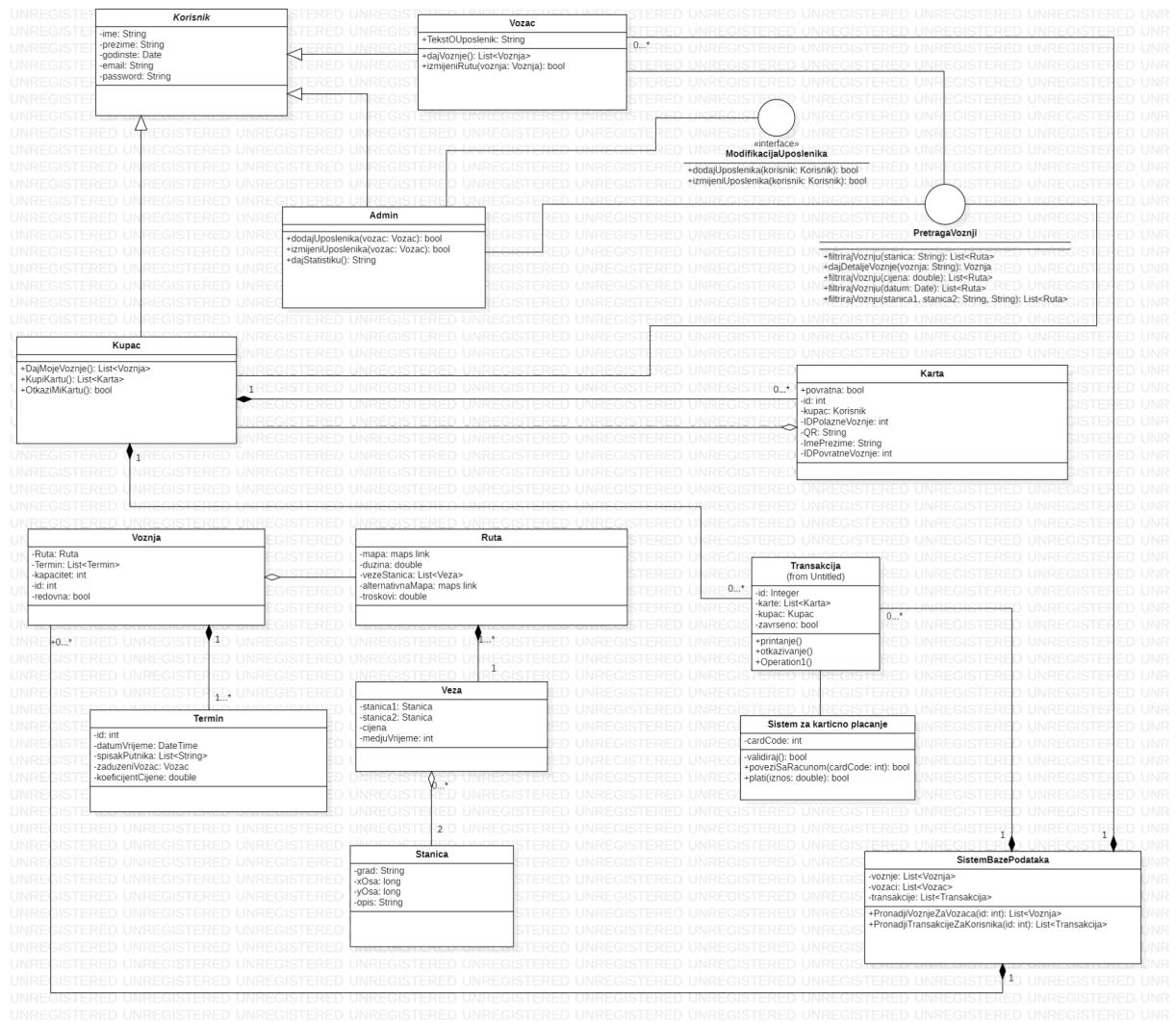


Kupovina karte

Do li ste sigurni da želite kupiti X - Y za period Z-K?

