

Univerzitet u Sarajevu
Elektrotehnički fakultet Sarajevo
Računarstvo i informatika

Naziv projekta: EventManager

Dokumentacija modela

Predmet: Objektno orijentisana analiza i dizajn

Grupa 4

Članovi grupe: Zehra Javdan, Selma Kahvedžić, Medina Kapo

Sarajevo, juni 2020. Godine

| | |
|---|----|
| TEMA: Event Manager | 3 |
| Opis projekta i ideja | 3 |
| Funkcionalnosti sistema..... | 3 |
| Korisnik..... | 3 |
| Ustanova | 3 |
| Administrator | 4 |
| Akteri sistema | 4 |
| USE-CASE dijagram..... | 5 |
| Scenariji..... | 6 |
| Dijagrami aktivnosti kreirani na osnovu scenarija | 22 |
| Prototipovi formi sistema | 28 |
| Inicijalni opis klasa za sistem..... | 44 |
| Dijagram klasa..... | 56 |
| SOLID principi..... | 58 |
| Paterni..... | 59 |
| Dijagrami sekvence | 73 |
| Dijagram komponenti | 81 |
| Dijagram složene strukture..... | 82 |
| ER dijagram baze podataka..... | 83 |
| MVC patern | 84 |
| WEB servis..... | 85 |

TEMA: Event Manager

Kulturni događaji na jednom mjestu

***pošto su dijagrami dosta glomazni I postoji mogućnost da se kvaljeta promijenila, svi ovdje navedeni dijagrami se nalaze na Github profile grupe u folderu "FINALNO".

Opis projekta i ideja

Zbog prevelikog broja obaveza koje nosi moderno doba, često ne stignemo pregledati na internetu koje su predstave aktuelne, koji je film zadnji došao u kina ili koji koncert se održava u blizini. Besplatna aplikacija koju ćemo razvijati će omogućavati pregled kulturnih događaja na jednom mjestu, pretragu događaja na osnovu lokacije, rezervisanje karte/mjesta za događaj. Ovaj projekat je namijenjen užurbanim ljudima, ali koji imaju želju da ostanu u toku sa svim što se dešava oko njih i što ih privlači. Događaji koje će aplikacija da obuhvata su događaji vezani za pozorišta, kina, koncerte, kulturni događaji na otvorenom, muzeji.

Funkcionalnosti sistema

Aplikacija bi trebala da omogući sljedeće funkcionalnosti:

Korisnik

- Pretraga novosti (događaja)
- Mogućnost pravljenja 2 vrste profila - besplatni profil i VIP profil
- Ostavljanje komentara na događaj, ocjenjivanje događaja kao i same aplikacije
- Mogućnost pravljenja liste želja, te pregled svih događaja koje je rezervisao
- Mogućnost rezervacije mjesta ili karte za događaj
- Mogućnost otkazivanja rezervacije
- Mogućnost sticanja 2 kategorije posebnog (VIP) statusa – srebreni i zlatni status, na osnovu broja transakcija u aplikaciji
- Mogućnost online plaćanja rezervacija
- Mogućnost printanja bilo karte ili potvrde o odrađenoj rezervaciji preko aplikacije, tj. mogućnost printanja izvještaja o stanju rezervacije
- Mogućnost provjere popunjenoosti kapaciteta događaja
- Mogućnost slanja zahtjeva za dodavanjem događaja u ime neke ustanove

Ustanova

- Mogućnost da kreira i briše ustanovu

- Mogućnost da dodaje novi događaj
- Mogućnost da šalje novosti korisnicima koji žele da primaju novosti iz ove ustanove
- Mogućnost kontrole rezervacija
- Mogućnost uređivanja događaja (stavljanje limita, otkazivanje)
- Mogućnost editovanja profila
- Mogućnost davanja ovlasti nekom korisniku da u ime ustanove dodaje događaje
- Mogućnost da zakaže dodatne popuste i način kako neko ko nije VIP korisnik da ostvari popust
- Mogućnost davanja VIP statusa korisniku

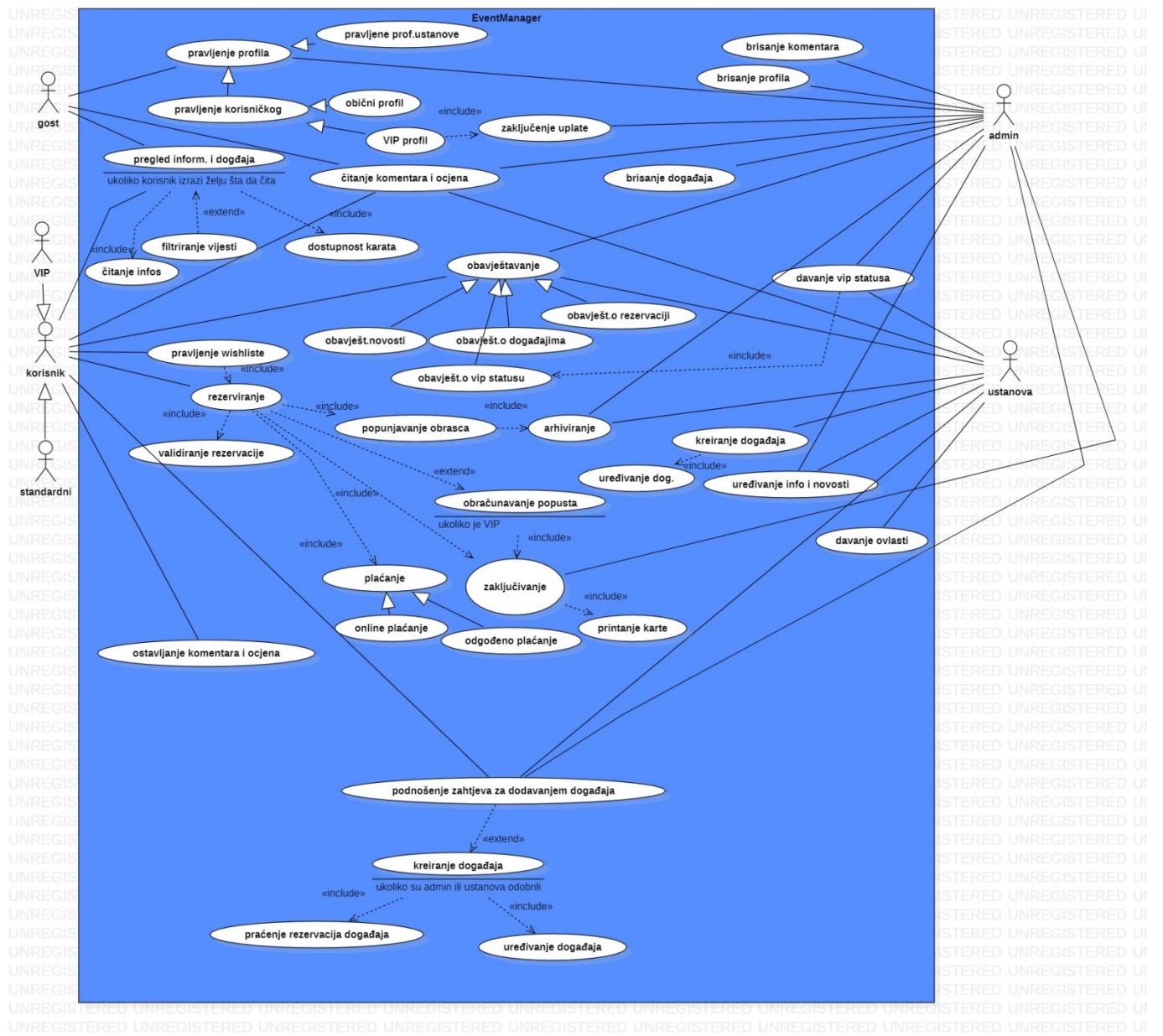
Administrator

- Mogućnost uređivanja događaja
- Mogućnost odobravanja korisničkih događaja
- Mogućnost ažuriranja stranice i informacija vezanih za mesta za koje su namijenjeni događaji
- Mogućnost brisanja profila
- Mogućnost kontrole kreiranja profila
- Mogućnost kontrole rezervacija i plaćanja
- Mogućnost kontrole komentara i brisanja neprimjerenih
- Mogućnost kontrole i dodjele VIP statusa za korisnike
- Mogućnost pristupa bazi podataka i arhivama

Akteri sistema

- Guest-neprijavljeni posjetilac, ima mogućnost pregleda osnovnih informacija i kreiranje profila
- Korisnik -prijavljeni korisnik (korisnik sa kreiranim korisničkim računom)
- VIP korisnik -ima iste mogućnosti kao i klasični korisnik, samo što može da odabere određen broj puta popust na neki događaj u mjesecu (odnosno godini) u kojem je proglašen za VIP i ima veću prednost prilikom rezervacije u odnosu na klasičnog korisnika
- Ustanova-korisnik koji predstavlja pravno lice, organizator događaja
- Administrator-osoba koja drži aplikaciju pod kontrolom i ima najveće ovlasti

USE-CASE dijagram



Scenariji

1. Prvo pristupanje aplikaciji u ulozi gosta

Scenarij 1: „Prvi pristup korisnika aplikaciji-pravljenje profila“

Glavni tok: Uspješno registrovan račun Preduvjeti: Gost ima dostupan Web interfejs Posljedice: Gost ima potvrdu da je uspješno registrovan i da je od sada korisnik

| GOST | SISTEM | USTANOVA | ADMIN |
|---|---|---|--|
| 1. Pristupanje interfejsu aplikacije | 2. Prikaz trenutnih događaja, te mogućnost izbora | | |
| 3. Čitanje komentara i ocjena, te ostalih vijesti | | | |
| 4. Filtriranje vijesti | | | |
| 5. Dostupnost i cijena karata | | | |
| 6. Pravljenje profila (unesenje imena, prezimena i drugih podataka) | 7. Validacija unesenih podataka | 8. Odobravanje | 9. Odobravanje |
| 10. Mogućnost dodavanja vlastite ustanove za organizaciju događaja | 11. Validacija unesene ustanove | 12. Obavještenje o uspješnoj rezervaciji ustanove | 13. Odobravanje rezervacije |
| | 14. Generiranje jedinstvenog broja korisnika | | 15. Potvrda jedinstvenog broja korisnika |
| | 16. Otvaranje prozora sa potvrdom o registraciji | | |

2. Kupovina karata

Glavni tok: Završava uspješnim plaćanjem karte Preuvijeti: Korisnik ima dostupan Web interfejs kao i registrovan korisnički profil Posljedice: Korisnik ima potvrdu o uspješnom plaćanju karte za događaj

| KORISNIK | SISTEM | USTANOVA | ADMIN |
|---|---|--------------------------------------|--|
| 1. Pristupanje interfejsu za kupovinu | 2. Prikaz aktuelnih događaja i omogućavanje izbora | | |
| 3. Izbor događaja i zaključivanje | | | |
| 4. Popunjavanje obrasca | | | |
| 5. Arhiviranje | 6. Validacija narudžbe | 7. Potvrda i odobravanje od ustanove | 8. Potvrda i odobravanje od administratora |
| 9. Unošenje imena, prezimena, adrese, broja kartice i datuma isteka | 10. Validacija unesenih podataka | | 11. Validacija podataka od administratora |
| 12. Obračun eventualnih popusta | | | |
| 13. Odabir načina plaćanja (odgođeno plaćanje, online plaćanje) | 14. Proslijedivanje transakcije ustanovi | 15. Prihvatanje naruđbe | 16. Potvrda naruđbe |
| | 17. Označavanje transakcije prihvaćenom i prikaz poruke o prihvatanju korisniku | | |

| | | | |
|--|--|--|--------------------------------------|
| | 18. Generisanje jedinstvenog broja transakcije | | 19. Potvrda generisanog broja |
| | 20. Prikaz prozora sa tiketom i brojem narudžbe, te mogućnost štampanja istog | | |
| | 21. Printanje tiketa i završavanje interakcije sa sistemom | | |

3. Brisanje profila korisnika

Glavni tok: Završava uspješno izbrzanim profilom
Preduvjeti: Korisnik ima dostupan Web interfejs i korisnički račun
Posljedice: Korisnik ima potvrdu da je njegov profil (svi podaci) izbrisana

| KORISNIK | SISTEM | USTANOVA | ADMIN |
|--|--|--|-------------------------------|
| 1. Pristupanje interfejsu | 2. Prikaz trenutne ponude i novosti | | |
| 3. Izbor opcije za postavke profila | | | |
| 4. Izbor opcije za brisanje | | | |
| 5. Unošenje podataka (ime, prezime, e mail, jedinstveni korisnicki broj i sl.) | 6. Provjera podataka korisnika | | |
| | 7. Provjera računa korisnika (da li je izmirio sve obaveze što se tiče plaćanja) | 8. Izdavanje potvrde da je korisnik uredno izmirio sve obaveze | 9. Odobravanje administratora |
| | 10. Otvaranje prozora sa upitom da li je siguran da želi izbrisati profil | | |
| | 11. Otvaranje prozora sa upitom koji je razlog zatvaranja | | |

| | | | |
|---|--|--|--|
| 12. Izbor nekog od ponuđenih razloga za gašenje profila ili unos vlastitog izbora | 13. Otvaranje potvrde da je profil obrisan | | |
| 14. Odlazak sa stranice | | | |

4. registrovanje VIP korisnika

Opis: Gost Web-interfejsa pregleda stranicu i odlučuje da se registruje na aplikaciju. Prilikom aplikacije ozdučuje da vrsta profila bude korisnički VIP profil.

Glavni tok: Završava uspješno kreiranim profilom

Preduvjeti: Gost ima dostupan Web-interfejs za pristup aplikaciji

Posljedice: Gost ima profil kojem može ubuduće da pristupa

Tok događaja:

| Gost | Sistem EventManager | Ustanova | Administrator |
|---|---|----------|---|
| 1.Pristupa interfejsu za upravljanje događajima | 2.Ponuda da se registruje | | |
| 3.Odlučuje se registrovati | 4.Nudi mogućnost registracije kao korisnik ili kao ustanova | | |
| 5.Bira mogućnost da pravi profil za korisnika | 6.Nudi mogućnost da bude obični ili VIP korisnik | | |
| 7.Korisnik bira VIP status | 8.Nudi obrazac za popunjavanjem podataka i za uplatu VIP članstva | | |
| 9.Unošenje osnovnih podataka (ime, prezime, mail, nadimak za profil, datum rođenja, broj kartice za upлатu, šifra za pristupanje) | 10.Validacija unešenih podataka | | |
| | 11.Prosljeđivanje obrazca za prijavu administratoru stranice | | 12.Izvještavanje da je prijava prihvaćena |
| | 13.Potvrđenje da je kreiranje profila | | |

| | | | |
|--|---|--|--|
| | uspjelo | | |
| | 14.Proslijeđivanje na stranicu sa vijestima/događajima uz mogućnost rezervacije | | |

Poslovno pravilo 1: Ukoliko gost koji se registrira na aplikaciju želi da bude VIP korisnik,tada mora da unese broj svoje kartice kako bi se izvršila uplata.Ukoliko se ne unese broj kartice ili administrator ne dobije obavijest da je uplaćen novac sa kartice,potrebno je obavijestiti korisnika i vratiti ga ponovno na popunjavanje obrasca.

Alternativni tok 1:Neuspješna validacija pristupnog obrasca-nedostaje broj kartice za uplatu

Preduvjeti:Na koraku 10.glavnog toka nije zadovoljeno poslovno pravilo 1.

Tok događaja:

| Gost | EventManager | Ustanova | Administrator |
|------|---|----------|--|
| | 1.Upozorenje da ne prolazi validaciju | | 2.Prijavni obrazac ne prolazi kontrolu zbog poslovnog pravila 1. |
| | | | 3.Upozoravanje korisnika da postoji problem |
| | 4.Ažuriranje stranice da korisnik ponovno unese svoje podatke | | |

Alternativni tok 2:Neuspješna validacija pristupnog obrasca-nije uplaćeno(transakcija odbijena)

Preduvjeti:Na koraku 10.glavnog toka nije zadovoljeno poslovno pravilo 1.

Tok događaja:

| Gost | EventManager | Ustanova | Administrator |
|------|---|----------|--|
| | | | 1.Prijavni obrazac ne prolazi kontrolu zbog poslovnog pravila 1. |
| | | | 2.Upozoravanje korisnika da postoji problem |
| | 3.Ažuriranje stranice da korisnik ponovno unese svoje podatke | | |

Alternativni tok 3: Odustajanje od pravljenja profila

Preduvjeti: Na bilo kojem od koraka od 2. do 10. koraka glavnog toka (dakle, od ponude na napravi profil do validacije)

Tok događaja:

| Gost | EventManager | Ustanova | Administrator |
|-------------------------------------|--|----------|---------------|
| 1. Želi da odustane od registracije | 2. Upozorenje da tako gubi sve podatke | | |
| | 3. Vraćanje na početnu | | |

Alternativni tok 4: Odustajanje od VIP profila, prelazak na obični profil

Preduvjeti: Na bilo kojem od koraka od 7. do 10. koraka glavnog toka, korisnik odustaje od VIP članstva

Tok događaja:

| Gost | EventManager | Ustanova | Administrator |
|------------------------------------|---|----------|---------------|
| 1. Želi da odustane od VIP profila | 2. Nudi mogućnost prelaska na obični profil | | |
| 3. Želi da ima obični profil | 4. Otvara obrazac za popunjavanje običnog profila | | |

5. rezervisanje karte VIP korisnika

Opis: Korisnik sa pristupom aplikaciji kao VIP korisnik želi da pregleda događaje, rezerviše mjesto na događaju i izvrši plaćanje. S obzirom da je u pitanju VIP korisnik, ukoliko nije potrošio sve bonusne, nudi mu se mogućnost da iskoristi popuste prilikom plaćanja. Korisnik odlučuje da uzme popust I da plati online. (Ovo je generalni prikaz za VIP korisnika (bez obzira da li je u pitanju srebreni, zlatni VIP ili korisnik koji je samoinicijativno odlučio da bude VIP korisnik)).