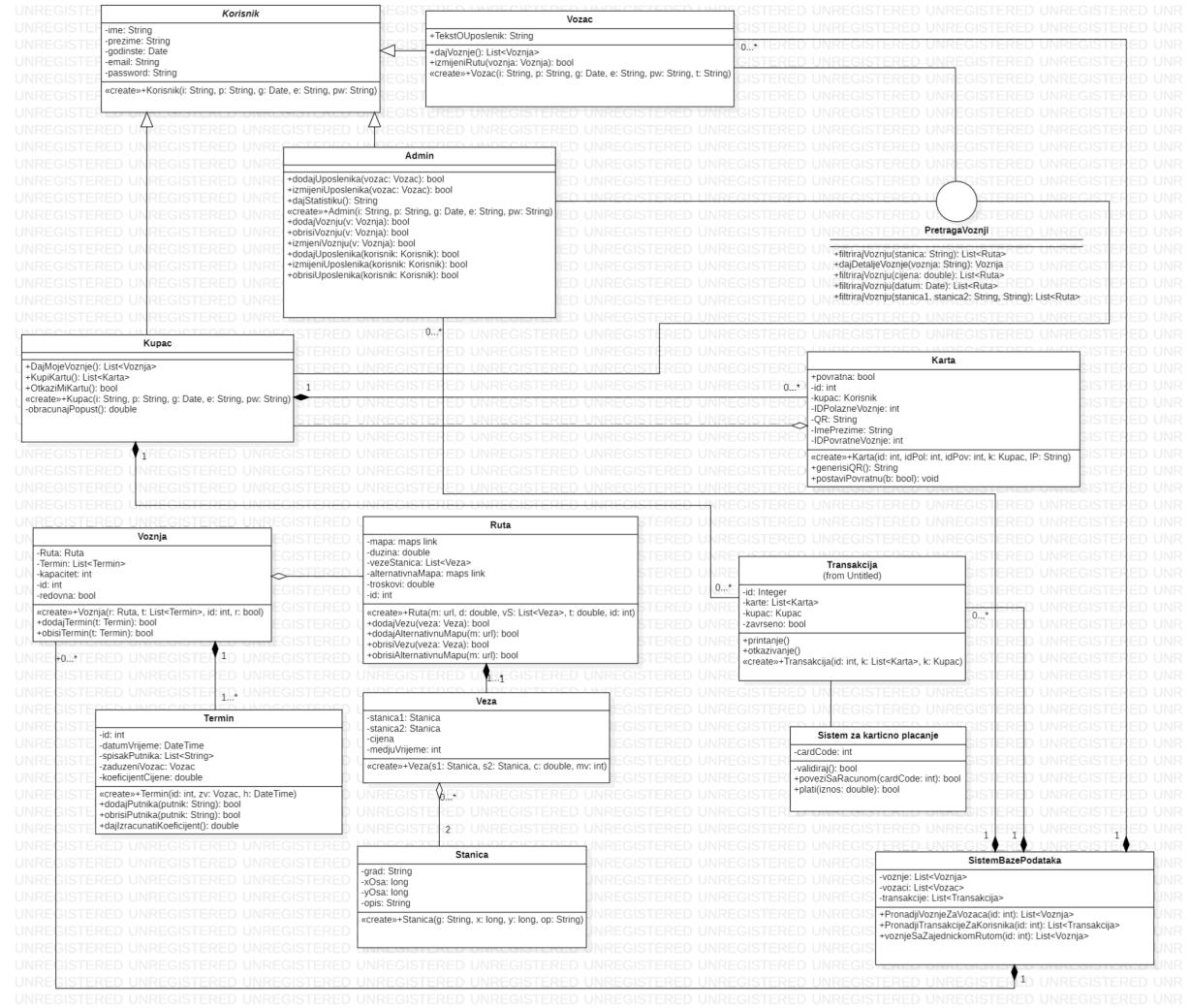
Da Ne

Prvi dijagram klasa



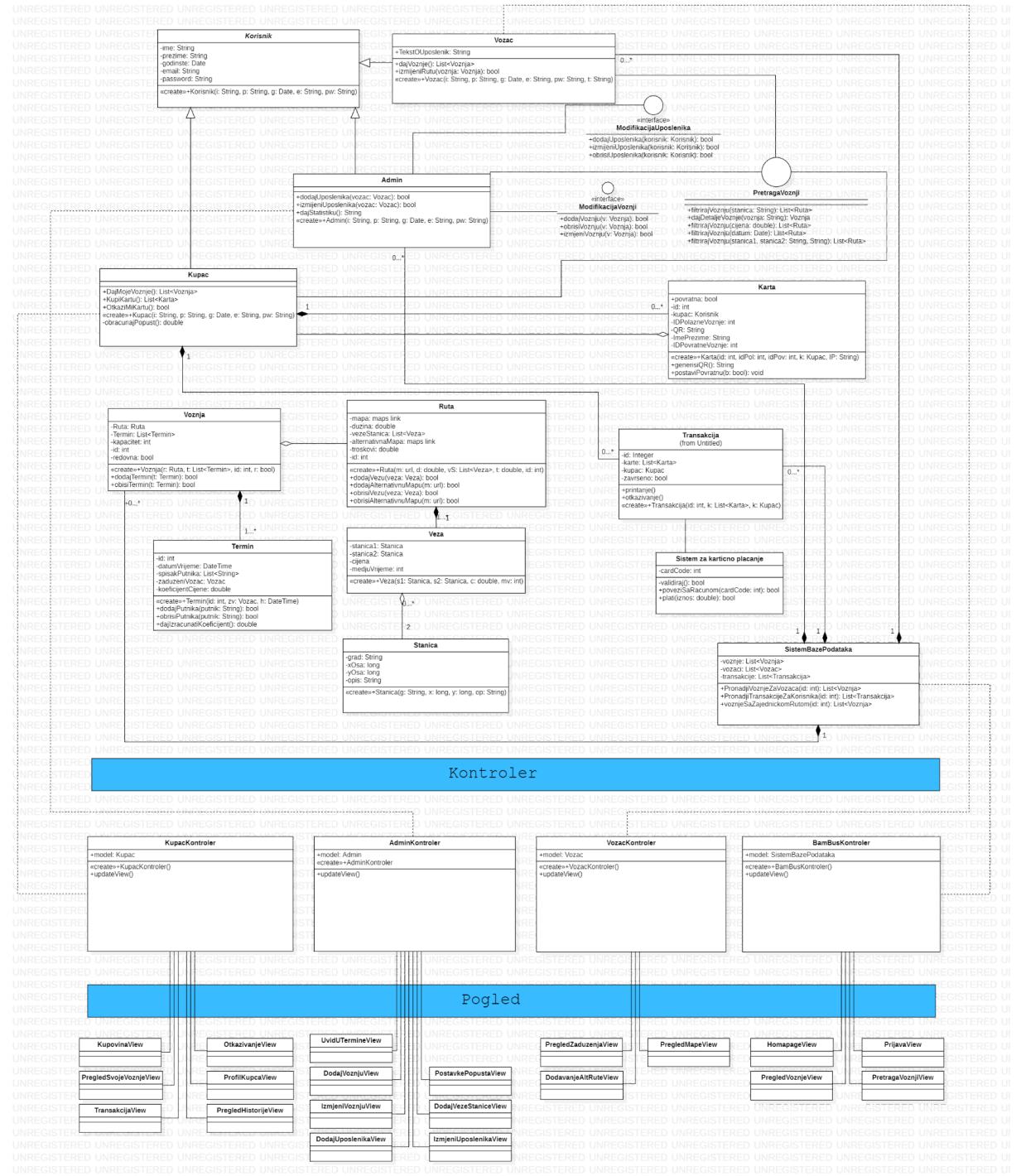
Nakon prve verzije, dijagram je pretrpio određene izmjene kako bi zadovoljio SOLID principe. Dobili smo sljedeći dijagram koji obuhvata sve SOLID principe.

Dijagram klasa sa SOLID principima



Nakon ovoga predđeno je na MVC dijogram klasa.

MVC dijagram klasa



STRUKTURALNI PATERNI

- **Adapter pattern**

Adapter patern služi da se postojeći objekat prilagodi za korištenje na neki novi način u odnosu na postojeći rad, bez mijenjanja same definicije objekta. U našem slučaju ovaj pattern bi se mogao relizovati dodavanjem adaptera za kupovinu karata na autobuskoj stanici tj. Šalterskoj prodaji. Prodavač bi bio loginovan kao kupac, samo sa različitim pogledima i metodama u odnosu na običnog kupca.

- **Facade pattern**

Ovaj patern koristimo kada nije potrebno koristiti i poznavati cijeli sistem već koristiti određeni dio funkcionalnosti. Konkretno, vozač može dodavati alternativne rute na Google mapama dodavanjem koordinata početne i krajnje tačke puta , pa time nije potrebno da poznaje realizaciju kompletne mape, već se koristi samo njen dio.

- **Decorator pattern**

Ovaj patern koristimo kada želimo modifikaciju vršiti pomoću postojećih klasa, bez dodavanja većeg broja novih. Ovaj patern je iskorišten u kreiranju vožnje. Pri kreiranju klase vožnja kaskadno učestvju klase ruta, veza, stanica i termin. Prema tome omogućena je modifikacija vožnje na različitom nivou.

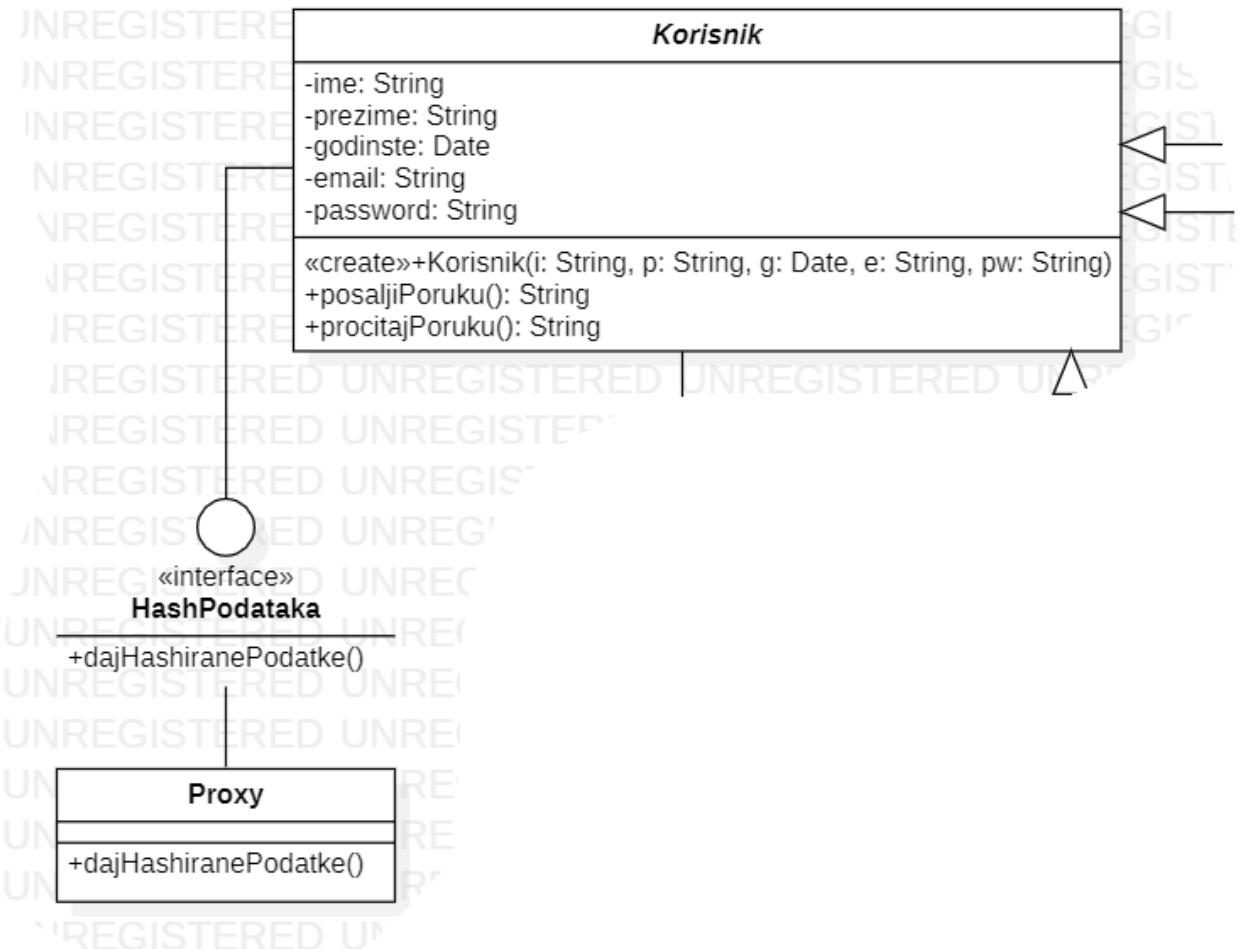
- **Bridge pattern**

Pattern omogućava da se iste operacije primjenjuju nad različitim podklasama. Bridge pattern je realizovan kod pretrage vožnji. Metode za pretragu vožnje su iste, ali su implementirane različito za različite korisnike.

- **Proxy pattern**

Proxy patern služi za dodatno osiguravanje objekata od pogrešne ili zlonamjerne upotrebe. Heširanjem e-maila i passworda korisnika sistema,

osiguravamo korisnike od krađe njihovih profila.



- **Composite pattern**

Omogućuje pozivanja iste metode nad različitim objektima sa različitim implementacijama. Ovaj pattern nije relizovan u našem projektu u potpunosti. Iako je zadovoljena hijerarhija klase, nije omogućeno pozivanje iste metode nad različitim objektima. U slučaju potrebe ovaj pattern je moguće realizovati.

- **Flyweight pattern**

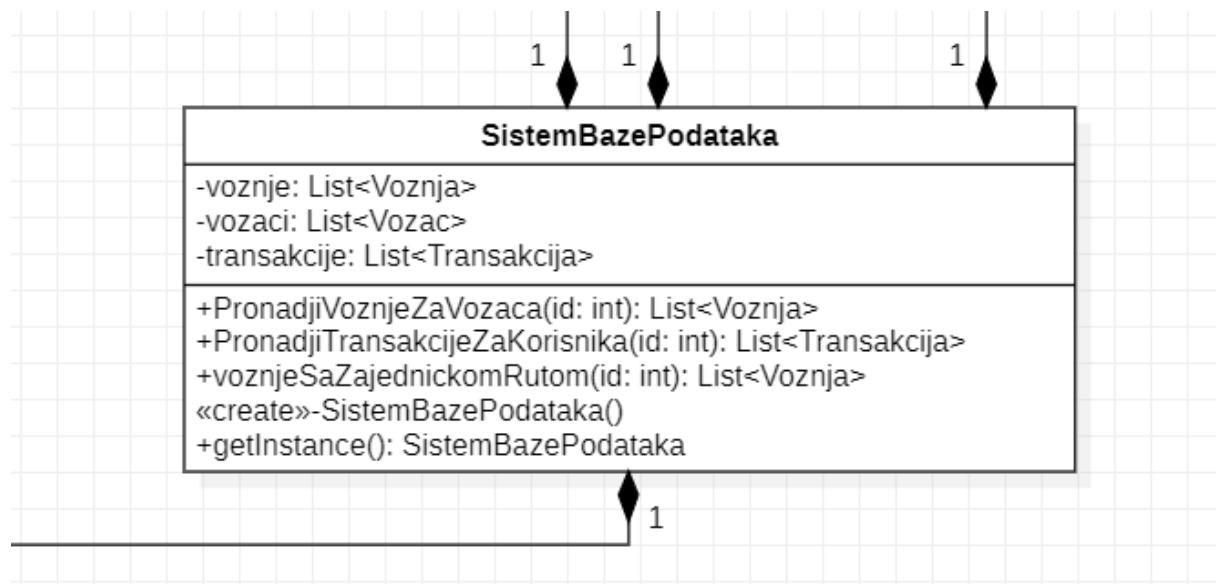
Ovaj pattern podrazumijeva mogućnost ponovne upotrebe istih objekata. Prilikom kreiranja ili modifikacije vožnji moguća je ponovna upotreba već

kreiranih stanica i veza između istih. Time je omogućena manja potrošnja memorije. Nismo implementirali ovaj pattern, međutim mogli bi koristiti već kreirane rute za nove vožnje.

Kreacijski patterni

1. Singleton pattern

Uloga Singleton paterna je da osigura da se klasa može instancirati samo jednom i da osigura globalni pristup kreiranoj instanci klase. BamBus projekat posjeduje klasu SistemBazePodataka (BamBus system) na koju je primjenjen singleton pattern. Ovu klasu je moguće instrancirati samo jednom. Time je izbjegnut mogući konflikt i neadekvatno ponašanje programa.



2. Prototype pattern

Uloga Prototype paterna je da kreira nove objekte klonirajući jednu od postojećih prototip instanci (postojeći objekat). Nažalost implementacija ovog patterna zahijteva velike promjene sistema. Na projektu ovaj pattern bi se mogao implementirati za klasu **Voznja**. Ukoliko vožnja ne bi sadržavala listu termina nego samo jedan termin bilo bi moguće kopiranje vožnji s tim da bi se jedino termin promijenio, a ostali atributi ostali isti.