Glavni tok: Završava uspješno rezervisanim mjestom/kartom na događaju

Preduvjeti: Korisnik ima već napravljen VIP račun

Posljedice: Korisnik može da isprinta bilo kartu bilo potvrdu o online rezervaciji

Tok događaja:

| VIP Korisnik | EventManager | Ustanova | Administrator |
|---|--|----------|---------------|
| 1. Pregleda događaje i može provjeriti zauzetost događaja | 2. Prikaz trenutne ponude događaja, provjera | | |

| | | | |
|--|--|--|---|
| | zauzetosti mjesta na događajima | | |
| 3.Događaj dodaje u wishlistu | 4.Nudi mogućnost pregleda wishliste | | |
| 5.Bira događaj u wishlisti i odlučuje da želi ići na događaj | 6.Nudi prijavni obrazac za popunjavanje osnovnih podataka | | |
| 7.Unošenje osnovnih podataka potrebnih za događaj(ime, prezime, mail, poziciju gdje želi da sjedi) | 8.Validacija unešenih podataka | 9.Arhiviranje podataka unešenih za događaj | 10.Arhiviranje podataka unešenih za događaj |
| | 11.Nudi mogućnost plaćanja | | |
| 12.Bira da plati online | 13.Nudi mogućnost da iskoristi VIP bon za popust | | |
| 14.Korisnik bira opciju za popust | 15.Obračunavanje popusta i prikaz iznosa za uplatu | | |
| 16.Unos kartice preko koje će ići plaćanje | 17.Validacija rezervacije | | 18. Kontroliše obračunati iznos i izvještava kako se uplata rezervacija uspješno završila |
| | 19.Zaključivanje rezervacije | | |
| | 20.Obavještava korisnika kako je uspješno završena rezervacija i transakcija | | |
| | 21.Nudi mogućnost korisniku da printa kartu | | |
| 22.Printa kartu | | | |

Pravilo koje se mora zadovoljiti prilikom validacije unešenih podataka je da su unešeni svi potrebni podaci i da su jednaki podacima unešenim prilikom registracije profila (isto ime, prezime, mail)

Alternativni tok 1: Neuspješna validacija unešenih podataka

Preduvjeti: Na koracima 8. i 9. nije zadovoljeno pravilo

Tok događaja:

| Korisnik VIP | EventManager | Ustanova | Administrator |
|--------------|---|----------|--|
| | | | 1. Prilikom arhiviranja unešeni podaci i podaci iz baze nisu isti - brisanje podataka iz arhive ustanove i arhive administratora |
| | | | 2. Obavještavanje korisnika kako nije sve uredu |
| | 3. Ažuriranje stranice | | |
| | 4. Data mogućnost ponovnog unosa podataka | | |

Pravilo koje se mora zadovoljiti prilikom validacije rezervacije je da se uplata izvrši

Alternativni tok 2: Neuspješna validacija rezervacije zbog neuplaćenog iznosa

Preduvjeti: Na koraku 16. validacija rezervacije nije uspjela

Tok događaja:

| Korisnik VIP | EventManager | Ustanova | Administrator |
|--------------|---|----------|---|
| | | | 1. Nedostaje uplata obračunatog iznosa |
| | | | 2. Obavještavanje korisnika kako nije sve uredu |
| | 3. Ažuriranje stranice | | |
| | 4. Data mogućnost ponovnog unosa podataka | | |

Alternativni tok 3: Odustajanje od rezervacije

Preduvjeti: Korisnik je već logovan i odabrao je događaj koji želi da rezerviše (odustajanje se može desiti na koraku 7.)

Tok događaja:

| Korisnik VIP | EventManager | Ustanova | Administrator |
|--------------|--------------|----------|---------------|
| | | | |

| | | | |
|--|--|--|--|
| | 1.Obavještava korisnika kako će tako da izgubi sve već unešene podatke na pristupnom obrascu | | |
| | 3.Ažuriranje stranice | | |
| | 4.Data mogućnost ponovnog unosa podataka | | |

Alternativni tok 4: Odustajanje od rezervacije nakon ispravnog unosa podataka,a prije unosa podataka vezanih za plaćanje

Preduvjeti:Korisnik je već logovan I odabrao je događaj koji želi da rezerviše,podaci koji su unešeni su već validirani I ispravni(odustajanje se može desiti na koracima 11.-17.)

Tok događaja:

| Korisnik VIP | EventManager | Ustanova | Administrator |
|--------------|--|----------|-------------------------------|
| | 1.Obavještava korisnika kako će tako da izgubi sve već unešene podatke na pristupnom obrascu | | 2.Brisanje podataka iz arhive |
| | 3.Ažuriranje stranice | | |
| | 4.Data mogućnost ponovnog unosa podataka | | |

Alternativni tok 5: Na koraku 14:Korisnik bira da iskoristi popust,a već je sve iskoristio

Preduvjeti:Korisnik je već logovan I odabrao je događaj koji želi da rezerviše,međutim iskoristio je sve popuste,te nastavlja sa uplatom rezervacije po punoj cijeni

Tok događaja:

| Korisnik VIP | EventManager | Ustanova | Administrator |
|--------------|--|----------|---------------|
| | 1.Obaviještava korisnika kako je iskoristio sve popuste I da nema pravo na popust u ovom mjesecu | | |
| | 2.Ažuriranje stranice | | |
| | 3.Data mogućnost update po punoj cijeni | | |

6. proglašavanje ne-VIP korisnika VIP korisnikom od strane administratora

Opis:Korisnik sa pristupom aplikaciji i već kreiranim ne-VIP računom,je u posljednjih mjesec dana ostvario potreban broj rezervacija i uplata(trenutno nevažno da li je u pitanju gold ili silver status,čisto se simulira način kako bi bilo javljeno korisniku),te postao VIP član.Korisnik dobija obavještenje prilikom prvog logovanja da je postao VIP.

Glavni tok: Završava stečenim VIP statusom I dobijanjem određenog broja bonova na korištenje

Preduvjeti: Korisnik ima već napravljen klasični račun

Posljedice: Korisnik u određenom periodu može da odabere određen broj rezervacija na kojima želi da ima popust

Tok događaja:

| Korisnik -obični | EventManager | Ustanova | Administrator |
|---------------------------------------|---|----------|---|
| | | | 1.Pristupanje arhivu |
| | | | 2.Prikaz ljudi u arhivu koji su u toku prošlog mjeseca imali potreban broj rezervacija I uplata da bi postali VIP |
| | | | 3.Obavještavanje korisnika o sticanju VIP statusa |
| | 4.Slanje obavijesti krajnjim korisnicima sa VIP statusom | | |
| 5.Otvara obavještenja na svom profilu | 6.Prilikom sljedeće rezervacije/uplate ima mogućnost da odabere da li želi iskoristiti popust | | |

7. davanje ovlasti korisniku da dodaje događaj u bazu od strane ustanove

Opis: Korisnik želi da objavi događaj u aplikaciji u ime neke ustanove koja već ima kreiran profil u aplikaciji.Korisnik nakon što se loguje,zatraži odobrenje za dodavanjem događaja.Nakon što administrator da odobrenje da se doda događaj,događaj mora biti odobren I od strane ustanove.Ustanova nakon toga daje ovlast tom korisniku da u njeno ime kreira događaj I pregleda stanje rezervacija,te da uređuje događaj.

Glavni tok: Novi događaj je kreiran,potvrđen,unešen u bazu,korisnik je dobio odobrenje da u ime ustanove objavljuje

Preduvjeti: Korisnik ima već napravljen korisnički račun koji nije račun za ustanove

Posljedice: Drugi korisnici mogu rezervisati mjesto na događaju I korisnik ima mogućnost da u ime ustanove objavljuje

Tok događaja:

| Korisnik | EventManager | Ustanova | Administrator |
|---|--|--|--|
| 1.Pristupa aplikaciji | | | |
| 2.Podnosi zahtjev za dodavanjem događaja u aplikaciju | | | 3.Odobrava događaj ukoliko je primjereno za aplikaciju |
| | | 4.Odobrava događaj | |
| | | 5.Daje ovlasti tom korisniku da upravlja tim događajem | |
| 6.Kreira događaj | 7.Izlazi obrazac za popunjavanje podataka o događaju | | |
| 8.Unosi osnovne podatke o događaju-ime,mjesto,vrijeme,kapacitet | 9.Validiranje podataka | | |
| | 10.Događaj se nalazi u aplikaciji | | |
| 11.Može da pregleda rezervacije I da uređuje događaj | | | |
| | | | 12.Briše događaj nakon isteka vremena |

Da bi događaj bio prihvaćen od strane administratora, on mora biti u istoj kategoriji za koju je aplikacija pravljena ili da ima nekih dodirnih tačaka sa nekom od ustanova. Dakle, ako je aplikacija za upravljanje kulturnim događajima, korisniku neće biti dozvoljeno da doda događaj koji je npr. događaj političke prirode

Alternativni tok 1: Nije zadovoljen uslov u 3.tački glavnog toka

Preduvjeti: Korisnik je već poslao zahtjev za kreiranje događaja

Tok događaja:

| Korisnik | EventManager | Ustanova | Administrator |
|----------|--------------|----------|-------------------------|
| | | | 1.Odbija događaj |
| | | | 2.Obavještava korisnika |

| | | | |
|--|--|--|---|
| | | | kako takav događaj nije moguće kreirati |
| | 3.Ažurira stanje | | |
| | 4.Nudi mogućnost ponovnog kreiranja događaja | | |

Alternativni tok 2:Nije zadovoljen uslov u 4.tački glavnog toka

Preduvjeti: Korisnik je već poslao zahtjev za kreiranje događaja

Tok događaja:

| Korisnik | EventManager | Ustanova | Administrator |
|----------|--|------------------|---|
| | | 1.Odbija događaj | |
| | | | 2.Obavještava korisnika kako takav događaj nije moguće kreirati |
| | 3.Ažurira stanje | | |
| | 4.Nudi mogućnost ponovnog kreiranja događaja | | |

8. davanje ovlasti korisniku da dodaje događaj u bazu od strane administratora

Opis: Korisnik želi da objavi neki događaj u aplikaciji,međutim taj korisnik nema dopuštenje ni od jedne ustanove da objavljuje u njeno ime,već želi da objavi event od svog interesa.Korisnik prilaže zahtjev za kreiranjem događaja,administrator odobrava ukoliko je događaj vezan za ono čime se aplikacija primarno bavi

Glavni tok: Novi događaj je kreiran,potvrđen i unešen u bazu

Preduvjeti: Korisnik ima već napravljen korisnički račun koji nije račun za ustanove

Posljedice: Drugi korisnici mogu rezervisati mjesto na događaju

Tok događaja:

| Korisnik | EventManager | Ustanova | Administrator |
|---|-------------------|----------|----------------------|
| 1.Podnosi zahtjev za kreiranjem događaja koji je van svih već unesenih ustanova | | | 2.Provjerava događaj |
| | | | 3.Odobrava događaj |
| 4.Kreira događaj | 5.Daje obrazac za | | |

| | | | |
|--|---|--|---|
| | prijavu novog događaja | | |
| 6.Popunjava podatke za događaj-ime,lokacija,vrijeme, kapacitet | 7.Validacija unešenih podataka | | |
| | 8.Događaj stavljen na rezervaciju korisnicima | | |
| 9.Pregledanje stanja događaja | | | 10.Brisanje događaja nakon završetka događaja |

Alternativni tok 1: Događaj nije primjeren aplikaciji u koju se želi postaviti

Preduvjeti: Ne zadovoljavanje tačke 2.glavnog toka

Tok događaja:

| Korisnik | EventManager | Ustanova | Administrator |
|----------|--------------|----------|--|
| | | | 1.Provjeravanje zahtjeva za postavljanjem događaja |
| | | | 2.Odbijanje događaja |
| | | | 3.Obavještavanje korisnika |

Alternativni tok 2: Brisanje događaja nakon kreiranja(nakon 8.tačke glavnog toka)

Preduvjeti: Događaj je kreiran I dat korisnicima na rezervaciju

Tok događaja:

| Korisnik | EventManager | Ustanova | Administrator |
|-------------------------------------|--|----------|---------------------|
| 1.Bira opciju za brisanjem događaja | 2.Obavještava korisnika kako će izgubiti sve podatke i rezervacije | | 3.Briše događaj |
| | 3.Obavještava sve korisnike koji su rezervisali kartu da je događaj poništen | | |
| | | | 5.Briše sve podatke |

| | | | |
|--|--|--|--------------------------------|
| | | | smještene u arhivi rezervacija |
| | 6.Ažuriranje stanja aplikacije | | |
| | 7.Nudi mogućnost ponovnog kreiranja događaja | | |

9. davanje VIP statusa korisnicima od strane ustanove

Opis: Korisnik može postati VIP korisnik ukoliko ustanova tako odredi.Korisnik nakon što ostvari neki uslov koji odgovara politici ustanove,tada može postati VIP.Ustanova listanjem arhive može doći do informacija koje je zanimaju,dati VIP status te obavijestiti korisnika da je postao VIP.U biti,ovo je samo grubi prikaz šta se dešava ako ustanova želi da da nekom korisniku VIP status.Ustanova može da odabere koje od 3 mogućnosti za VIP će biti iskorištene,međutim sve tri se svode na isto-davanje popusta,samo različit broj popusta,a sva ostala procedura je ista za sve VIP korisnike.

Glavni tok: Korisnik dobija VIP status

Preduvjeti: Korisnik ima već napravljen korisnički račun i ostvarene ciljeve koji odgovaraju politici ustanove

Posljedice: Korisnik postaje VIP,te ima pravo na dodatne popuste

Tok događaja:

| Korisnik-obični profil | EventManager | Ustanova | Administrator |
|---|-------------------------------|---|---------------|
| | | 1.Omogućen pregled arhive rezervacija | |
| | | 2.Odabir korisnika koji zadovoljavaju uslov | |
| | | 3.Davanje VIP statusa | |
| | | 4.Obavještavanje korisnika da je postao VIP | |
| | 5.Slanje obavijesti korisniku | | |
| 6.Ima mogućnost da bira popuste u sljedećim rezervacijama | | | |

10. Odbijena rezervacija od strane ustanove

Opis: Korisnik Web-interfejsa pregleda stranicu i odučuje da rezerviše određen broj karata za neki događaj koji će biti održan u nekoj ustanovi, međutim broj mjesta na tom događaju je popunjeno, tako da ustanova vraća obavještenje da rezervacija nije uspjela.

Glavni tok: Odbijen zahtjev korisnika za rezervaciju karata

Preduvjeti: Korisnik ima dostupan Web interfejs kao i registrovan korisnički profil

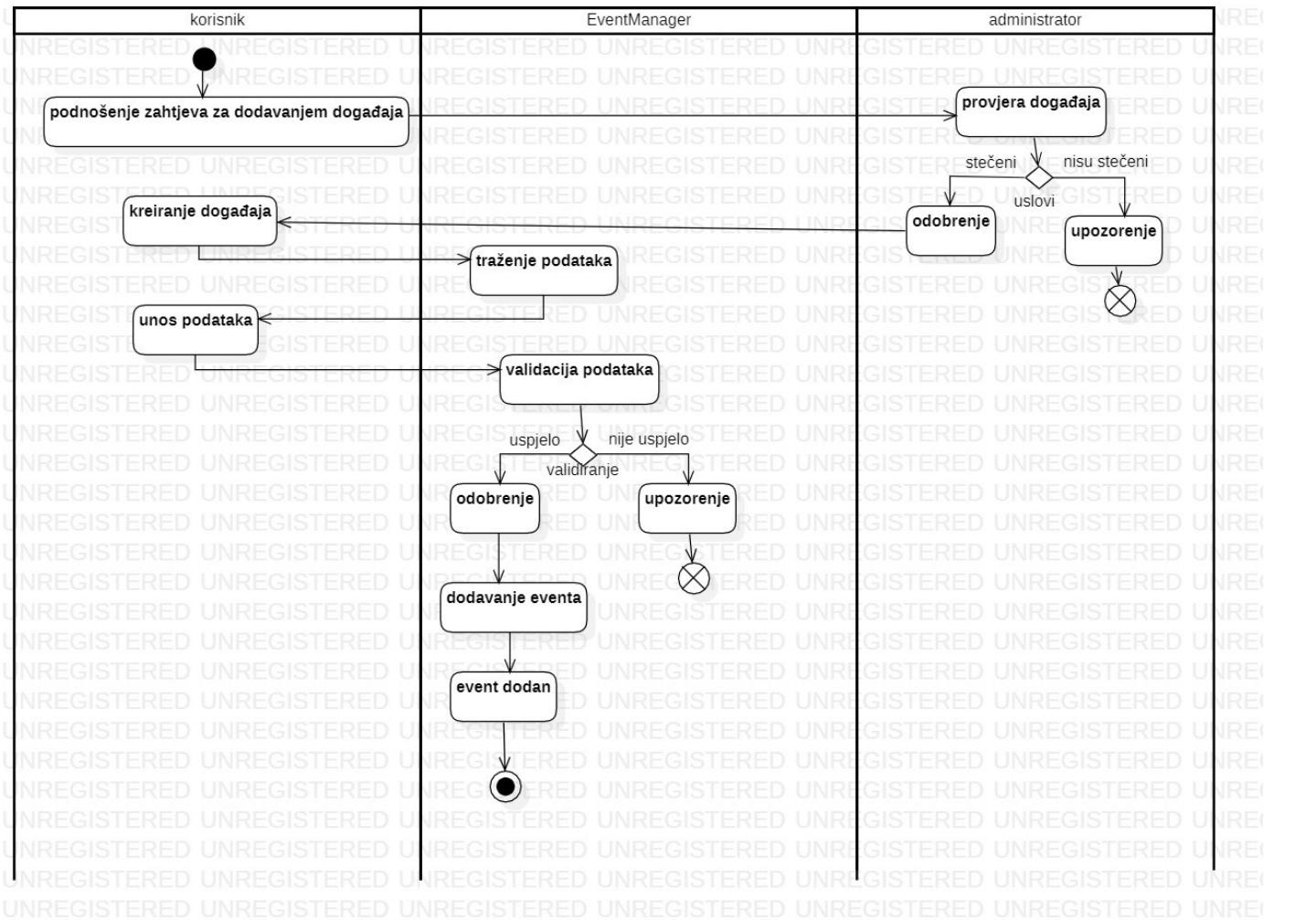
Posljedice: Korisnik dobiva obavještenje o neuspjeloj kupovini karata

Tok događaja:

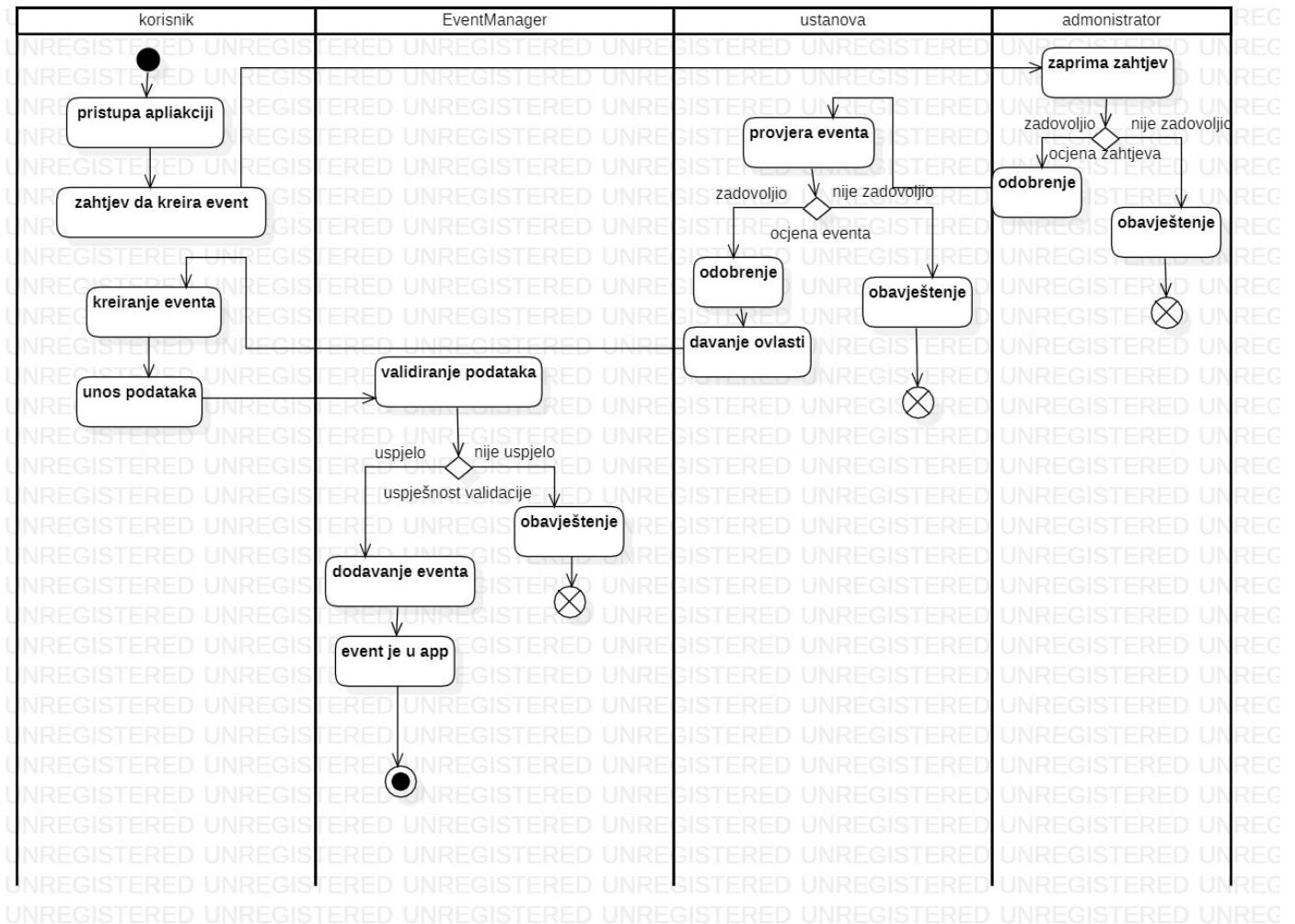
| KORISNIK | SISTEM | USTANOVA | ADMIN |
|---------------------------------------|---|--|-------|
| 1. Pristupanje interfejsu za kupovinu | 2. Prikaz aktuelnih događaja i omogućavanje izbora | | |
| 3. Izbor događaja i zaključivanje | | | |
| 4. Popunjavanje obrasca | | | |
| 5. Arhiviranje | 6. Validacija narudžbe | 7. Provjera stanja | |
| | | 8. Slanje obavještenja korisniku o odbijanju rezervacije | |
| | 9. Prikaz obavještenja od strane ustanove | | |
| | 10. Ponovni prikaz aktuelnih događaja i omogućavanje izbora | | |