3. Factory Method pattern

Uloga Factory Method paterna je da omogući kreiranje objekata na način da podklase odluče koju klasu instancirati. Različite podklase mogu na različite načine implementirati interfejs. Ovaj pattern nije iskorišten jer nema velike potrebe za njim, međutim njegovu primjenu bi vidjeli na

korištenju klase kupac, vozač i admin koji bi u nekoj situaciji kroz polimorfizam pozivali odgovarajuće metode.

4. Abstract Factory pattern

Abstract Factory patern omogućava da se kreiraju familije povezanih objekata/produkata. Na osnovu apstraktne familije produkata kreiraju se konkretne fabrike (factories) produkata različitih tipova i različitih kombinacija. Mogla bi se napraviti klasa **popust** koja bi bila atribut klase **Kupac**. Klasa popust bi za različite vrste kupaca (student, penzioner, VIP) generisalo različite vrste popusta. Samim tim bi se izbjeglo nepotrebna upotreba if-else iskaza.

5. Builder pattern

Uloga Builder paterna je odvajanje specifikacije kompleksnih objekata od njihove stvarne konstrukcije. Isti konstrukcijski proces može kreirati različite reprezentacije. Builder pattern je primjenjiv u relaciji klasa **Ruta**, **Veza**, **Stanica**. Osnovna komponenta je klasa Ruta koja bi se gradila pomoću njenih veza i stanica.

PATERNI PONAŠANJA

1) Strategy pattern

Strategy pattern služi kako bi se različite implementacije istog algoritma izdvojile u različite klase, te kako bi se omogućila brza i jednostavna promjena implementacije koja se želi koristiti u bilo kojem trenutku. Na ovaj način omogućava se i jednostavno brisanje ili dodavanje novog algoritma koji se može koristiti po želji. Ovaj pattern nismo implementirali. Međutim kada bi željeli implementirati ovaj pattern to bi bilo kod plaćanja karte, tačnije kada bi omogućili drugačije načine plaćanja rezervisanja karte. Dakle dodali bi interfejs **iStrategija**, koji bi imao jednu metodu **placanjeUzivo**, i napravili bi nove klase koje implementiraju ovaj interfejs, npr. klasa za plaćanje karticom, te klasa za plaćanje uživo. Napravili bi posebnu klasu koja služi samo za plaćanje karte, u njoj bi imali atribut **iStrategija**, te metodu za promjenu načina plaćanja.

2) State pattern

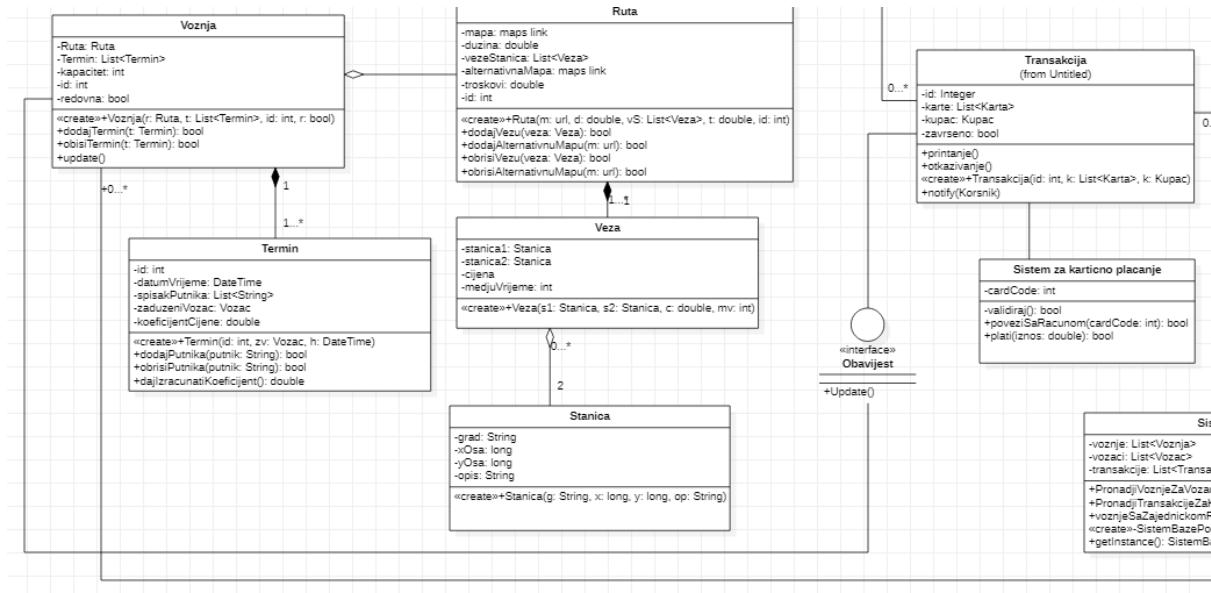
State pattern omogućava objektu da mijenja svoja stanja, od kojih zavisi njegovo ponašanje. Sa promjenom stanja objekt se počinje ponašati kao da je promijenio klasu. Stanja se ne mijenjaju po želji klijenta, već automatski, kada se za to steknu uslovi. State pattern nije implementiran, ali bi se mogao implementirati za klasu **Transakcija**. Atribut tipa bool **Završeno**, bi mijenjao svoje stanje u zavisnosti od Sistema za kartično plaćanje. Ukoliko bi **Završeno** bilo tipa true, omogućilo bi se printanje karte kao i zabrana brisanja te instance transakcije. Napravio bi se interfejs **iStanje** kojeg bi implementirale dvije klase za promjenu stanja sistema.

3) Template Method pattern

Template method pattern služi za omogućavanje izmjene ponašanja u jednom ili više dijelova. Najčešće se primjenjuje kada se za neki kompleksni algoritam uvijek trebaju izvršiti isti koraci, no pojedinačne korake moguće je izvršiti na različite načine.

4) Observer pattern

Observer pattern služi kako bi se na jednostavan način kreirao mehanizam pretplaćivanja. Preplatnici dobivaju obavještenja o sadržajima na koje su pretplaćeni, a za slanje obavještenja zadužena je nadležna klasa. Na ovaj način uspostavlja se relacija između klase kako bi se mogle prilagoditi međusobnim promjenama. BamBus sistem implementira Observer pattern na način da šalje obavijest putem email-a korsniku o uspješno izvršenoj transakciji. Unutar email-a korisnik dobija dokument koji predstavlja njegovu kartu koju može printati. Ovaj pattern je prikazan na sljedećoj slici.



5) Iterator pattern

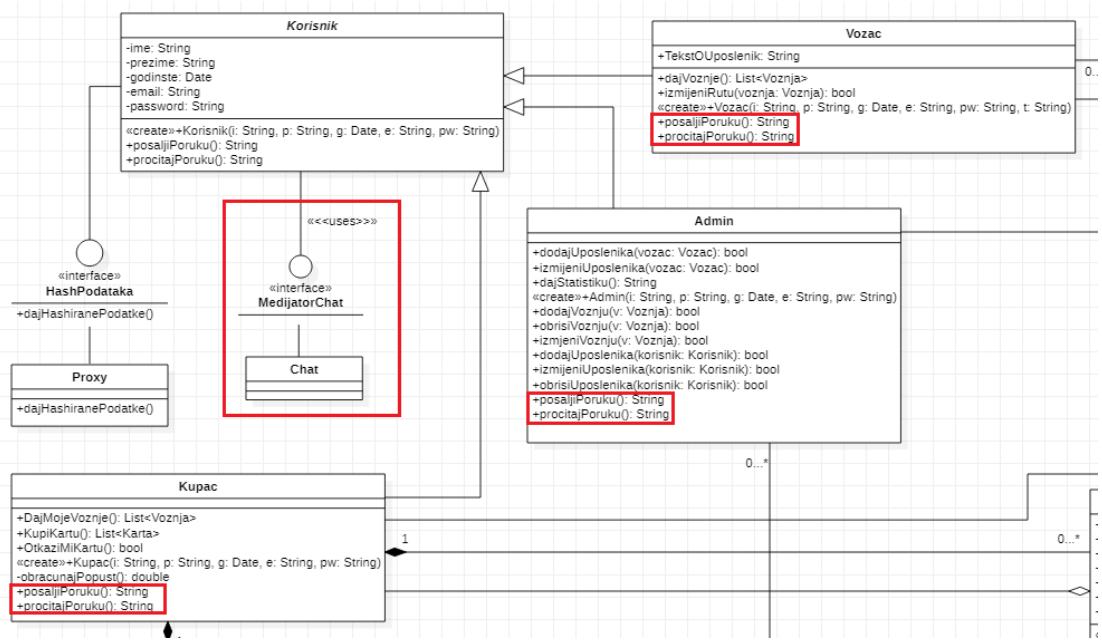
Iterator pattern namijenjen je kako bi se omogućio prolazak kroz listu elemenata bez da je neophodno poznavati implementacijske detalje strukture u kojoj se čuvaju elementi liste. Izvedba liste može biti u obliku stabla, jednostrukе liste, niza i sl., no klijentu se omogućava da na jednostavan način dolazi do željenih elemenata. Osim toga, ovaj patern preporučljivo je iskoristiti kada se za iteriranje koristi kompleksna logika koja ovisi o više kriterija. Ovaj patern nije implementiran. Iterator pattern bi se mogao implementirati na klasi vožnja. Napravio bi se interfejs `iVoznja` koji bi regulisao iteraciju za svaku instancu vožnje.

6) Chain of responsibility

Dati pattern je moguće realizirati ukoliko pravimo posebnu klasu za popust koja bi trebala više drugih faktora za realizaciju. Posmatrajmo situaciju ukoliko imamo pomoćnu klasu koja je zadužena da klasi popust javi da je kupac napravio recimo desetu vožnju čime mu se omogućava popust na neko sljedeće putovanje. Tada još ako je i student i taj faktor bi se uzeo obzir na sljedeću cijenu zajedno sa varijantom popusta nakon 10-te vožnje. Dakle različite vrste popusta i statusa kupca sastavljaju ustvari novu kartu koja važi određeni period.

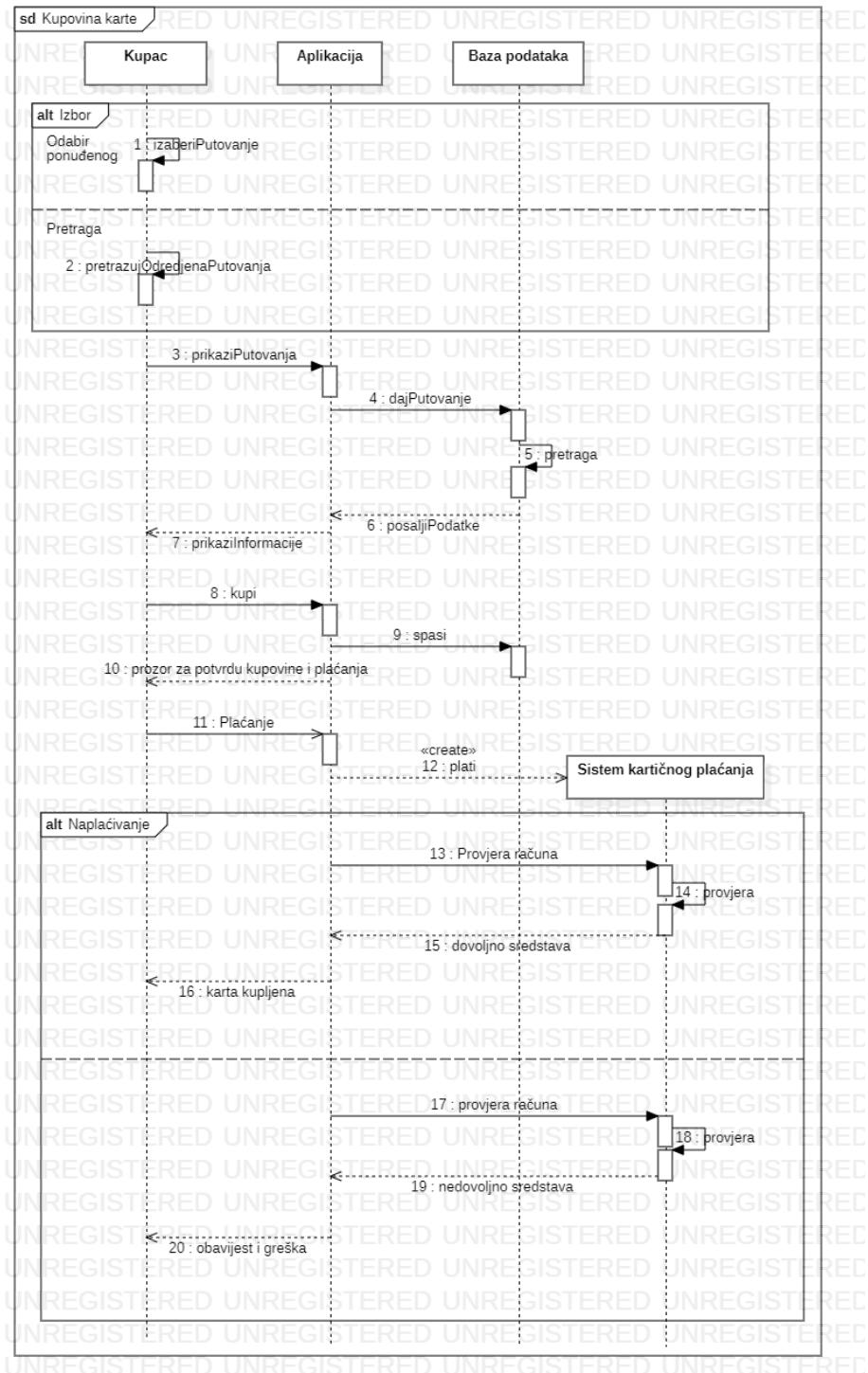
7) Mediator pattern

Patern koji je primjenjen na komunikaciji između Vozača i Admina (kojeg možemo posmatrati kao kordinatora) i Kupca putem svojstvenog chata za potrebe obavljanja njihovog dijela posla. Kreiran je interface `MediatorChat` kao i konkretna klasa `Chat`.