Dijagrami aktivnosti kreirani na osnovu scenarija

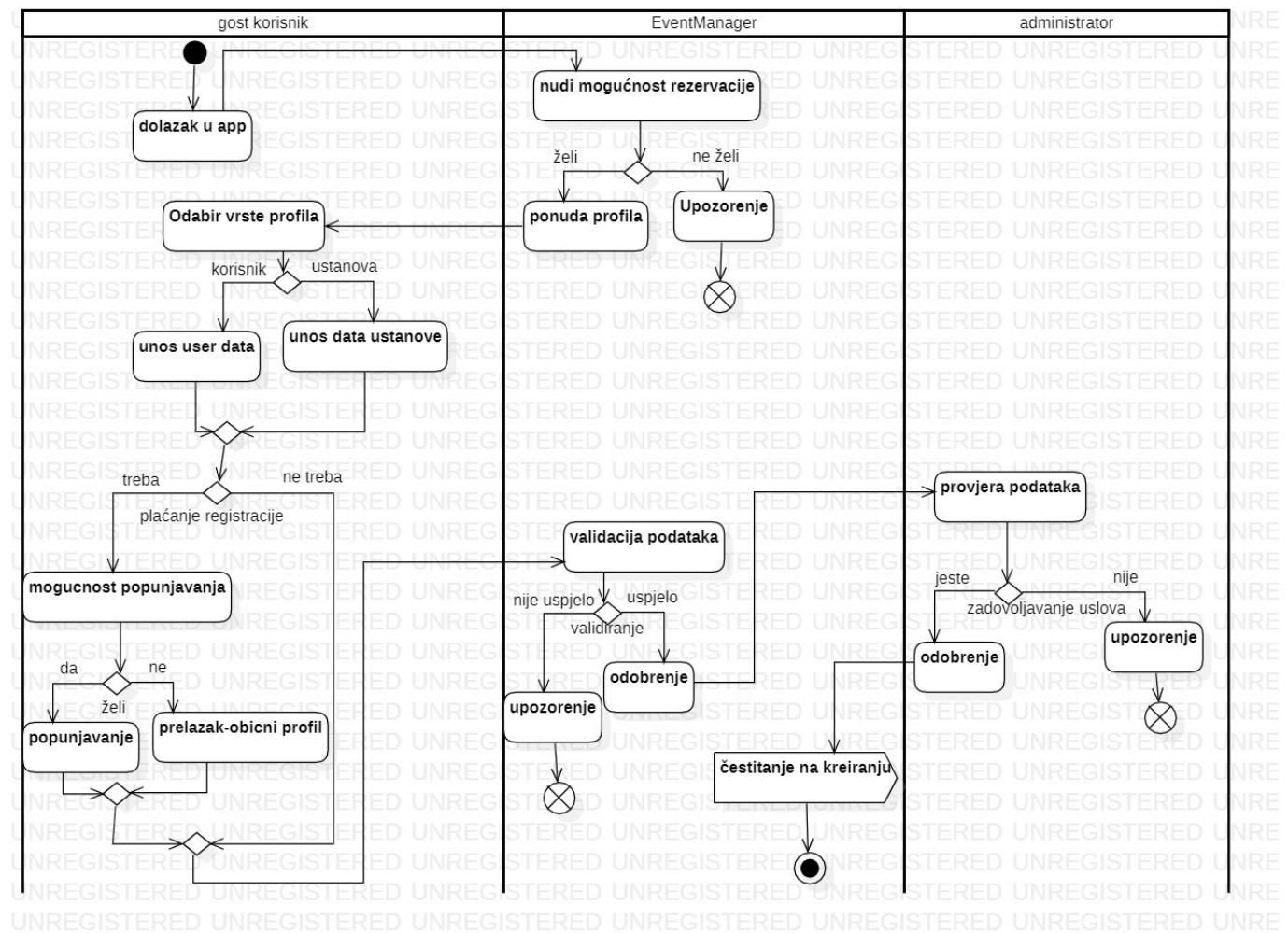
1. Korisnik – dodavanje eventa



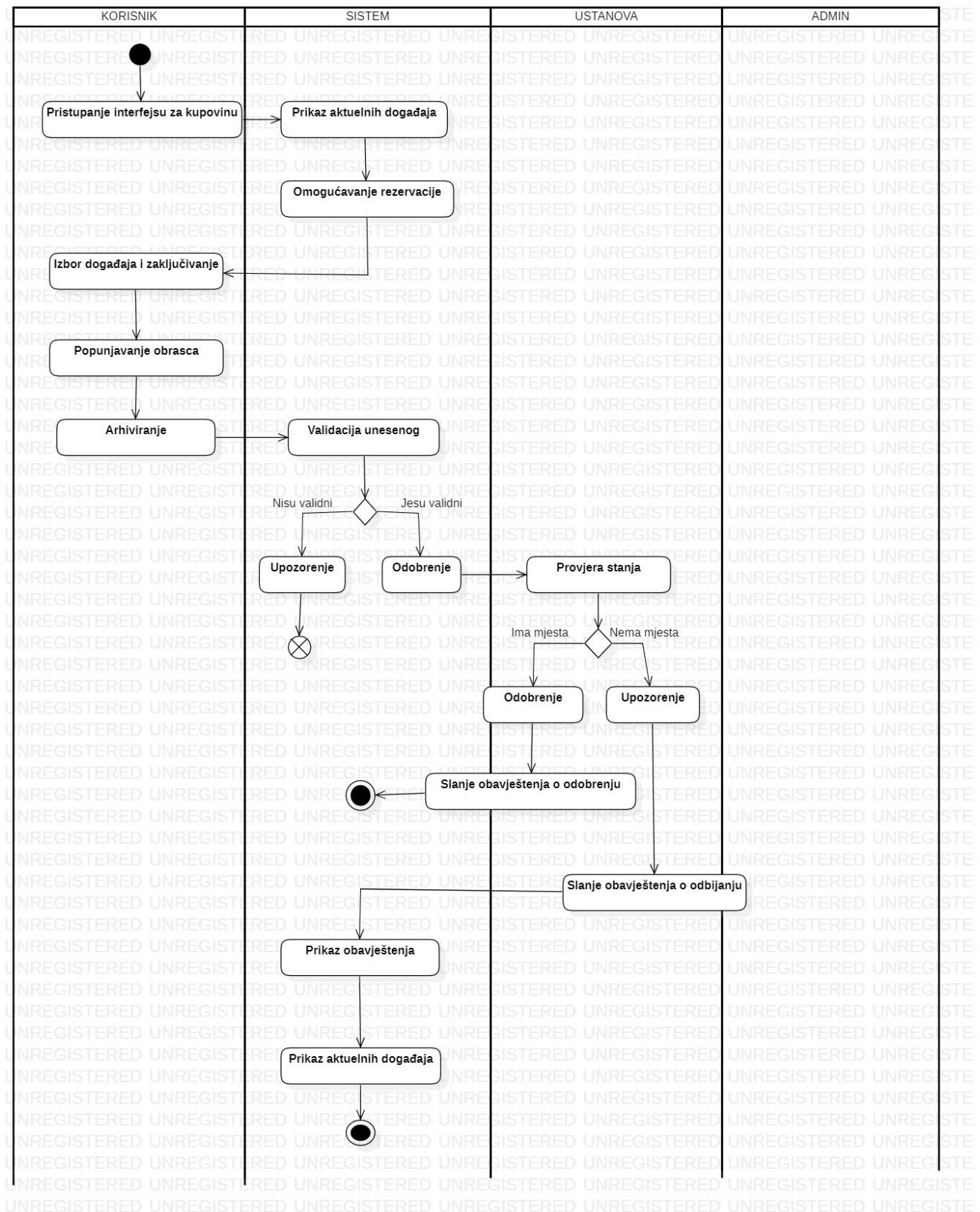
2.Korisnik - dodavanje eventa u ime ustanove



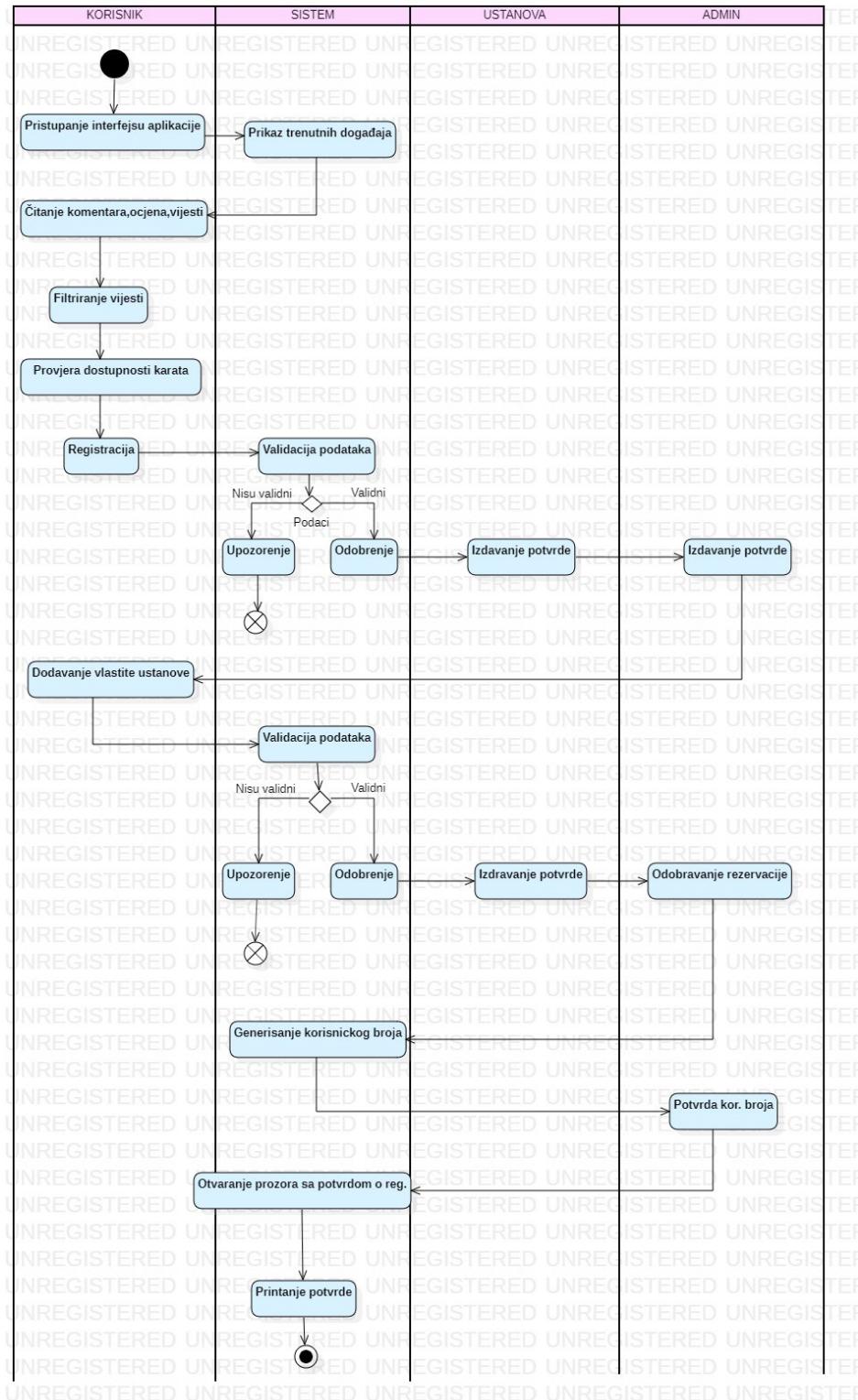
3. Kreiranje VIP profila



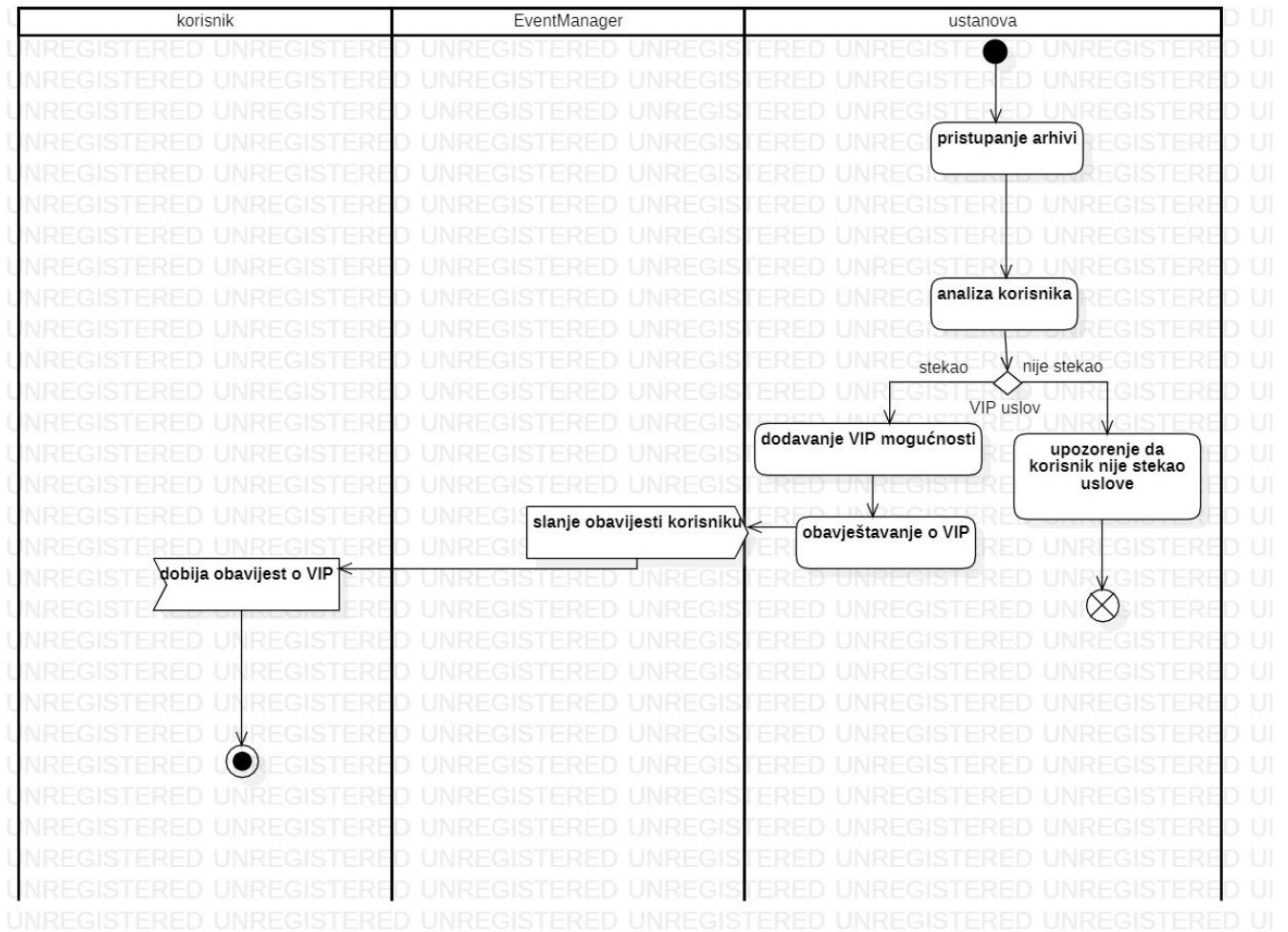
4. Odbijanje od strane ustanove



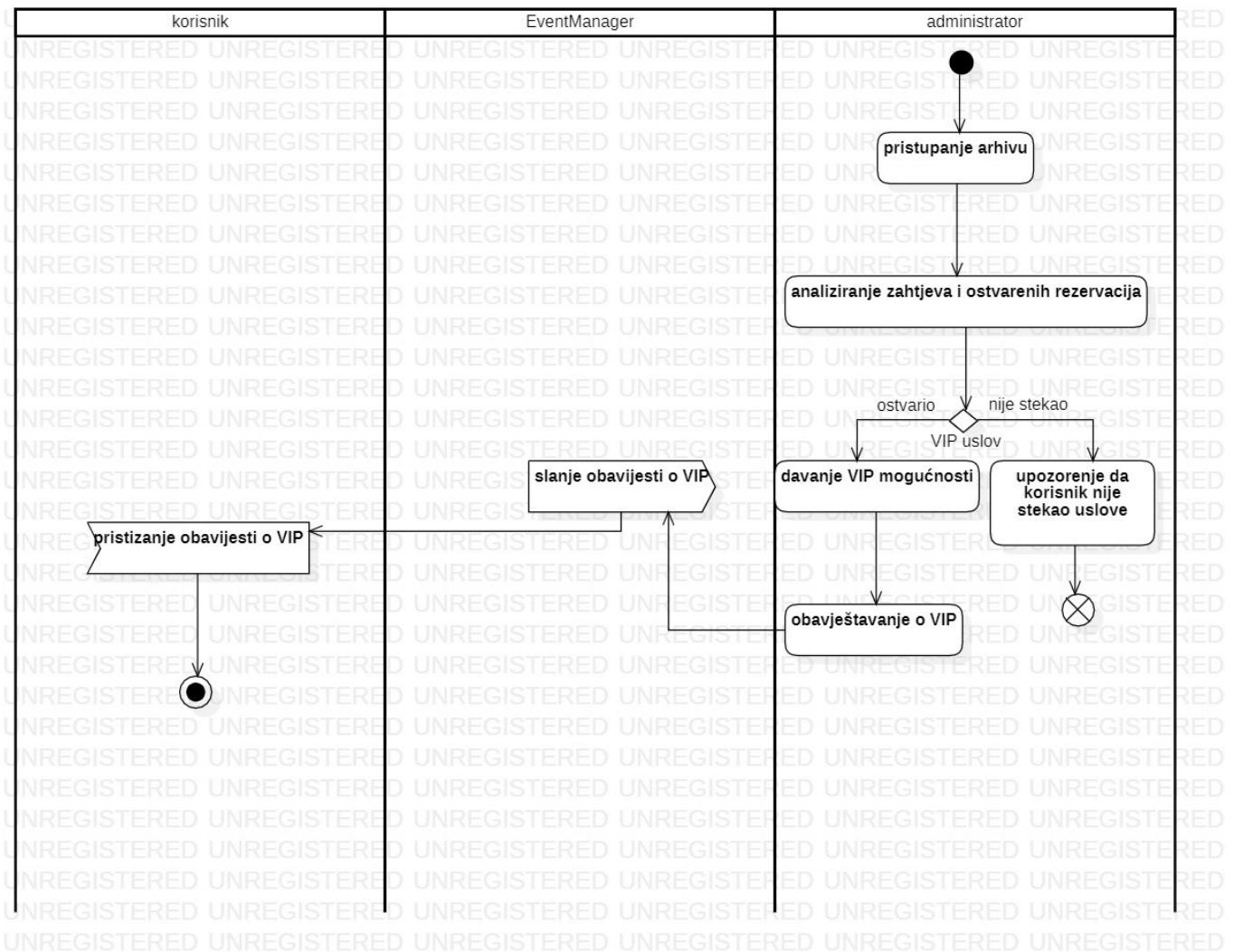
5. Prvo prijavljivanje



6. Ustanova daje VIP



7. Admin dodjeljuje VIP



Prototipovi formi sistema

Početna stranica

The screenshot shows the homepage of the EventManager website. At the top, there is a header bar with navigation icons (back, forward, search, refresh) and a URL bar showing <https://www.EventManager.ba/pocetna>. Below the header is a menu bar with links for Početna, Eventi, Registracija, and Admin. To the right of the menu is a search bar and a help icon. The main content area features a large title "EventManager". Below the title is a welcome message: "Dobro došli u EventManager! Na jednom mjestu možete naći kulturne događaje, novosti iz kulturnih ustanova te informacije o kulturnim ustanovama. Ukoliko već niste, savjetujemo Vam da se registrujete zbog ostvarivanja pogodnosti i lakše rezervacije." On the left, there is a box titled "Najnoviji eventi:" containing five items: "event1", "event2", "event3", "event4", and "event5". In the center, there is a box titled "Novosti iz ustanova:" containing five items: "novost1", "novost2", "novost3", "novost4", and "novost5". To the right, there is a calendar for April 2020. At the bottom, there is a footer bar with links for Eventi, Ustanove, O nama, Kontakt, and social media icons for Facebook, Instagram, Google+, and LinkedIn.

Registracija novog korisnika

The screenshot shows the "Registracija novog korisnika" (New User Registration) page. At the top, there is a header bar with navigation icons and a URL bar showing <https://www.EventManager.ba/registracija>. Below the header is a menu bar with links for Početna, Eventi, Registracija, and Admin. To the right of the menu is a search bar and a help icon. The main content area features a large title "Registracija novog korisnika". Below the title, there is a message: "Drago nam je što želite biti dio našeg društva!" and "Odaberite vrstu profila:". There is a large red box containing three profile options: "Korisnički profil (besplatni)" with a "FREE" button, "Korisnički VIP profil" with a "VIP" button, and "Profil za ustanovu" with a "USTANOVNI" button. At the bottom right, there is a "Nazad" (Back) button.

Registracija VIP korisnika

EventManager
Početna Eventi Registracija Admin search ?

Registracija VIP korisnika

*Ime:

*Prezime:

*Spol: Muško Žensko

*E-mail:

*Datum rođenja: 3 3 3

*Ime koje će biti korišteno u aplikaciji:

*Lozinka:

*Ponoviti lozinku:

*Broj kreditne kartice:

Slažem se sa uslovima korištenja i ponašanja

Kreiraj Odustani

Registracija nove ustanove

EventManager
Početna Eventi Registracija Admin search ?

Registracija nove ustanove

Noziv ustanove:

Radno vrijeme: : - :

Lokacija:

Broj telefona:

Fax:

E-mail:

Opis:

Lozinka:

Ponoviti lozinku:

Slažem se sa uslovima korištenja i pravilima ponašanja

Locirajte ustanovu

Dodavanje novog eventa-ustanova

EventManager

[Početna](#) [Eventi](#) [Uređivanje profila](#) [Obavijesti](#) [Rezervacije](#) [Admin](#)

https://wwwEventManagerba/ustanova

search ?

Dodavanje novog eventa

Ime eventa:

Kapacitet:

Vrste karata: Obična VIP Fan Studentska

Kontakt:

Datum eventa:

Vrijeme eventa: :

Datum do kojeg je moguće rezervisati:

Vrijeme do kojeg je moguće rezervisati: :

Omogućiti printanje karte:

Omogućiti online plaćanje:

Podijeliti event:

Unesite lokaciju i označite na karti:

Sačuvaj Odustani

VIP rezervacija karte

EventManager

[Početna](#) [Eventi](#) [Uređivanje profila](#) [Obavijesti](#) [Wish lista](#) [Rezervacije](#) [Admin](#)

https://wwwEventManagerba/rezervacijaVIP

search ?

Rezervacija karata - VIP korisnik

Molimo unesite sljedeće podatke neophodne za rezervaciju na događaj:

*Ime:

*Prezime:

*E-mail:

*Kontakt telefon:

*Vrsta karte: Obična VIP Fan Studentska

*Broj karata za rezervaciju:

Do li želite iskoristiti popust? Da Ne

Zelim potvrdu rezervacije na e-mail

Zelim printati potvrdu rezervacije

Rezerviši Odustani

Problem kod VIP rezervacije

EventManager

[Početna](#) [Eventi](#) [Uređivanje profila](#) [Obavijesti](#) [Wish lista](#) [Rezervacije](#) [Admin](#) [?](#)

Rezervacija karata - VIP korisnik

Molimo,unesite sljedeće podatke neophodne za rezervaciju na događaj:

*Ime:
*Prezime:
*E-mail:
*Kontakt telefona:
*Vrsta kartice:
*Broj kartice:
*Broj kartica:

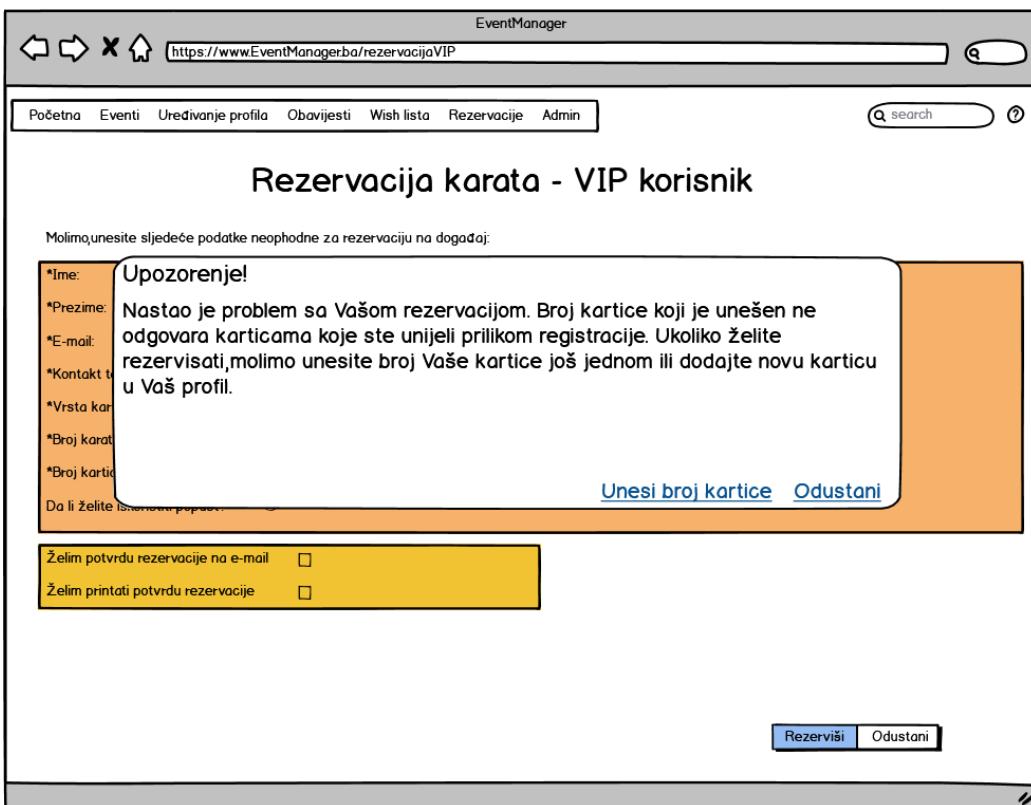
Upozorenje!
Nastao je problem sa Vašom rezervacijom. Broj kartice koji je unešen ne odgovara karticama koje ste unijeli prilikom registracije. Ukoliko želite rezervisati,molimo unesite broj Vaše kartice još jednom ili dodajte novu karticu u Vaš profil.

Da li želite da izaberete pop up ovaj?

[Unesi broj kartice](#) [Odustani](#)

Želim potvrdu rezervacije na e-mail
Želim printati potvrdu rezervacije

[Rezerviši](#) [Odustani](#)



Pregled WishListe

EventManager

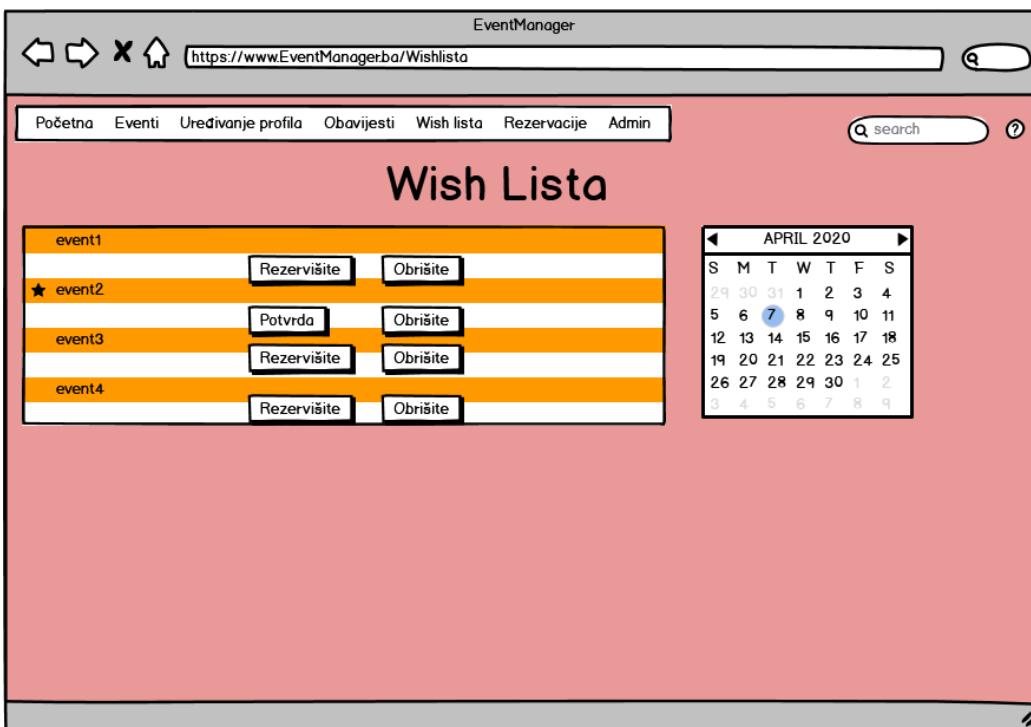
[Početna](#) [Eventi](#) [Uređivanje profila](#) [Obavijesti](#) [Wish lista](#) [Rezervacije](#) [Admin](#) [?](#)

Wish Lista

| | | |
|----------|-----------------------------|--------------------------|
| event1 | Rezervišite | Obrišite |
| ★ event2 | Potvrda | Obrišite |
| event3 | Rezervišite | Obrišite |
| event4 | Rezervišite | Obrišite |

APRIL 2020

| S | M | T | W | T | F | S |
|----|----|----|----|----|----|----|
| 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 | 9 |



Pregledanje rezervacija od strane admina

Pregled rezervacija

Korisnik Filtriraj

1.Suljo Suljić, 29.10.2019,non VIP
 2.Suljo Suljić, 5.11.2019,non VIP
 3.Suljo Suljić, 11.11.2019,non VIP
 4.Suljo Suljić, 16.11.2019,non VIP
 ----Korisnik Suljo Suljić ima pravo na VIP-ostvario je 3 rezervacije u jednom mjesecu.
 Dozvoliti prelazak na VIP?

5.Mujo Mujić, 10.10.2019,VIP

Sačuvaj

Ok Odustani

Dodavanje događaja od strane korisnika koji nije ustanova

Dodavanje eventa

Kao korisnik, možete tražiti zahtjev da dodate događaj. Naprije toga, molimo Vas da unesete par osnovnih informacija o eventu kojeg želite da dajete u bazu podataka.

Ime eventa:

Veza za ustanovu: Ustanove

Datum održavanja: /

Vrijeme održavanja: :

Kapacitet:

Opis:

Tražite dozvolu Nazad

Odobrenje eventa od strane admina

EventManager
<https://wwwEventManager.ba/AdminOdobrenjeEvent>

Početna Eventi Obavijesti Prijava problema search ?

Pregledaj korisnike
Pregledaj evenete
Pregledaj ustanove
Pregledaj komentare
Pregledaj rezervacije
Pregledaj transakcije
Pregledaj zahtjeve za dodavanjem događaja
Izađi

Odobrenje eventa

Korisnik Suljo Suljić je podnio zahtjev za dodavanjem događaja. Ovdje su unešeni podaci:

Ime eventa:

Vezan za ustanovu: Ustanove

Datum održavanja: /

Vrijeme održavanja: :

Kapacitet:

Opis:

Označi event kao neprimjeran Odobri event

Sačuvaj

Ok Odustani

Davanje odobrenja korisniku od strane ustanove

EventManager
<https://wwwEventManager.ba/ustanovaDavanjeOvlasti>

Početna Eventi Uređivanje profila Obavijesti Rezervacije Admin search ?

Davanje odobrenja

Korisnik Suljo Suljić je podnio zahtjev da u Vaše ime objavi event. Ispod su navedene karakteristike zatraženog eventa

Ime eventa:

Vezan za ustanovu: Ustanova

Datum održavanja: /

Vrijeme održavanja: :

Kapacitet:

Opis:

Označiti event kao neprimjeran:
 Označiti event kao primjeran:
 Dati korisniku ovlasti samo za ovaj event:
 Dati korisniku ovlasti za ustanovu:

Sačuvaj

Ok Odustani

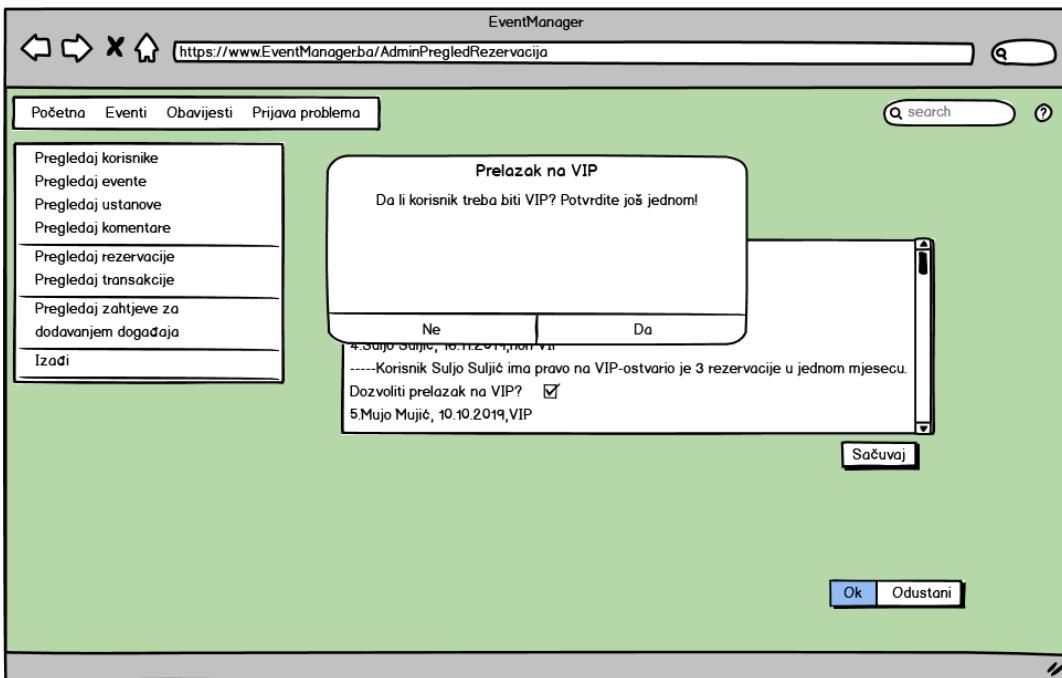
Prikaz početne stranice VIP profila

The screenshot shows the EventManager website with the URL <https://www.EventManager.ba/profil>. The page title is "EventManager". The navigation menu includes "Početni", "Eventi", "Uređivanje profila", "Obavijesti", "Wish lista", "Rezervacije", and "Admin". A search bar and a help icon are also present. The user's name "Suljo Suljić" is displayed at the top right, along with a "Odjava" button. On the left, there is a sidebar titled "Osnovni podaci" containing fields for "Ime", "Prezime", "Email", "Datum rođenja", and "Username". Below this is a "VIP profil" section with a star icon. The main content area is titled "Novosti" and features a filter section with "Najnovije", "Ustanove", and "Prikaži sve" buttons. It lists four news items: "Novost1", "Novost2", "Novost3", and "Event1" through "Event4", each with "Dodajte u wish listu" and "Rezervišite" buttons. To the right, a sidebar titled "Obavještenja" lists "obavijest1", "obavijest2", and "obavijest3".