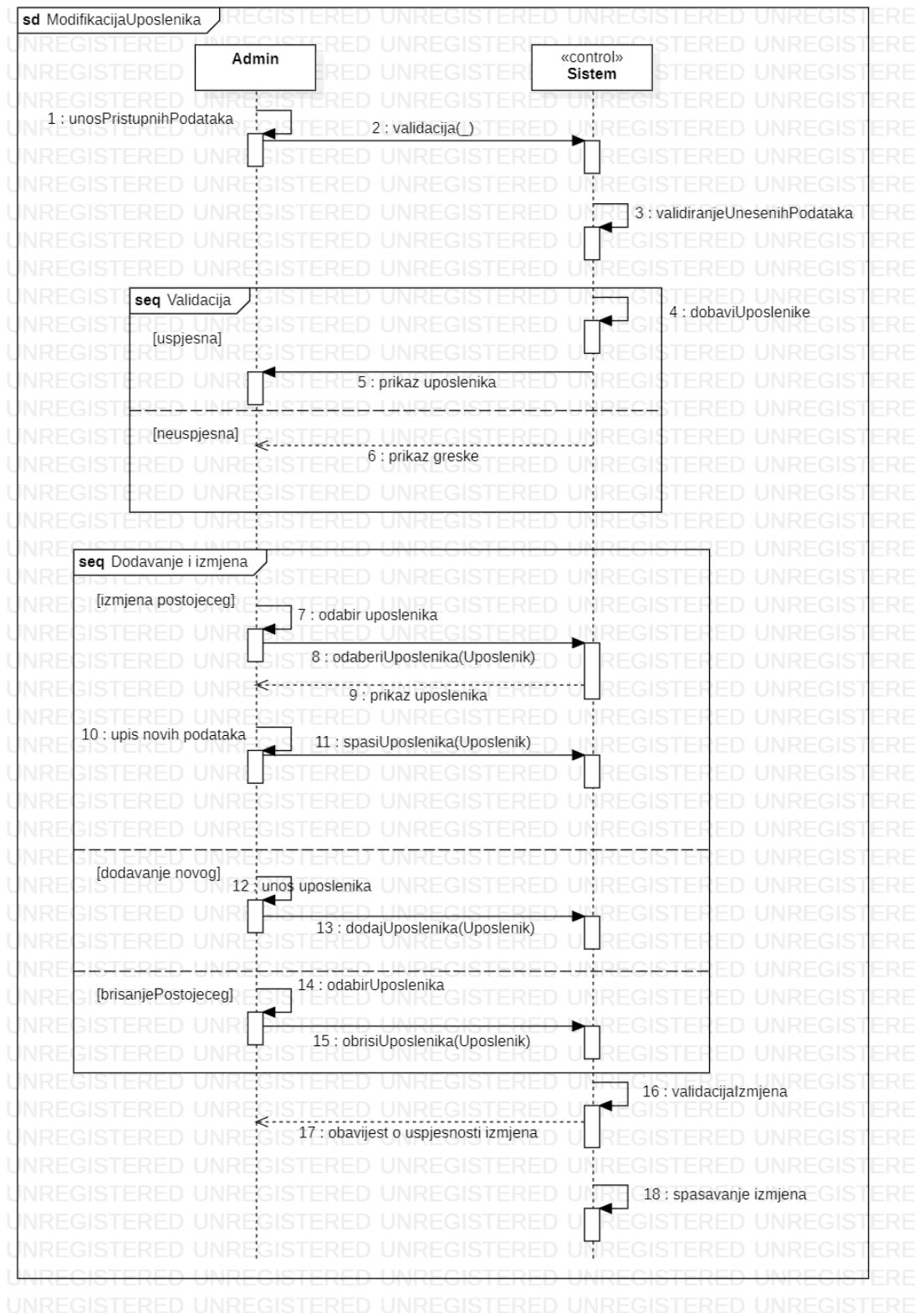


Dijagram sekvenci

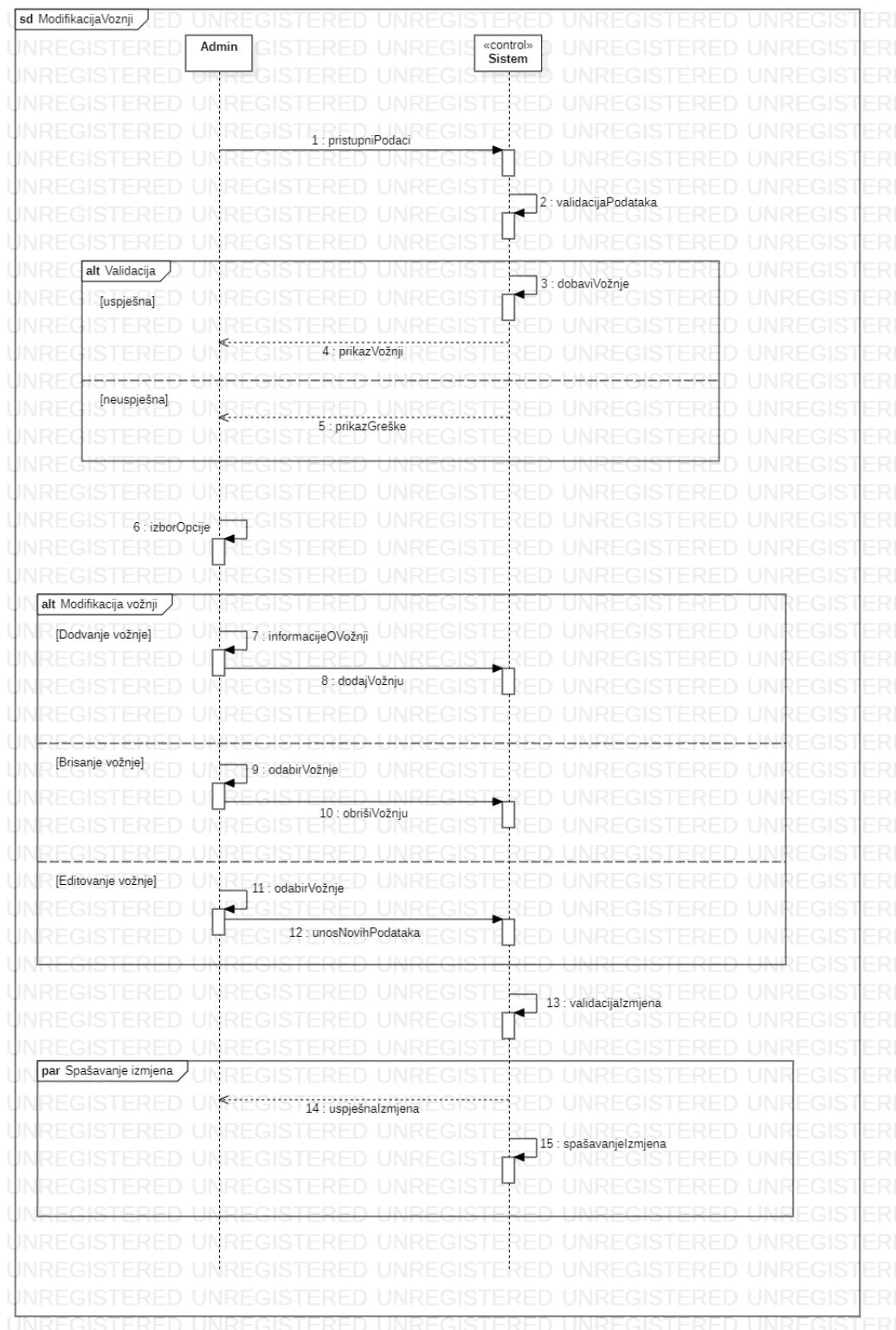