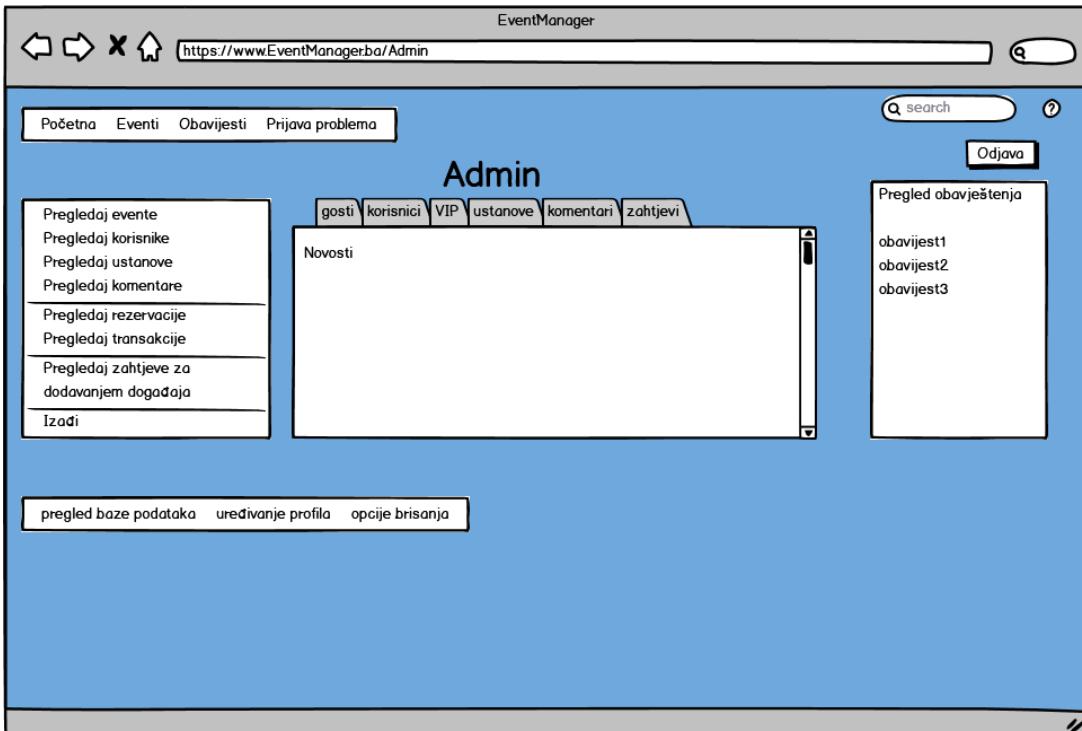
Obavijest o sticanju VIP

The screenshot shows the same EventManager website interface. A modal window titled "Novosti" appears, displaying a message: "POSTALI STE VIP!" followed by "Čestitamo Vam na ostvarenim rezervacijama u proteklom periodu, te ste zaslužili VIP status. Iskoristite Vaše popuste već danas!" Below the message are three yellow stars. At the bottom of the modal are buttons for "Idi na Evente" and "Zatvori". The rest of the page structure remains the same, including the sidebar with "Osnovni podaci" and the "Obavještenja" sidebar.

Potvrda o prelasku na VIP – admin side



Pregled amin profila



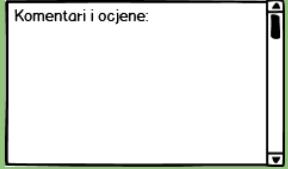
Pregled eventa od strane logovanog korisnika

EventManager
[Početna](#) [Eventi](#) [Uređivanje profila](#) [Obavijesti](#) [Wish lista](#) [Rezervacije](#) [Admin](#) [search](#) [?](#)

EVENT: <IME_EVENTA>

Lokacija: 

| APRIL 2020 | | | | | | |
|------------|----|----|----|----|----|----|
| S | M | T | W | T | F | S |
| 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 | 9 |

Komentari i ocjene: 

Ostavite komentar:

Ustanova 

Naziv:
Adresa:
Kontakt:

[Rezervište](#) [Dodata u wish listu](#) [Nazad](#)

Pregled eventa od strane ustanove

EventManager
[Početna](#) [Eventi](#) [Uređivanje profila](#) [Obavijesti](#) [Rezervacije](#) [Admin](#) [search](#) [?](#)

EVENT: <IME_EVENTA>

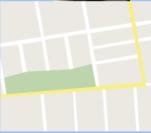
Event objavljen dana: Izmjena: / / 

Event ostaje aktivan do: Izmjena: / / 

Kapacitet: Izmjena: 3 

Vrsta karte: Obična VIP Fan Studentska

Cijena: Izmjena: 3 

Lokacija: Izmjena: 

[Sačuvati](#)

Komentari 

Dodataj komentar:

[Sačuvati](#)

Izvještaj o rezervacijama:

Planirano rezervacija:
 Planirana dobit:
 Ostvareno rezervacija:
 Ostvareno dobit:
 Broj običnih rezervacija:
 Broj VIP rezervacija:
 Broj studentskih rezervacija:
 Broj Fan rezervacija:
 Broj rezervacija sa popustom:
 Broj rezervacija bez popusta:

Ukupno dobiti: 

[Sačuvati](#) [Nazad](#)

Prikaz profila ustanove

The screenshot shows the 'EventManager' application interface. At the top, there is a navigation bar with links: Početna, Eventi, Uređivanje profila, Obavijesti, Rezervacije, Admin, and a search bar. Below the navigation bar, the title is '<NAZIV USTANOVE>' (Organization Name). A sidebar on the left contains a placeholder image of a person, a 'Ustanova' button, and a 'Osnovni podaci' section with fields for Naziv, Lokacija, and Kontakt. The main content area has a 'Filter' section with options for Datum and Naziv. To the right, a sidebar titled 'Obavještenja' lists several notifications: obavijest1, obavijest2, obavijest3, and others. A 'Odjava' (Logout) button is located at the top right of the main content area.

Kreiranje profila

The screenshot shows a 'Create user profile' form. At the top, it says 'Kreiranje korisničkog profila' and 'Dobro došli !'. On the left, there is a placeholder image for a profile picture with the text 'Izaberite sliku profila'. Below it, there is a red box for writing about oneself with the text 'Recite nam nešto o sebi:' followed by a wavy black line. In the center, there are two stacked boxes for general information. The top box contains fields for Ime (Name), Prezime (Last Name), Korisničko ime (Username), and Broj telefona (Phone number). The bottom box contains fields for Adresa stanovanja (Address), Broj kartice (Card number), Lozinka (Password), and Potvrda lozinke (Confirm password). Above the bottom box, a message says 'Vaše informacije su uspješno primljene !' and 'Vaš korisnički ID: 77845236985'. To the right, there is a map with a green polygon and a yellow line, with the text 'Odaberite adresu stanovanja na mapi'. At the bottom, there are social media icons for Facebook, Snapchat, and Instagram, with the text 'Ne zaboravite nas zapraviti na društvenim mrežama'. A 'POTVRDI' (Confirm) button is at the very bottom.

Kupovina karte običnog korisnika

A Web Page
https://

Kupovina karte za običnog korisnika

 Nemate korisnički račun? Registrirajte se klikom na ikonu.

*Ime:

*Prezime:

Korisničko ime ili ID:

*Lozinka:

e-mail:

Datum rođenja: / / 

*Naziv događaja:

*Korisnički ID:

*Broj kreditne kartice: 

Složem se sa odredbama korištenja ovih usluga


APRIL 2020
S M T W T F S
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 31
3 4 5 6 7 8 9

Pratite nas i na društvenim mrežama

Odabir razloga za brisanje korisničkog računa

< Početna stranica

Postavke profila

PROMIJENI LOZINKU:



[Promjeni sliku profila > ...](#)

O meni:



[Uredi tekst o sebi > ...](#)

Some text

UREDI OPŠTE INFO:

| | | |
|--------------------|---|-----------------------|
| Ime: | <input type="text"/> | Uredi |
| Prezime: | <input type="text"/> | Uredi |
| e-mail: | <input type="text"/> | Uredi |
| Broj telefona: | <input type="text"/> <input type="text"/> | Uredi |
| Korisničko ime: | <input type="text"/> | Uredi |
| Datum rođenja: | <input type="date"/> | Uredi |
| Adresa stanovanja: | <input type="text"/> | Uredi |
| Broj kartice: | <input type="text"/> <input type="text"/> | Uredi |

OBRIŠI RAČUN:

| | |
|-------------------|--------------------------|
| Ime: | <input type="text"/> |
| Prezime: | <input type="text"/> |
| *Korisnički ID: | <input type="text"/> |
| *Lozinka: | <input type="password"/> |
| *Potvrda lozinke: | <input type="password"/> |

Pristojem na to da moj korisnički profil bude trajno obrisan, uključujući sve aktivnosti i podatke unesene do ovog trenutka kroz ovu aplikaciju

Razlog brisanja profila?

Premašo sadržaja

Nedostatak vlastitog vremena

Visoke cijene

Loš interfejs

Ostalo

POTVRDI

Obavijest da je profil obrisan

The screenshot shows a web interface titled "Postavke profila". On the left, there's a placeholder for a user profile picture with the text "Promjeni sliku profila >...". Below it is a box for editing text about oneself, with the placeholder "O meni:" and the text "Uredi tekst o sebi >...". In the center, there are three main sections: "PROMIJEANI LOZINKU:" (with fields for Old password, New password, and Confirmation), "OBRIŠI RAČUN:" (with fields for Name, Surname, User ID, Password, and Confirmation), and "UREDI OPŠTE INFO:" (with fields for Name, Surname, Email, Phone number, Nickname, Date of birth, Address, and Credit card number). Each section has a "POTVRDI" button. A large red button at the bottom center displays the message "Vaš profil je uspješno obrisan" (Your profile has been successfully deleted) with a "Potezna stranica" link below it. At the very bottom, there's a feedback section with checkboxes for reporting issues like broken links or interface problems.

Odbijanje rezervacije

A Web Page
 https://

Kupovina karte za običnog korisnika

Nemate korisnički račun? Registrirajte se klikom na ikonu

| | | |
|-----------------------------|----------------------------|--|
| *Ime: [redacted] | *Prezime: [redacted] | Korisničko ime ili ID: [redacted] |
| *Lozinka: [redacted] | e-mail: [redacted] | Datum rođenja: 3 / / [calendar icon] |
| *Naziv događaja: [redacted] | *Korisnički ID: [redacted] | *Broj kreditne kartice: [redacted] VISA [redacted] |

Upozorenje
Nažalost, trenutno da odabranom događaju nije dostupan broj traženih karata.

[Vrati](#) [Lista događaja](#)

Prijedložite nas i na društvenim mrežama

Slažem se sa odredbama korištenja ovih usluga

Postavke korisničkog profila-brisanje

< Početna stranica

Postavke profila

 PROMIJENI LOZINKU:

| |
|----------------------------------|
| Stara lozinka: [redacted] |
| Nova lozinka: [redacted] |
| Potvrda nove lozинke: [redacted] |

[Promijeni sliku profila > ...](#)

 O meni:
[redacted]
[redacted]
[redacted]

[Uredi tekst o sebi > ...](#)

 UREDI OPŠTE INFO:

| |
|--|
| Ime: [redacted] Uredi |
| Prezime: [redacted] Uredi |
| e-mail: [redacted] Uredi |
| Broj telefona: [redacted] Uredi |
| Korisničko ime: [redacted] Uredi |
| Datum rođenja: 3 / / [calendar icon] Uredi |
| Adresa stanovanja: [redacted] Uredi |
| Broj kartice: [redacted] Uredi |

 OBRIŠI RAČUN:

| |
|------------------------------|
| Ime: [redacted] |
| Prezime: [redacted] |
| *Korisnički ID: [redacted] |
| *Lozinka: [redacted] |
| *Potvrda lozinke: [redacted] |

Pristojem na to da moj korisnički profil bude trajno obrisan, uključujući sve aktivnosti i podatke unesene do ovog trenutka kroz ovu aplikaciju

[POTVRDI](#)

Potvrda o uspješnoj kupovini karata

A Web Page
https://

Kupovina karte za običnog korisnika

Nemate korisnički račun? Registrujte se klikom na ikonu

*Ime: [input]
*Prezime: [input]
Korisničko ime ili ID: [input]
*Lozinka: [input]
e-mail: [input]
Datum rođenja: [input] / [input] / [input] [calendar]
*Naziv događaja: [input]
*Korisnički ID: [input]
*Broj kreditne kartice: [input] [VISA]

Obavijest
Kupovina karata je uspješno obavljenja.
Hvala na povjerenju.

[Vrati](#) [Printaj rezervaciju](#)

| | | | | | | |
|----|----|----|----|----|----|----|
| S | M | T | W | T | F | S |
| 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 | 9 |

Pratite nas i na društvenim mrežama

Slažem se sa odredbama korištenja ovih usluga

Prozor sa potvrdom o uspješnom registrovanju

A Web Page
https://

Kreiranje korisničkog profila

Dobro došli!

Izaberite profil: [input]
Recite nam nešto o mjeri: [input]
OPŠTE INFORMACIJE:

Ime: [input]
Datum rođenja: [input] / [input] / [input] [calendar]
Adresa stanovanja: [input] [Odaberi na mapi](#)
Broj kartice: [input]
Lozinka: [input]
Potvrda lozinke: [input]

USPJEŠNO STE REGISTROVALI PROFIL

[POTVRDI](#)

Ne zaboravite nos zapratiti na društvenim mrežama

Registracija novog korisnika

A Web Page
<https://>

Kreiranje korisničkog profila

Dobro došli !

Izaberite sliku profila

Izaberite adresu stanovanja na mapi

Ne zaboravite nas zapratiti na društvenim mrežama

POTVRDI!

OPŠTE INFORMACIJE:

| | |
|--------------------|--|
| Ime: | <input type="text"/> |
| Prezime: | <input type="text"/> |
| Korisničko ime: | <input type="text"/> |
| Broj telefona: | <input type="text"/> |
| e-mail: | <input type="text"/> |
| Datum rođenja: | <input type="text"/> 3 <input type="button" value="++"/> <input type="text"/> / <input type="button" value="--"/> <input type="text"/> <input type="button" value="kalendar"/> |
| Adresa stanovanja: | <input type="text"/> Odaberi na mapi |
| Broj kartice: | <input type="text"/> |
| Lozinka: | <input type="text"/> |
| Potvrda lozinke: | <input type="text"/> |

Inicijalni opis klasa za sistem (koncept)

EventManager – opis sistema

Klase

Klasa Gost

Atributi

BrojacPosjete-statički integer

ŽeliRegistraciju-boolean

Metode

Konstruktor/getter/setter

PromijeniStatusRegistracije():Boolean

Klasa Korisnik-apstraktna,bazna klasa

Atributi

Username-String

Password-String

Email-String

TrenutnoAktivan-Boolean

Adresa-String

ListaObavjestenja-Obavijest

IDKorisnika-statički integer

Metode

Konstruktori/getteri/setteri

promjeniUsername(noviUsername:String): void

promjeniPassword(noviPassword:String): void

promjeniEmail(noviEmail:String): void

prijavaSeNaApp(name:String, pass:String): void

odjavaSeSaApp(korisnik:Korisnik): void

izlistajObavjestenja(): void

obrisiObavijest(): void

dajID(): Integer

Klasa FizickoLice-naslijedena iz klase Korisnik

Atributi

Ime:String

Prezime:String

BrojKartice:String

osobaOdgovornaUlmeUstanove: Boolean = false

datumRodjenja: Date

tipFizickogLica: TipFizickogLica

listaTransakcija: list<Transakcija>

wishListaDogadjaja: list<Dogadjaj>

listaZavrsenihRezervacija: list<Rezervacija>

stanjeRacuna: Double

listaRezervacija: list<Rezervacija>
popust: Integer = 0
spol: String

Metode

Konstruktori/getteri/setteri

izlistajListuTransakcija(): void
izlistajWishListu(): void
izlistajListuZavrsenihTransakcija(): void
promijeniIme(ime:String): void
promijeniPrezime(prezime:String): void
promijeniBrojKartice(brojKartice:String): void
promijeniDatumRodjenja(datumRodjenja:Date): void
promijeniTipFizickogLica(tipFizickogLica:TipFizickogLica): void
obrisiDogadjajIzWishListe(dogadjaj:Dogadjaj): void
postaviNovuOdgovornost(): void
posaljiZahtjevZaKreiranjeDogadjaja(): void
izvrsiPlacanjeKarte(rezervacija:Rezervacija): Obavijest
izlistajObavjestenja(): void
dodajObavjestenjeKorisniku(obavijest:Obavijest): void
obrisiObavijest(obavijest:Obavijest): void
oznaciKaoNeprocitano(obavijest:Obavijest): void
promijeniUsername(noviUsername:String): void
promijeniPassword(noviPassword:String): void
promijeniEmail(noviEmail:String): void
prijavaSeNaApp(name:String, pass:String): void
odjaviSeSaApp(korisnik:Korisnik): void
dajID(): Integer

Klasa Ustanova-naslijeđena iz korisnika

Atributi

naziv: String
brojTelefona: String
brojRacunaUBanci: String
odgovorniKorisnik: FizickoLice
listaKreiranihDogadjaja: list<Dogadjaj>
arhivaDogadjaja: list<Dogadjaj>
stanjeUplata: Double

Metode

Konstruktori/getteri/setteri

promjeniNaziv(naziv:String): void
promjeniBrojTelefona(brojTelefona:String): void
promjeniBrojRacunaUBanci(brojRacunaUBanci:String): void
dodijeliOdgovornuOsobu(odgovorna:FizickoLice): Obavijest
izlistajListuKreiranihDogadjaja(): void
izlistajArhivuDogadjaja(): void
dajID(): Integer
staviDogadjajUArhivu(ID:Integer): void
dodijeliVIP(korisnik:FizickoLice): void
radSaBazomPodataka(baza:BazaPodataka): void
promjeniUsername(noviUsername:String): void
promjeniPassword(noviPassword:String): void
promjeniEmail(noviEmail:String): void
prijavaSeNaApp(name:String, pass:String): void
objaviSeSaApp(korisnik:Korisnik): void
izlistajObavjestenja(): void
obrisiObavijest(): void
dajID(): Integer
azurirajInfo(): void

Klasa Admin-naslijeđena iz korisnika

Atributi

Ime:String

Spol:String

Metode

Konstruktori/getteri/setteri

dajOdobrenjeZaDogadjaj(dogadjaj:Dogadjaj): Boolean

dodijeliVIPStatus(ID:Integer): void

zakljućiUplatu(ID:Integer): void

posaljiObavijest(): void

dajOdobrenjeZaKorisnika(ID:Integer): boolean

azurirajInformacijeOKorisnicima(): void

radSaBazom(baza:BazaPodataka): void

odobriRezervaciju(rezervacija:Rezervacija): void

promjeniUsername(noviUsername:String): void

promjeniPassword(noviPassword:String): void

promjeniEmail(noviEmail:String): void

prijaviSeNaApp(name:String, pass:String): void

odjaviSeSaApp(korisnik:Korisnik): void

izlistajObavijestenja(): void

obrisiObavijest(): void

dajID(): Integer

Klasa VIPKorisnik-naslijedena iz FizickoLice

Atributi

uplatioVIPClanstvo: Boolean = false

uplataZaClanstvo: Double

brojDostupnihPopusta: Integer = 0

daLiSulskoristeniPopusti: Boolean = false

VIPlstice: Date

Metode

Konstruktori/getteri/setteri

provjeralskoristenihPopusta(): Boolean
uplatiVIPClanstvo(clanstvo:Double): Boolean
printanjelzvjestaja(): void

Klasa Transakcija

Atributi

iznos: Double
korisnik: FizickoLice
nacinPlacanja: NacinPlacanja
IDTransakcije: Integer = 0

Metode

Konstruktori/getteri/setteri
obracunaj(iznos:Double, korisnik:Korisnik): Double
stornirajTransakciju(): Obavijest
printanjelzvjestaja(): void

Klasa Rezervacija

Atributi

dogadjaj: Dogadjaj
ustanovaDomacin: Ustanova
IDRezervacije: Integer = 0
napravioRezervaciju: FizickoLice
ukupnoZaUplatu: Double
brojRezervisanihMjesta: Integer = 1
datumRezervacije: Date
vrstaKarte: TipKarte
cijena: Double

Metode

Konstruktori/getteri/setteri
obrisiRezervaciju(korisnik:Korisnik): void

azurirajRezervaciju(): void
rezervisiDogadjaj(korisnik:Korisnik): void
dajIznosZaUplatu(): Double
obrisiRezervaciju(korisnik:Korisnik): void
printanjeJelvjestaja(): void

Klasa Dogadjaj

Atributi

nazivDogadjaja: String
mjestoDogadjaja: String
ustanovaDomacin: Ustanova
odgovornoFizickoLice: FizickoLice
kapacitet: Integer = 0
cijena: Double = 0
odobrenaStudentskaKarta: Boolean = false
odobrenaObicnaKarta: Boolean = false
odobrenaPenzionerskaKarta: Boolean = false
odobrenaVIPKarta: Boolean = false
krajnjeVrijemeZaRez: Date
listaRezervacija: list<Rezervacija>
brojZvjezdica: Integer = 0
datumDogadjaja: Date
listaKomentara: list<Komentar>
listaZvjezdica: list<Integer>

Metode

Konstruktori/getteri/setteri
postaviOdgovornoFizickoLice(odgovorno:FizickoLice): void
postaviOdobrenjeStudentske(odobrenje:Boolean): void
postaviOdobrenjeObicne(odobrenje:Boolean): void
postaviOdobrenjePenzionerske(odobrenje:Boolean): void

postaviOdobrenjeZaVIP(odobrenje:Boolean): void
postaviKrajnjeVrijemeZaRez(krajnjeVrijeme:Date): void
izlistajListuRez(): void
izracunajOcjenu(): Double
promjeniKapacitet(kapacitet:Integer): void
promjeniCijenu(cijena:Double): void
promjeniMjesto(mjesto:String): void
promjeniOdgovornoLice(odgovornoLice:FizickoLice): void
promjeniUstanovu(novaUstanova:Ustanova): void
promjeniNaziv(naziv:String): void
dodajRezervaciju(rezervacija:Rezervacija): Boolean
azurirajDogadjaj(): void
promjeniDatum(datum:Datum): void
izlistajKomentare(): void
dajBrojZvjezdica(): Integer

Klasa Obavijest

Atributi

tekst:String
uspješnoKreiranjeRacuna:Boolean
neuspješnoKreiranje:Boolean
obavijestOBrisanjuRacuna:Boolean
obavijestORezervaciji:Boolean
obavijestOBrisanjuRezervacije:Boolean
obavijestOKreiranjuDogadjaja:Boolean
obavijestOBrisanjuDogadjaja:Boolean
obavijestOProblemu:Boolean
obavijestOOdobrenjuFizickomLicu:Boolean
ObavijestOUplati:Boolean
obavijestOPrelaskuNaVip: Boolean = false

obavijestOStavljanjuUArhivu: Boolean = false

obavijestOAzuriranjuInfo: Boolean = false

datumSlanja: Date

primaocObavjestenja: Korisnik

Metode

Konstruktori/getteri/setteri

dajPrimaocaObavjestenja(): Korisnik

postaviDatumSlanja(datum:Date): void

dajDatumSlanja(): Date

postaviObavijestUspješnoKreiranje:Boolean

postaviObavijestNeuspješnoKreiranje:Boolean

postaviObavijestOBrisanju:Boolean

postaviObavijestORezervaciji:Boolean

postaviObavijestOBrisanjuRezervacije:Boolean

postaviObavijestOKreiranjuDogadjaja:Boolean

postaviObavijestOBrisanjuDogadjaja:Boolean

postaviObavijestOProblemu:Boolean

postaviObavijestOOdobrenjuFizickomLicu:Boolean

postaviObavijestOUplati:Boolean

posaljiObavijestOUspjesnomKreiranju:void

posaljiObavijestONeuspjesnomKreiranju:void

posaljiObavijestOBrisanju:void

posaljiObavijestORezervaciji:void

posaljiObavijestOKreiranjuDogadjaja:void

posaljiObavijestOBrisanjuDogadjaja:void

posaljiObavijestOProblemu:void

posaljiObavijestOOdobrenjuFizickomLicu:void

posaljiObavijestOBrisanjuRezervacije:void

posaljiObavijestOUplati:void

Klasa Komentar

Atributi

komentar: String

kometarOstavio: Korisnik

datumOstavljanja: Date

komentarNaDogadjaj: Dogadjaj

brojZvjezdica: Integer

neprimjereniKomentar: Boolean = false

Metode

Konstruktori/getteri/setteri

ostaviZvjezdice(brojZvjezdica:Integer)

ostaviKomentar(komentar:String)

oznaciKaoNeprimjeren()

Klasa EventManager

Atributi

gost: Gost

listaFizickihLica: list<FizickoLice>

listaUstanova: list<Ustanova>

listaAdmina: list<Admin>

listaDogadjaja: list<Dogadjaj>

listaRezervacija: list<Rezervacija>

listaArhiviranihDogadjaja: list<Dogadjaj>

listaArhiviranihKorisnika: list<Korisnik>

listaKomentara: list<Komentar>

listaTransakcija: list<Transakcija>

listaArhiviranihKomentara: list<Komentar>

Metode

Konstruktori/getteri/setteri

izlistajFizickaLica(): void

izlistajDogadjaje(): void
izlistajUstanove(): void
izlistajAdmine(): void
izlistajRezervacije(): void
izlistajArhiviraneDogadjaje(): void
izlistajArhiviraneKorisnike(): void
izlistajKomentare(): void
dodajNoviDogadjaj(dogadjaj:Dogadjaj): Obavijest
dodajNovogKorisnika(korisnik:Korisnik): Obavijest
filtrirajDogadjaje(mjesto:String): list<Dogadjaj>
filtirajUstanove(ustanova:Ustanova): list<Ustanova>
filtirajKomentare(korisnik:Korisnik): list<Komentar>
validirajPodatkeKorisnika(): Boolean
validirajPodatkeTransakcije(): Boolean
izbrisikorisnika(ID:Integer): void
izbrisidogadjaj(ID:Integer): void
odobriZahtjevAplikativno(dogadjaj:Dogadjaj): Boolean
odobriZahtjevAplikativno(rezervacija:Rezervacija): Boolean
odobriZahtjevAplikativno(korisnik:Korisnik): Boolean
obrisiKomentar(komentar:Komentar): void
arhivirajKorisnika(korisnik:Korisnik): void
arhivirajDogadjaj(dogadjaj:Dogadjaj): void
arhivirajKomentar(komentar:Komentar): void
zakljuciUplatu(transakcija:Transakcija): Boolean
dodajTransakcijuUListu(transakcija:Transakcija): void
printanjezvestaja(): void

Klasa BazaPodataka
Atributi
Ime:String

Driver:String

Metode

poveziSeNaBazu(): Boolean

citajIzBaze()

Enumi

TipFizickogLica

-obican

-student

-penzioner

TipKarte

-obicna

-studentska

-penzionerska

-vip

NacinPlacanja

-kartica

-online

-naLicuMjesta

Interfejsi

IIzracunClanarine-VIPKorisnik

PravljenjeIzvjestaja-ustanova

IzvjestajOVIP-VIPKorisnik

IzvjestajOTransakciji-Transakcija

IPlacanje-transakcija

IValidacija-EventManager

Izvjestaj-EventManager

IOdobravanjeEM-EventManager

IzvjestajORezervaciji-Rezervacija

IObavjestavanje-obavijest

I Odobravanje-admin

Veze između klasa

FizickoLice, Admin, Ustanova sa Korisnikom – generalizacija

FizickoLice I VIPKorisnik –generalizacija

Komentar I korisnik-kompozicija

Komentar I dogadjaj-kompozicija

EventManager I komentar-agregacija

Obavijest I korisnik-kompozicija

Ustanova I FizickoLice-kompozicija

dogadjaj I ustanova-kompozicija

Rezervacija I ustanova-kompozicija

Ustanova I BazaPodataka-asocijacija

Admin I BazaPodataka-asocijacija

Transakcija I FizickoLice-kompozicija

EventManager I BazaPodataka-kompozicija

EventManager I Korisnik/Komentar/Gost/Transakcija/Rezervacija/ dogadjaj - agregacija

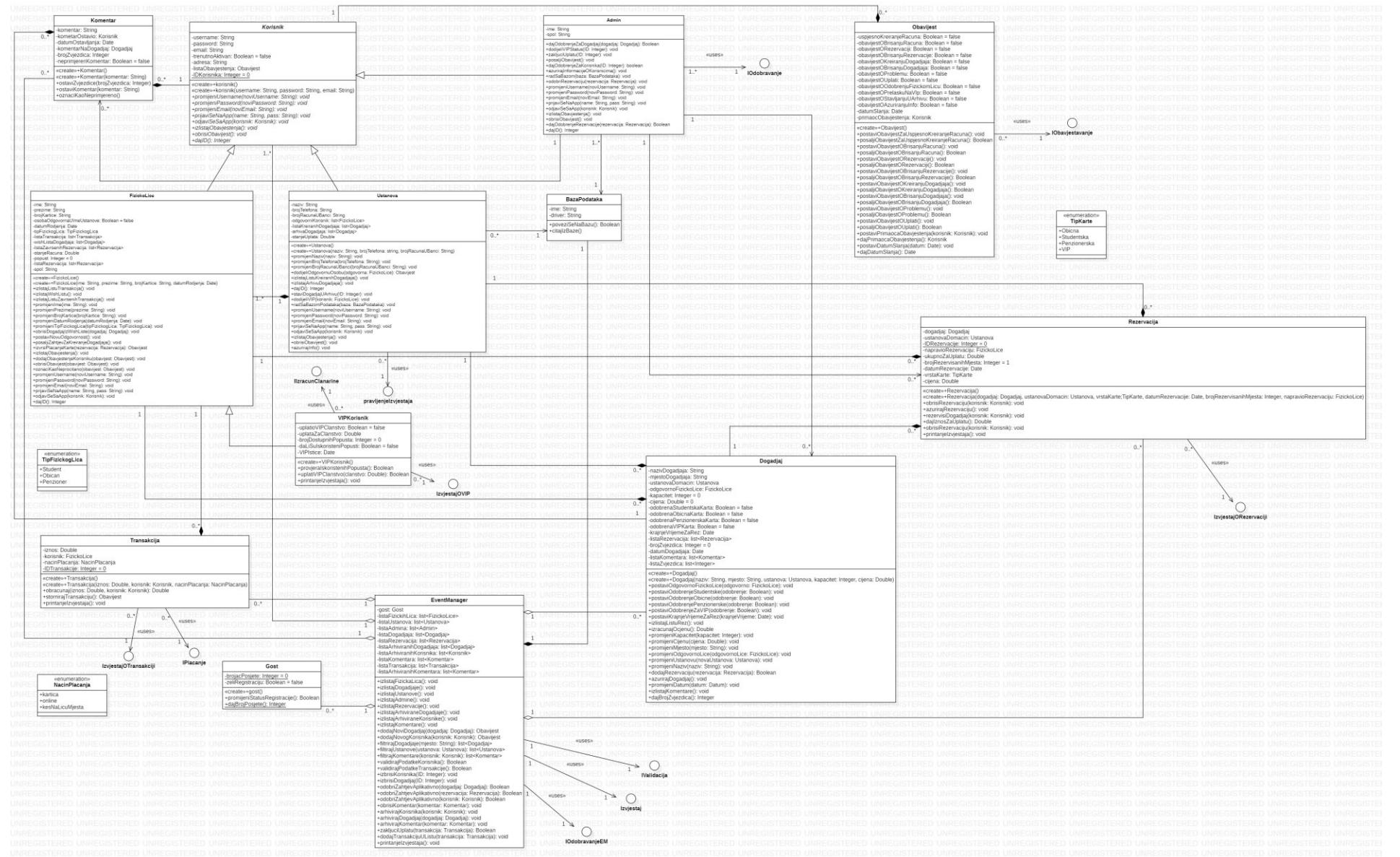
Admin I dogadjaj-asocijacija

Rezervacija I dogadjaj-kompozicija

Admin I Rezervacija-asocijacija

dogadjaj I FizickoLice-kompozicija

[Dijagram klasa](#)



SOLID principi

Analiza primijenjenih SOLID principa

Napomena: dijagram klase ne sadrži gettere/settere I sve konstruktore koji se mogu javiti čisto zbog preglednosti.

- Princip S: Svaka klasa treba imati samo jednu ulogu

U našem projektu, pokušali smo da jedna klasa ima samo jednu odgovornost: klasa Korisnik je klasa iz koje su izvedeni korisnici aplikacije,dakle korisnik "FizickoLice","Ustanova","Admin",VIPKorisnik",te ta klasa se brine o osnovnim podacima koji se tiču korisnika I njihovim izmjenama. Dakle,te sve klase se brinu o dodavanju I modifikaciji objekata od kojih se klase sastoje I isčitavanjem tih vrijednosti.Dalje,klasa koja bi potencijalno mogla da bude faktor za rušenjem ovog pricipa je klasa Ustanova,koja može da komunicira sa BazomPodataka,međutim,operacije koje služe ze pristup bazi I čitanje iz baze su izdvojene u posebnu klasu BazaPodataka,te smo na taj način pokušale riješiti taj problem. Dalje,klase koje imaju mogućnost da printaju odgovarajuće izvještaje,omogućeno je da te akcije budu izdvojene u zasebne interfejse. Klasa Događaj se brine o svojim atributima,njihovim modificiranjem I dodavanjem,baš kao I klasa Rezervacija.Klasa Obavijest je poseban vid klase koja nema mnogo atributa I samo služi da na jednom mjestu imamo templete obavještenja I da kroz svoje metode omogući da obavještenja se upisuju u liste obavještenja odgovarajućih korisnika.Klasa Obavijest služi za implementaciju metoda iz interfejsa za obavještavanje.Klasa Admin ima odgovornosti vezane za odobravanje događaja,zahtjeva,kreiranja računa,a za te akcije postoji ideja da se kroz ove metode utvrdi da li je događaj ili zahtjev validan da se pojavi u aplikaciji,te da u tim metodama admin pošalje obavještenje odgovarajućem korisniku. Uz admin,postoji I interfejs koji služi za odobravanje zahtijeva. Klasa Transakcija,nam služi da uplatimo rezervaciju I da se brine o osnovim podacima transakcije. Također,uz ovu klasu,imamo I interfejs za transakciju odnosno plaćanje. Najveća klasa u sistemu je klasa EventManager,koja komunicira najviše sa bazom podataka,daje odobrenja za zahtjeve,nudi mogućnost pravljenja izvještaja. Dakle,sve što je potencijalno van uloge jedne klase,izdvojeno je u odgovarajuće interfejse,da bi nam bilo lakše poslije kod nadogradnje programa.

- Princip O: Klasa treba biti otvorena za nadogradnje,ali zatvorena za modifikacije

Što se tiče ovog principa, klase koriste druge klase,ali u većini metoda koriste se postupci za ubacivanje objekata u liste,što je generička operacija. Također,za pojedine klase,postoje enumerativni tipovi koji omogućavaju da se metode ne moraju mijenjati,ukoliko se kreira novi tip,I da ukoliko se kreira neki novi tip,npr.novi tip karte,dovoljno je samo dodati novu jednu metodu u klasu Rezervacije,koja će da ubuduće dati mogućnosti odabira te vrste karte pri rezervaciji.Dakle,dešavat će se samo nadogradnja I ponuda nove mogućnosti.

- Princip L: Svaka osnovna klasa treba biti zamjenjiva svim svojim podtipovima bez da to utječe na ispravnost rada programa

Na našem primjeru, na mjestima gdje imamo pravljenje arhivskih listi ili nekih listi korisnika, korisnik koji je fizičko lice je također korisnik,baš kao I ustanova I admin, te sasvim je uredu da u jednoj metodi koja dodaje korisnike u listu napišemo da je to lista tipa Korisnik I to će imati smisla. Također,klasa koja se bavi obavještavanjem korisnika ima atribut primaocObavjestenja koji je tipa Korisnik. Na ovom mjestu,sasvim je svejedno da li obavještavamo fizičko lice,ustanovu ili admina,u pitanju je korisnik.

- Princip I: Bolje je imati više specifičnih interfejsa, nego jedan generalizovani

Da bismo,što je više moguće, ispoštovale princip S kod klasa,uvedeno je nekoliko interfejsa koji bi trebali da olakšaju posao,tako da postoje interfejsi koji omogućavaju kreiranje izvještaja u zavisnosti od potrebe klase I njenih osobina (neće isti izvještaj biti o uplati VIP članarine I izvještaj koji služi ustanovi da ima uvid u sve transakcije I rezervacije u jednom mjesecu),postoje izvještaji koji treba da pomognu klasama kod davanja odobrenja,izračuna transakcija,članarina,validacije podataka I obavještavanja.Dakle,ti interfejsi nam služe kao mini podprogrami koje će da koriste klase.

- Princip D: Sistem klasa I njegovo funkcionisanje treba ovisiti o apstrakcijama,a ne o konkretnim implementacijama

Ovaj princip zahtijeva da pri naslijedivanju od strane više klasa bazna klasa bude uvijek apstraktna,što je kod nas I slučaj-klasa korisnik je apstraktna klasa.

Paterni

*Nazivi paterna označenih **rozom** bojom,označavaju paterne implementirane na dijagramu klasa priloženom u folderu pod nazivom “Paterni”.Također,na dijagramu je pokušano da se implementira dosta paterna,što ne znači da će svi biti implementirani kada se zaista krene sa prebacivanjem ovih ideja u kod.Razlog je u tome što možda nisu potrebni svi paterni koji se sada nalaze na dijagramu i što bi možda implementacija svih navedenih dovela do rušenja kompletnosti i pojavljivanja krhkikh dijelova u programu.