Dijogram sekvence kupovine karte



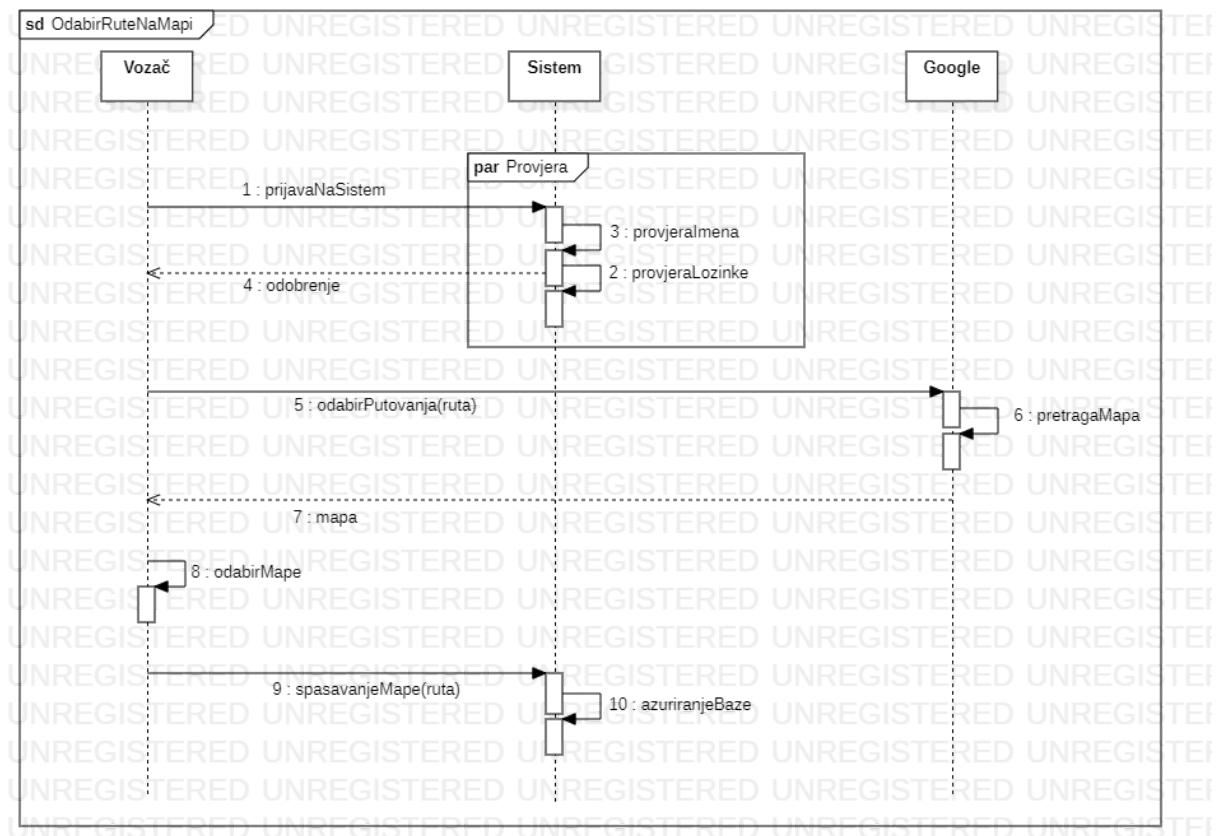
Dijagram sekvenca modifikacije uposlenika