STRUKTURALNI PATERNI

| NAZIV PATERNA | OPIS |
|-----------------|---|
| 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. Na taj način obezbeđuje se da će se objekti i dalje moći upotrebljavati na način kako su se dosad upotrebljavali, a u isto vrijeme će se omogućiti njihovo prilagođavanje novim uslovima. Ovaj patern bi se u našem slučaju mogao iskoristiti kod plaćanja, jer može se desiti da neko ko živi van države planira doći na neki događaj kod nas, a želi da mu se vrijednost karte iz KM preračuna u eure I da onda izvrši transakciju.Za potrebe ovog slučaja, moraćemo dodati jedan interfejs za obračun rezervacije u EUR,klasu Adapter koja će služiti da implementira metodu iz interfejsa I to će biti povezano sa klasom Transakcija,jer će ta implementirana metoda pozvati metodu za obračunavanje iz klase Transakcija,naravno nakon izvršene konverzije valuta.Isto tako,ovaj adapter nam nudi mogućnost nadogradnje koda,ukoliko se ubace |

| | |
|------------------|--|
| | još neke valute u sistem transakcija. |
| FACADE PATERN | <p>Fasadni patern služi kako bi se klijentima pojednostavilo korištenje kompleksnih sistema. Klijenti vide samo fasadu, odnosno krajnji izgled objekta, dok je njegova unutrašnja struktura skrivena. Na ovaj način smanjuje se mogućnost pojavljivanja grešaka jer klijenti ne moraju dobro poznavati sistem kako bi ga mogli koristiti.</p> <p>Ovaj patern se u našem dijagramu može povezati sa klasom EventManager, koja omogućava izlistavanje podataka o korisnicima, a korisnika u sistemu ima više vrsta, te sa tim u vezi ima više relevantnih atributa koji se moraju prikazati za svakog korisnika, a onaj ko želi da vidi podatke, ne treba da se zamara sa atributima npr. klase FizickoLice i klase Ustanova.</p> <p>Ovaj patern se u našem dijagramu može povezati sa klasom Korisnik, jer ona objedinjuje više vrsta korisnika, koji je svaki specifičan na svoj način, a svi posjeduju nekoliko zajedničkih atributa.</p> <p>U našem dijagramu, primjenjujući ovog paterna vidimo kod klase za obavještavanje korisnika, jer u njoj prosti postoje metode za slanje obavijesti za generički tip zvani Korisnik I korisnik aplikacije ne treba da unosi sve podatke za korisnika kojem želi poslati obavijest već je dovoljan samo ID ili ime. Uz to, ukoliko se iskoristi decorator patern, kada želimo da se pošalje obavijest korisniku, korisnik ne treba da se zamara sa tim kakve vrste obavijesti prima korisnik kojem šalje obavijest - da li je to putem emaila, Faceboka i sl.</p> |
| DECORATOR PATERN | <p>Decorator patern služi za omogućavanja različitih nadogradnji objektima koji svi u osnovi predstavljaju jednu vrstu objekta (odnosno, koji imaju istu osnovu). Umjesto da se definije veliki broj izvedenih klasa, dovoljno je omogućiti različito dekoriranje objekata (tj. dodavanje različitih detalja), te se na taj način pojednostavljuje i rukovanje objektima klijentima, i samo implementiranje modela objekata.</p> <p>U našem dijagramu bi se ovaj patern mogao iskoristiti u slučaju klase Obavijest na sljedeći način: Ukoliko želimo da korisnika obavještavamo putem</p> |

| | |
|------------------|--|
| | <p>emaila,SMS,Facebooka,moguće je ukoliko imamo interfejs IObavjestavanje koji će sadržati definiciju metode za obavještavanje posaljiObavijest I biće dodane još četiri klase EmailObavijest,SMSObavijest, FacebookObavijest i klasa EMOlavestavanje,koja je vezana za obavjestavanje u našoj aplikaciji.</p> |
| BRIDGE PATERN | <p>Bridge patern služi kako bi se apstrakcija nekog objekta odvojila od njegove implementacije. Ovaj patern veoma je važan jer omogućava ispunjavanje Open-Closed SOLID principa, odnosno uz poštivanje ovog paterna omogućava se nadogradnja modela klasa u budućnosti te osigurava da se neće morati vršiti određene promjene u postojećim klasama. U našem dijagramu vidimo potencijalno mjesto za iskorištavanje ovog paterna-kod transakcije,pojavljuje se interfejs za plaćanje.Pošto nije isti obračun za fizičko lice sa običnim profilom I sa VIP profilom,zato će trebati izmijeniti stanje u ovom dijelu dijagrama na način da se doda interfejs za obračun,koji će sadržavati definiciju metode za obračun rezervacije.Dodati novu klasu Bridge I njoj će jedino moći pristupiti klasa Transakcija.</p> |
| COMPOSITE PATERN | <p>Composite patern služi za kreiranje hijerarhije objekata. Koristi se kada svi objekti imaju različite implementacije nekih metoda, no potrebno im je svima pristupati na isti način, te se na taj način pojednostavljuje njihova implementacija.U našem dijagramu se može pronaći potencijalno mjesto za korištenje ovog paterna-kod klase EventManager,ova klasa ima mogućnost da pravi izvještaje.Isto tako,mogućnost pravljenja izvještaja imaju klase Ustanova,Rezervacija,Transakcija,VIPKorisnik.Pošto bi to trebali biti generički izvještaji za svaku ovu klasu,tj.izvještaj o instanci te klase za njen život unutar aplikacije i najosnovnije podatke,moguće je sve ove interfejs spojiti u jedan interfejs Izvjestaj,te povezati ga sa svim ovim klasama.Tako ako bi neko htio da vidi sve izvještaje o radu aktera u aplikaciji,to bi bilo moguće nakon realizacije ovog paterna.</p> |

| | |
|------------------|--|
| PROXY PATERN | <p>Proxy patern služi za dodatno osiguravanje objekata od pogrešne ili zlonamjerne upotrebe. Primjenom ovog paterna omogućava se kontrola pristupa objektima, te se onemogućava manipulacija objektima ukoliko neki uslov nije ispunjen, odnosno ukoliko korisnik nema prava pristupa traženom objektu. Ovaj patern će biti iskorišten kod ispitivanja prava pristupa, konkretno, admin može pristupiti svim userima i svim njihovim podacima koji su unešeni u bazu podataka, dok korisnik ima pravo da pregleda sve svoje informacije, a samo osnovne podatke od drugih korisnika. Zato će nam trebati klasa Proxy, koja će vršiti autentifikaciju putem metode pristup() i ova klasa će naslijediti metode iz interfejsa IProxy. Isto tako, ovaj patern se može iskoristiti kod komentara-gost može da čita komentare na događaje, dok logovani korisnik, bilo FizickoLice ili Ustanova ili Admin, mogu i da čitaju i da pišu komentare. Tako da će biti potreban interfejs IProxyCom, a njegove metode će da implementira klasa ProxyCom, koja će vršiti autentifikaciju putem metode pristup() za pristupu komentarima.</p> |
| FLYWEIGHT PATERN | <p>Flyweight patern koristi se kako bi se onemogućilo bespotrebno stvaranje velikog broja instanci objekata koji svi u suštini predstavljaju jedan objekat. Samo ukoliko postoji potreba za kreiranjem specifičnog objekta sa jedinstvenim karakteristikama (tzv. specifično stanje), vrši se njegova instantacija, dok se u svim ostalim slučajevima koristi postojeća opća instanca objekta (tzv. bezlično stanje). Korištenje ovog paterna veoma je korisno u slučajevima kada je potrebno vršiti uštedu memorije.</p> <p>U našem primjeru kada bi se enumeracije za TipKarte, TipFizickogLica, VrstePlaćanja pretvorile u klase, njihovi elementi bi bile klase sa istim osobinama. Tako bi nam ovdje trebao interfejs, recimo za vrste karata, IKarte i koja će imati definiciju metode za razlikovanje vrste karata, a u klasi Rezervacija samo dodati metodu koja će se pozvati kod biranja vrste karte. U našem dijagramu, ovaj patern ima potencijalno</p> |

| | |
|--|---|
| | <p>mjesto za svoju implementaciju-kod klase Admin,postoji interfejs za odobravanje zahtjeva koje dobije admin.Pošto admin može davati više vrsta odobrenja-odobrenje za kreiranje događaja,računa,odobrenje za izvršenje uplate,transakcije,rezervacije,...,ovaj interfejs se može povezati sa još par klasa koje bi se trebale implementirati,npr.klase DogadjajOdobrenje,RezervacijaOdobrenje,Racun Odobrenje,TransakcijaOdobrenje,a klasa Admin bi se trebala povezati sa klasom Odobrenja(ovaj slučaj nije implementiran).Takvo slično potencijalno mjesto je kod klase EventManager,jer je ona povezana sa interfejsom za davanje aplikativnih odobrenja.Tako bi se mogla klasa EventManager povezati sa klasom Odobrenje,ta klasa sa ovim interfejsom koji već postoji,te kreirati klase OdobriZahtjevAplikativnoDogadjaj, OdobriZahtjevAplikativnoRezervacija, OdobriZahtjevAplikativnoKorisnik.</p> |
|--|---|

KREACIJSKI PATERNI

| NAZIV PATERNA | OPIS |
|------------------|--|
| SINGLETON PATERN | <p>Singleton patern služi kako bi se neka klasa mogla instancirati samo jednom. Na ovaj način može se omogućiti i tzv. lazy initialization, odnosno instantacija klase tek onda kada se to prvi put traži. Osim toga, osigurava se i globalni pristup jedinstvenoj instanci - svaki put kada joj se pokuša pristupiti, dobiti će se ista instance klase. Ovo olakšava i kontrolu pristupa u slučaju kada je neophodno da postoji samo jedan objekat određenog tipa. U našoj aplikaciji bi se ovaj patern mogao iskoristiti kod klase Admin,ali kod nas je moguće da postoji više admina,tako da se neće dobiti jedinstvena instance. Mjesto koje bi moglo da ispunji ovaj patern je klasa BazaPodataka,jer je to jedinstvena baza za cijelu aplikaciju i ova klasa bi se instancirala samo</p> |

| | |
|-----------------------|--|
| | jednom.Zbog toga,u ovu klasu će se dodati statički atribut BazaSingleton,statičku metodu koja će vršiti vraćanje statičkog atributa,te u ovoj klasi će postojati metode za pristup bazi. |
| PROTOTYPE PATERN | Prototype patern omogućava smanjenje kompleksnosti kreiranja novog objekta tako što se uvodi operacija kloniranja. Na taj način prave se prototipi objekata koje je moguće replicirati više puta a zatim naknadno promijeniti jednu ili više karakteristika, bez potrebe za kreiranjem novog objekta nanovo od početka. Ovime se osigurava pojednostavljenje procesa kreiranja novih instanci, posebno kada objekti sadrže veliki broj atributa koji su za većinu instanci isti. U našem slučaju moguće je kreirati isti događaj, u isto vrijeme, ali da imaju drugačiju publiku:npr.kino kreira događaj za premijeru filma.Premijera se dešava na datum xx.yy.zzzz,a kino kreira dva takva događaja,za isti film i za isti datum.Jedina razlika između ta dva događaja jeste ta što je jedan događaj vezan za salu A,gdje je premijera za posebne zvanice i drugi događaj je vezan za salu B,gdje je premijera za sve ljudе koji žele doći na premijeru.Nakon što se kreiraju ta dva događaja,moguće je za premijeru koja je u sali B,povećati ili smanjiti kapacitet karata za prodaju ili vrijeme događaja (tipa prologirati za pola sata i sl.),bez da to utiče na događaj za ljudе koji idu u salu A. |
| FACTORY METHOD PATERN | Factory method patern služi za omogućavanje instanciranje različitih vrsta podklasa koristeći factory metodu koja odlučuje koja će se podklasa instancirati i koja programska logika izvršiti. Na ovaj način osigurava se ispunjavanje O SOLID principa, jer se kod za kreiranje objekata različitih naslijeđenih klasa ne smješta samo u jednu zajedničku metodu, već svaka podklasa ima svoju logiku za instanciranje željenih klasa, a samo instanciranje kontroliše factory metoda koju različite klase implementiraju na različit način. Ovaj patern je u našem dijagramu već potencijalno primijenjen kod klase Korisnik,ona sadrži metode koje |

| | |
|--------------------------------|---|
| | naslijeduje svaka naslijeđena klasa,samo što bi trebalo ku klasu Korisnik dodati factory metodu,kreirati interfejs IKorisnikFactory,koji će zamijeniti metode za kreiranje računa,a taj interfejs naslijediti klase FizickoLice,Ustanova,Admin. |
| ABSTRACT FACTORY METHOD PATERN | <p>Abstract factory patern služi kako bi se izbjeglo korištenje velikog broja if-else uslova pri kreiranju različitih hijerarhija objekata. Ukoliko postoji više tipova istih objekata te različite klase koriste različite podtipove, te klase postaju fabrike za kreiranje objekata zadanog podtipa bez potrebe za specificiranjem pojedinačnih objekata. Na ovaj način se, korištenjem nasljeđivanja, ukida potreba za postojanjem if-else uslova jer određeni tip fabrike sadrži određene tipove objekata i zna se tačno koju podklasu će instancirati. Ovaj patern bi se mogao u našem slučaju iskoristiti kod rezervacija,jer postoji više vrsta karata u zavisnosti od vrste fizičkog lica-obična,studentska,VIP,penzionerska.Stoga,ne bi bilo loše posjedovati interfejs IFactoryVrste,koji će omogućiti implementiranje metoda koje su zajedničke za sve fabrika-klase.Zajedničke metode di bile dajPopust(),dajOznaku().Isto tako,trebale bi se kreirati klase npr.StudentFactory,koja bi odjedinjavala atributе i metode vezane za studenta,a to su studentska karta i student kao tip fizičkog lica.Pošto se u već nekom od prijašnjih paterna iskoristila slična osobina,ovaj patern se neće implementirati na dijagramu.Sa druge strane,uvodenje još klasa bi možda još više zakomplikovalo implementaciju i možda narušilo kompletnost sistema.</p> |
| BUILDER PATERN | Builder patern služi za apstrakciju procesa konstrukcije objekta, kako bi se kao rezultat mogle dobiti različite specifikacije objekta koristeći isti proces konstrukcije. Ovaj patern koristi se kako bi se izbjeglo kreiranje kompleksne hijerarhije klasa te kako bi se izbjegao kompleksni programski kod konstruktora jedne klase koja može imati različite konfiguracije atributa. |

Različiti dijelovi konstrukcije objekta izdvajaju se u posebne metode koje se zatim pozivaju različitim redoslijedom ili se poziv nekih dijelova izostavlja, kako bi se dobili željeni različiti podtipovi objekta bez potrebe za kreiranjem velikog broja podklasa. Ovaj patern bi se u našem slučaju mogao iskorititi kod filtriranja događaja u novostima korisnika-korisnik može da postavi zahtjeve kako želi da mu se izlistavaju aktuelni događaji,dok može da postavi da sistem automatski izbacuje događaje na dashboard.Za ovaj slučaj,biće potrebno kreirati interfejs IBuilderFilter,te kreirati dvije nove klase ManuelniFilter i AutomatskiFilter,te ih povezati sa interfejsom,a interfejs sa klasom EventManager,zbog atributa lista dogadjaja.

dijagram klase sa primijenjenim ovim paternima

PATERNI PONAŠANJA

Dijagram klasa sa implementiranim paternima ponašanja se nalazi u folderu "Paterni" i u folderu sa dijagramima klasa pod nazivom "DijagramKlasaSaSvimPaternima". Napominjem da su neki paterni stavljeni na dijagram zbog pokazivanja kako izgledaju na dijagramu, ali ne znači da će svi biti implementirani (konkretno, postoje preklapanja paterna ponašanja i kreacijskih paterna na nekim mjestima, što znači da će se za implementaciju razmotriti najbolje rješenje).

Elementi paterna ponašanja su na dijagramu obojeni **plavom** bojom, kao i u tabeli.

| NAZIV PATERNA | OPIS |
|------------------------|---|
| STRATEGY PATERN | Strategy patern 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. S obzirom da ovaj patern rješava problem razbacanih if-else struktura u programu, u našem slučaju bi se ovaj patern mogao iskoristiti kod odabira načina plaćanja, odabira vrste fizičkog lica i odabira vrste karte. |
| STATE PATERN | State patern 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. U našem slučaju, klasa obavijesti se sastoji od puno atributa tipa bool i puno metoda koje ovise o ovim atributima. Ovakvo stanje klase bi u implementaciji izazvalo puno if-else struktura. Zbog toga, bilo bi praktičnije dodati interfejs IObavijestMode koji će sadržavati metode čija je implementacija različita za različite vrste obavijesti koje se šalju korisnicima. Dodati ovaj interfejs kao atribut klase Obavijest i sa njim ćemo zamijeniti sve ove bool attribute. Dalje, treba povezati ovaj interfejs sa postojećim klasama za vrste obaveštenja koja postoje u aplikaciji. |
| TEMPLATE METHOD PATERN | Template method patern služi za omogućavanje izmjene ponašanja u jednom ili više dijelova. Najčešće se primjenjuje kada se za neki |

| | |
|-----------------|--|
| | <p>kompleksni algoritam uvijek trebaju izvršiti isti koraci, no pojedinačne korake moguće je izvršiti na različite načine. U biti, ovaj patern je već primjenjen kod nas, jer kao "template" za glave aktere u sistemu je klasa Korisnik i svi korisnici koji postoje i koji se izvode iz ove klase, imaju zajedničke osobine, ali i dosta dijelova u kojima je ponašanje izmijenjeno i koje se izvršava na različite načine.</p> |
| OBSERVER PATERN | <p>Observer patern služi kako bi se na jednostavan način kreirao mehanizam pretplaćivanja. Pretplatnici dobivaju obavještenja o sadržajima na koje su pretplaćeni, a za slanje obavještenja zadužena je nadležna klasa. Na ovaj način uspostavlja se relacija između klasa kako bi se mogle prilagoditi međusobnim promjenama. Ovaj patern se već nalazi u našem sistemu kod klase Obavijest, sa tim da ako se pogleda dijagram klasa, ova klasa je zadužena da se svakako šalju obavijesti, ali bismo mogli ubaciti mogućnost da se korisnik pretplati na obavještenja iz neke ustanove. Dodat ćemo interfejs IPreplatnik koji će sadržavati metodu za pregled kompletног događaja u aplikaciji i metodu koja će omogućavati rezervaciju nakon primljene obavijesti. Trebamo u klasu Ustanova dodati atribut lista pretplatnika i pretplatnici će naslijedivati interfejs IPreplatnik. Interfejs će se trebati povezati sa klasama za rezervaciju i za događaj. U biti, ovaj patern je kod klase Obavijest već na neki način obrađen.</p> |
| ITERATOR PATERN | <p>Iterator patern 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 drveta, jednostrukih 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 možemo primijeniti kod situacije da</p> |

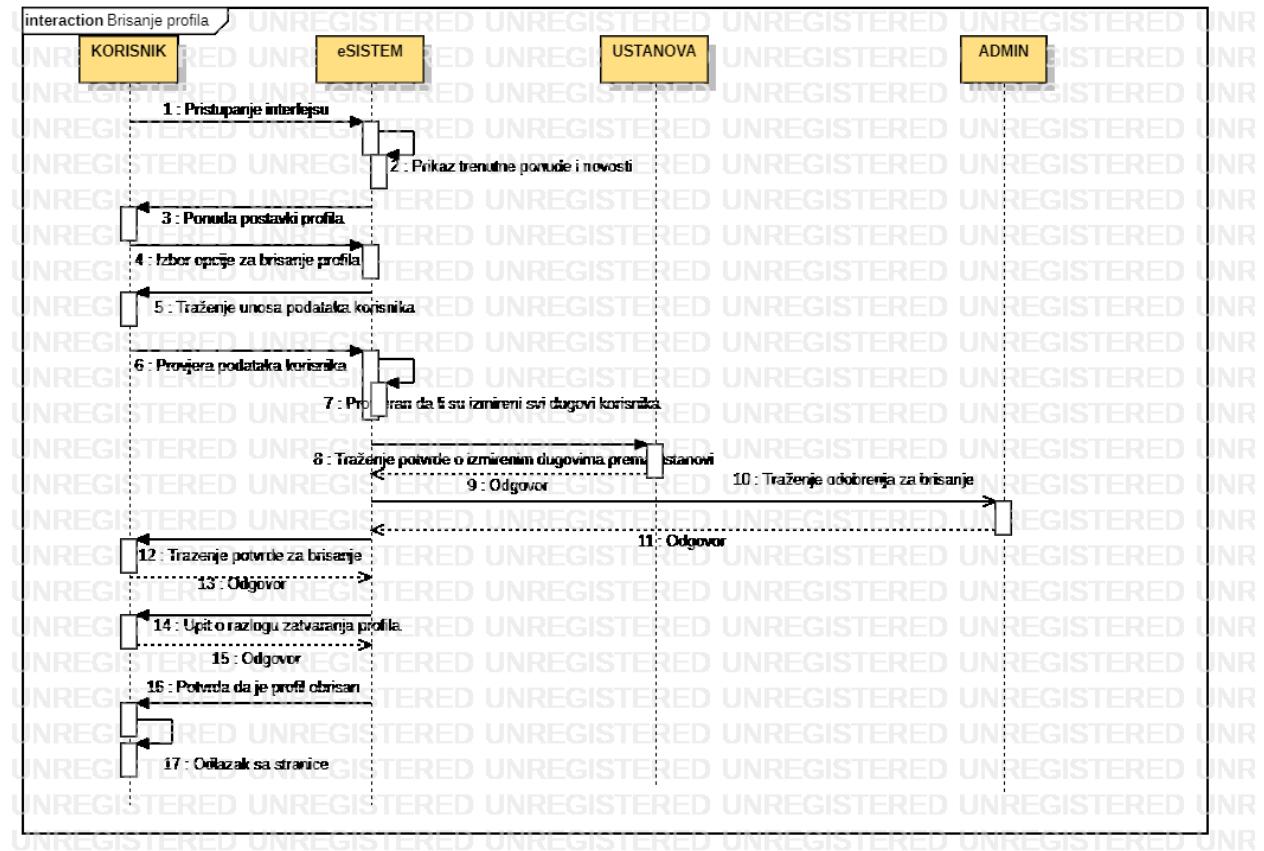
| | |
|---------------------------------|---|
| | <p>korisnik želi da filtrira i pretražuje događaje po određenom kriterijumu. Možemo dodati interfejs ISljedeciDogadjaj, a ovaj interfejs će da naslijedi iteratore: nasumičnilterator, abecednilterator, ustanovalterator. Dalje, dodaćemo interfejs IKreatorIteratora, koji će biti povezan sa klasom EventManager i omogućitiće kretanje kroz listu događaja koji se već nalaze u bazi podataka aplikacije.</p> |
| CHAIN OF RESPONSIBILITY PATTERN | <p>Chain of responsibility patern namijenjen je kako bi se jedan kompleksni proces obrade razdvojio na način da više objekata na različite načine procesiraju primljene podatke. Kreiranjem lanca obrade, pri čemu nakon vršenja parcijalne obrade svaki objekat prosljeđuje podatke narednom objektu u nizu, postiže se smanjenje kompleksnosti pojedinačnih procesa obrade i jednostavnije razumijevanje cijelog procesa. Osim toga, svakom objektu dolaze samo oni podaci koje trebaju obraditi te se na taj način izbjegava duplicitanje sadržaj. U našem slučaju postoji mogućnost da fizičko lice podnese zahtjev da objavi događaj u ime neke ustanove. Da bi se taj događaj na kraju našao u bazi podataka i u aplikaciji, taj događaj moraju da potvrde i klasa EventManager, tj. da on ispunjava zahtjeve aplikacije šta jedan zahtjev i događaj treba da ima od ključnih stvari, klasa Ustanova za koju se objavljuje događaj i klasa Admin. Zato nam trebaju klasa Zahtjev sa osnovnim atributima koje jedan zahtjev treba da ima, klasa ZahtjevPotvrda, interfejs IHandler, te klase za obradu koje će naslijediti klasu ZahtjevPotvrda, one će obrađivati zahtjev na osnovu datih informacija i one će implementirati vrituelnu metodu obradi na način da vrše pozivanje onih dijelova za koje su zaduženi.</p> |
| MEDIATOR PATTERN | <p>Mediator patern namijenjen je za smanjenje broja veza između objekata. Umjesto direktnog međusobnog povezivanja velikog broja objekata, objekti se povezuju sa međuobjektom medijatorom, koji je zadužen za njihovu komunikaciju. Kada neki objekt želi poslati poruku drugom objektu, on šalje poruku</p> |