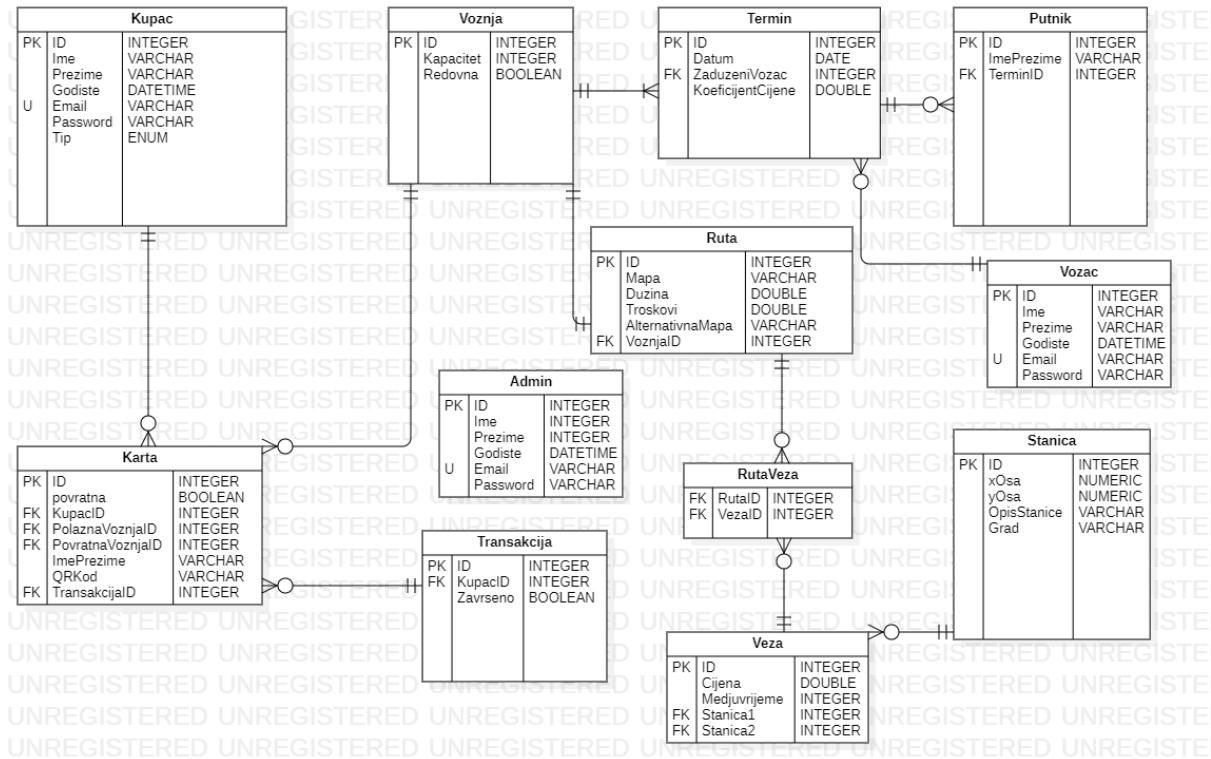
Dijagram sekvenca modifikacije vožnji



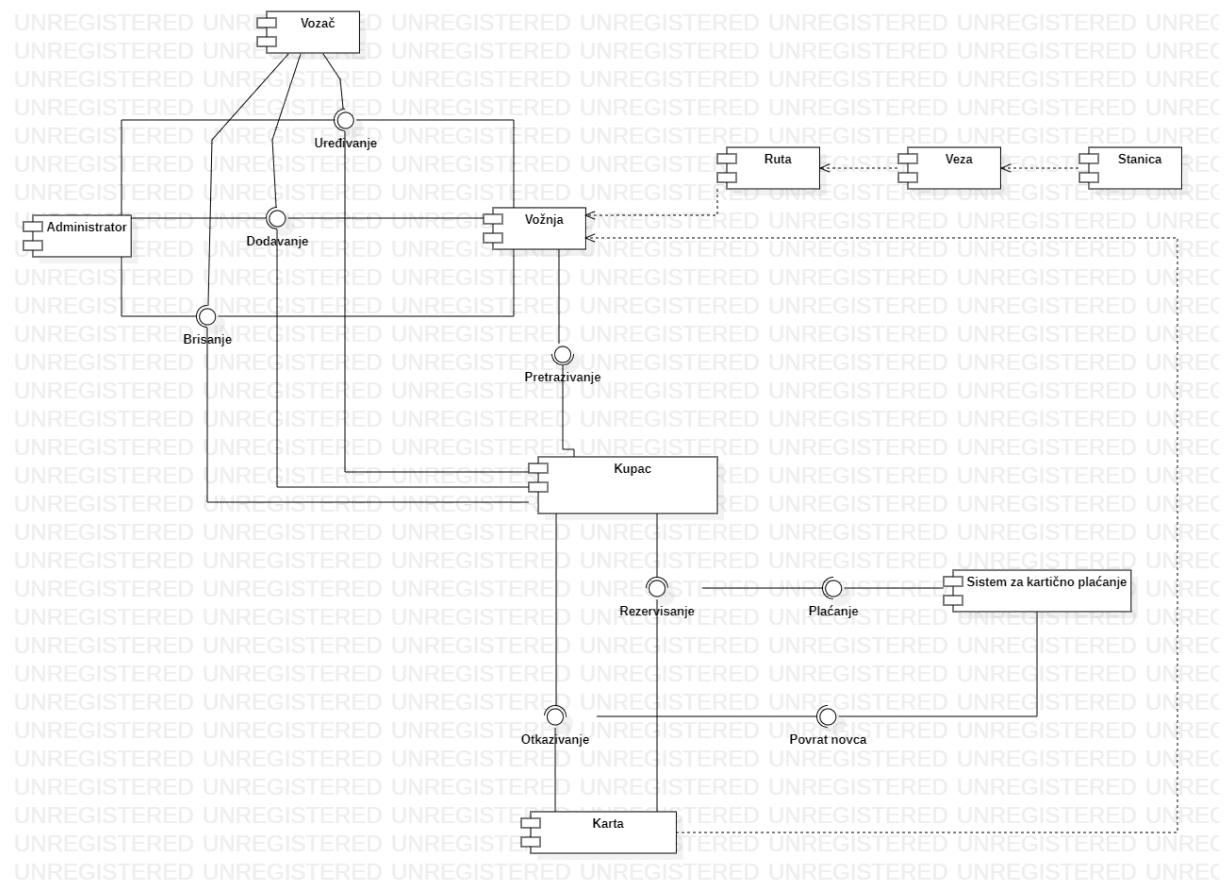
Dijagram sekvenca odabira rute na mapi



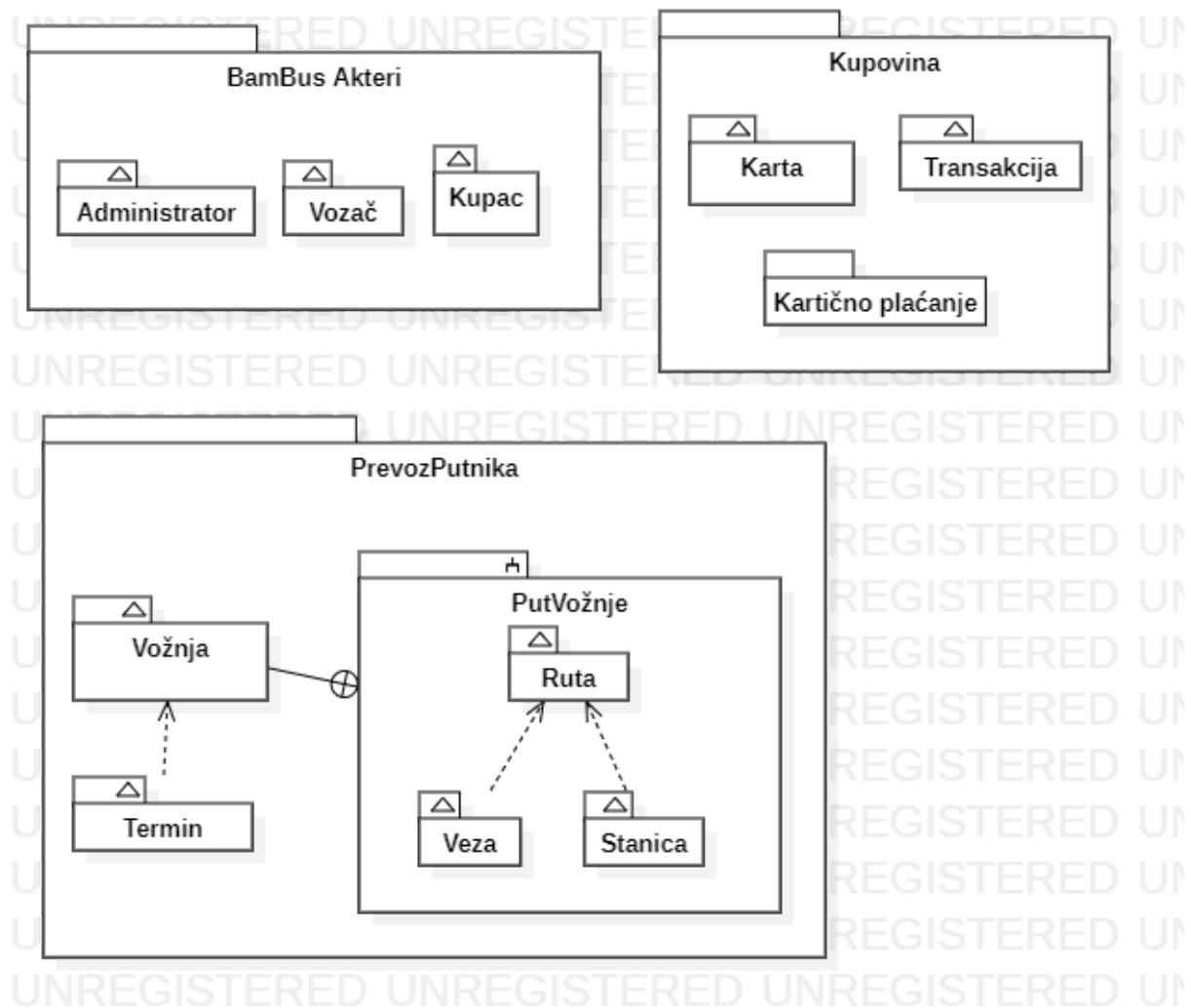
ERD dijgram



Dijagram komponenti



Dijagram paketa



Dijagram rasporedivanja