mediјatoru, a mediјator proslijeđuje tu poruku namijenjenom objektu ukoliko je isto dozvoljeno. Na ovaj način stvara se centralizovano mjesto kontrole objekata te onemogućavanje komunikacije između objekata u slučajevima kada je to potrebno. Ovaj patern bismo iskoristili ukoliko bi dodali mogućnost stvaranja chara za korisnike aplikacije, i da onda razgraničimo poruke za ustanovu i za fizičko lice, recimo za ustanovu bi se vršila određena filtracija, kao i za maloljetne korisnike sistema.

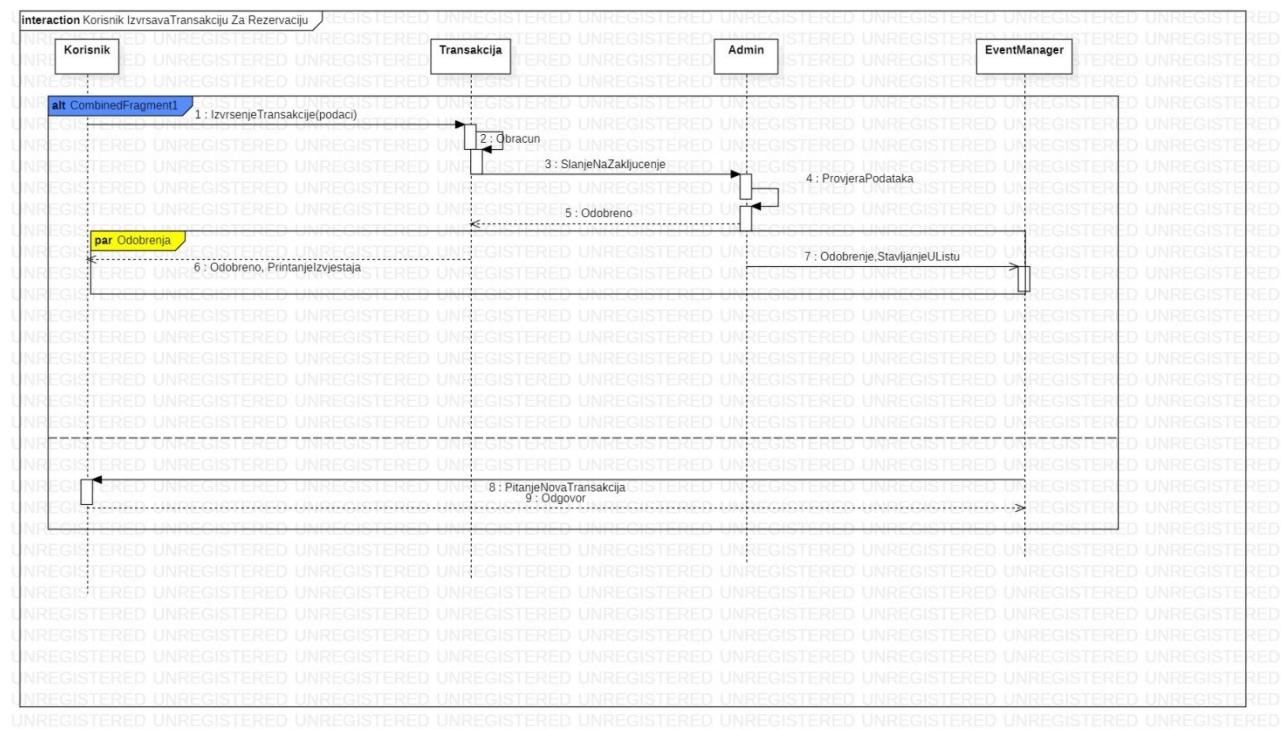
Dijagram klase sa primijenjenim svim paternima

Dijagrami sekvence

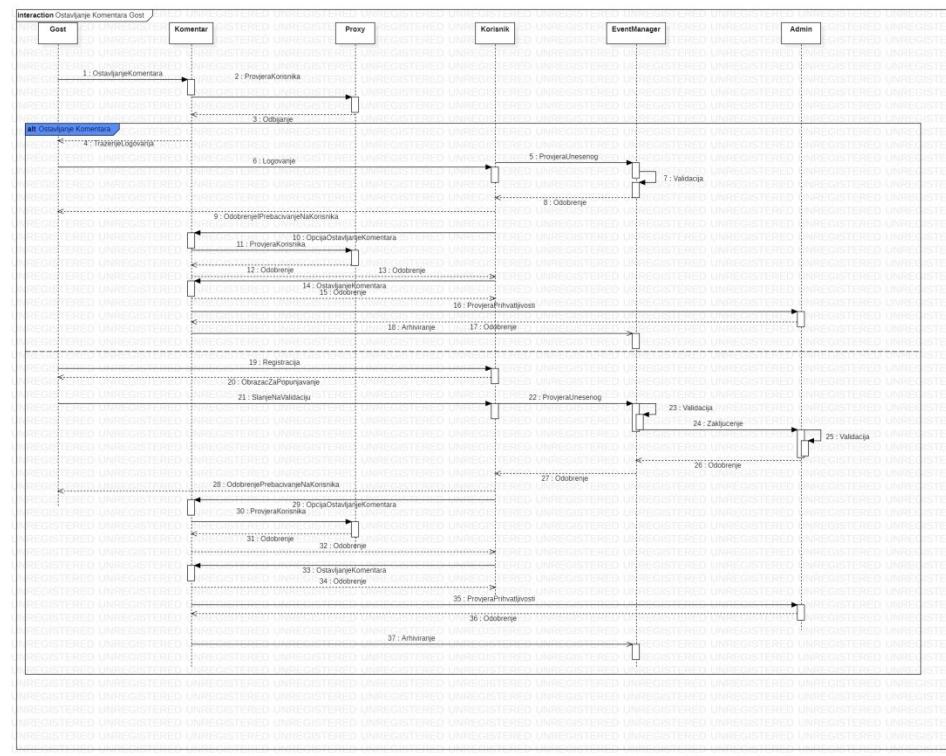
Brisanje profila



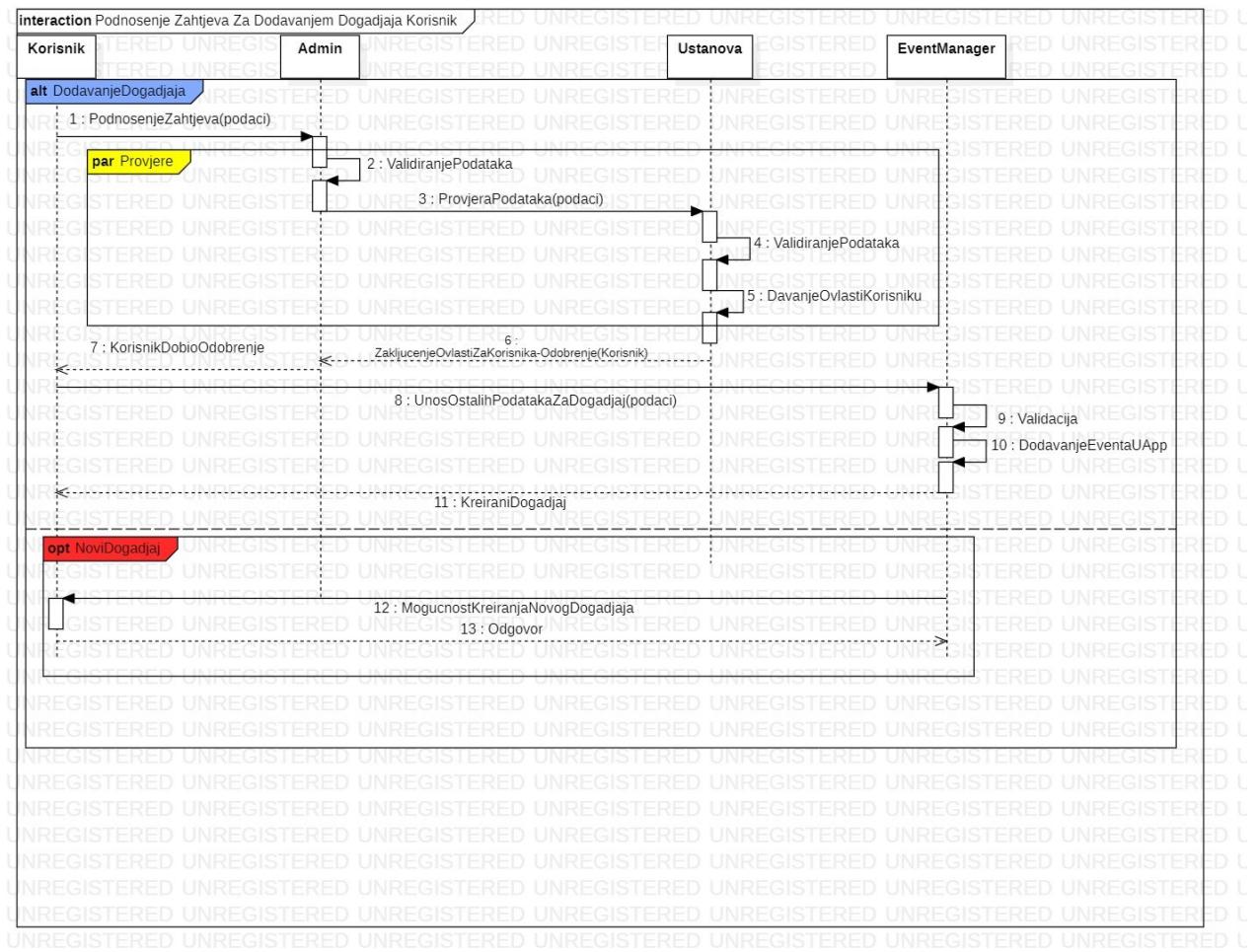
Korisnik izvršava transakciju



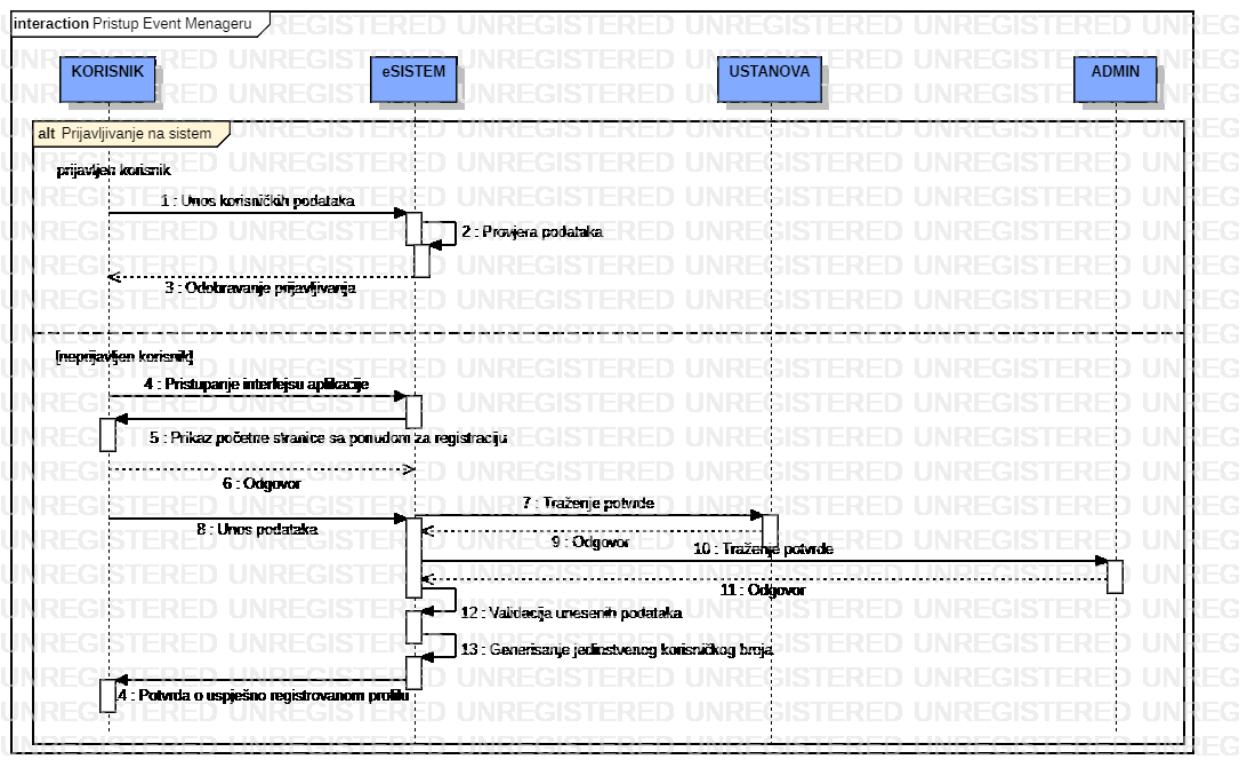
Ostavljanje komentara - gost



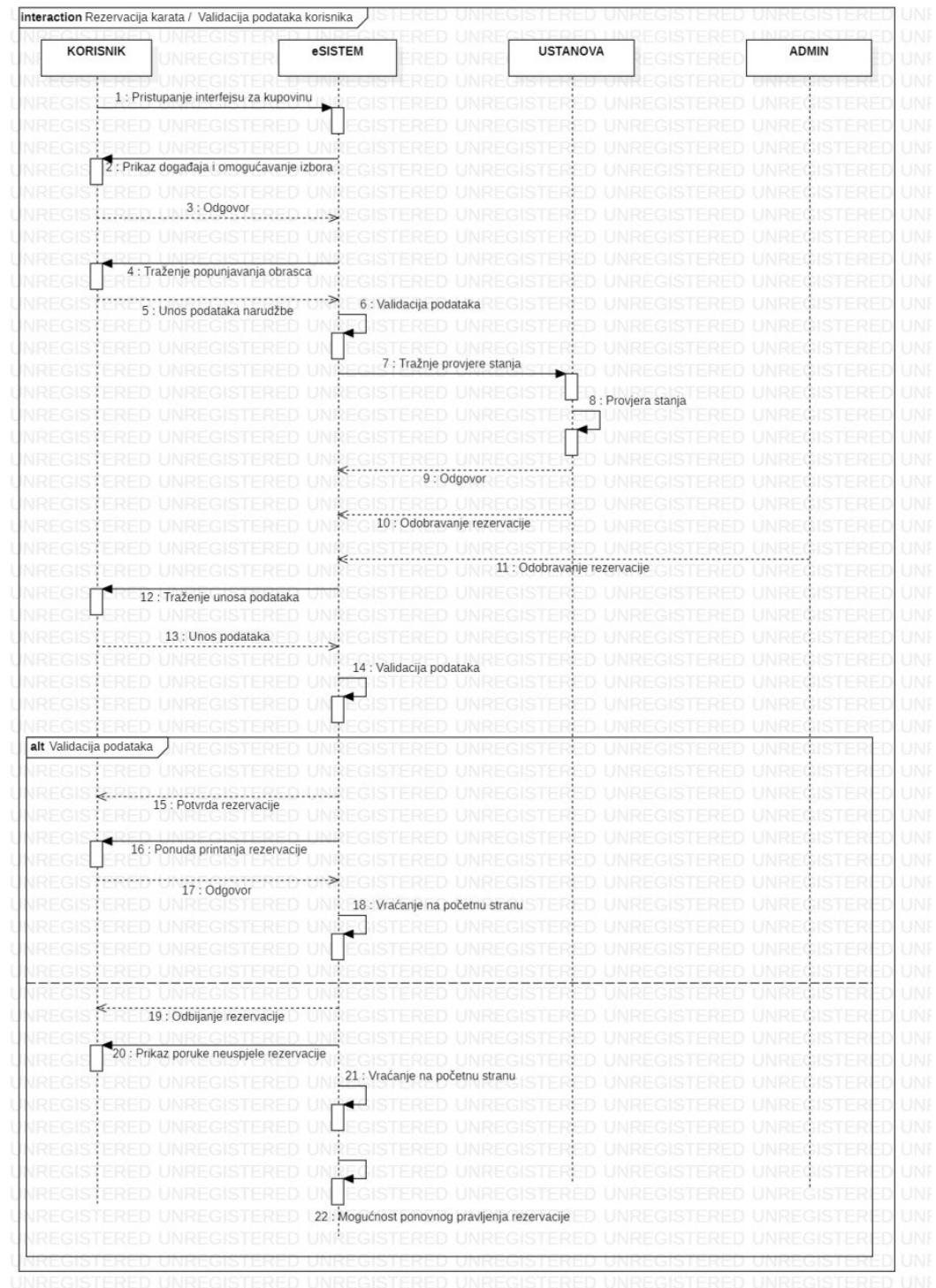
Podnošenje zahtjeva za dodavanjem događaja-korisnik



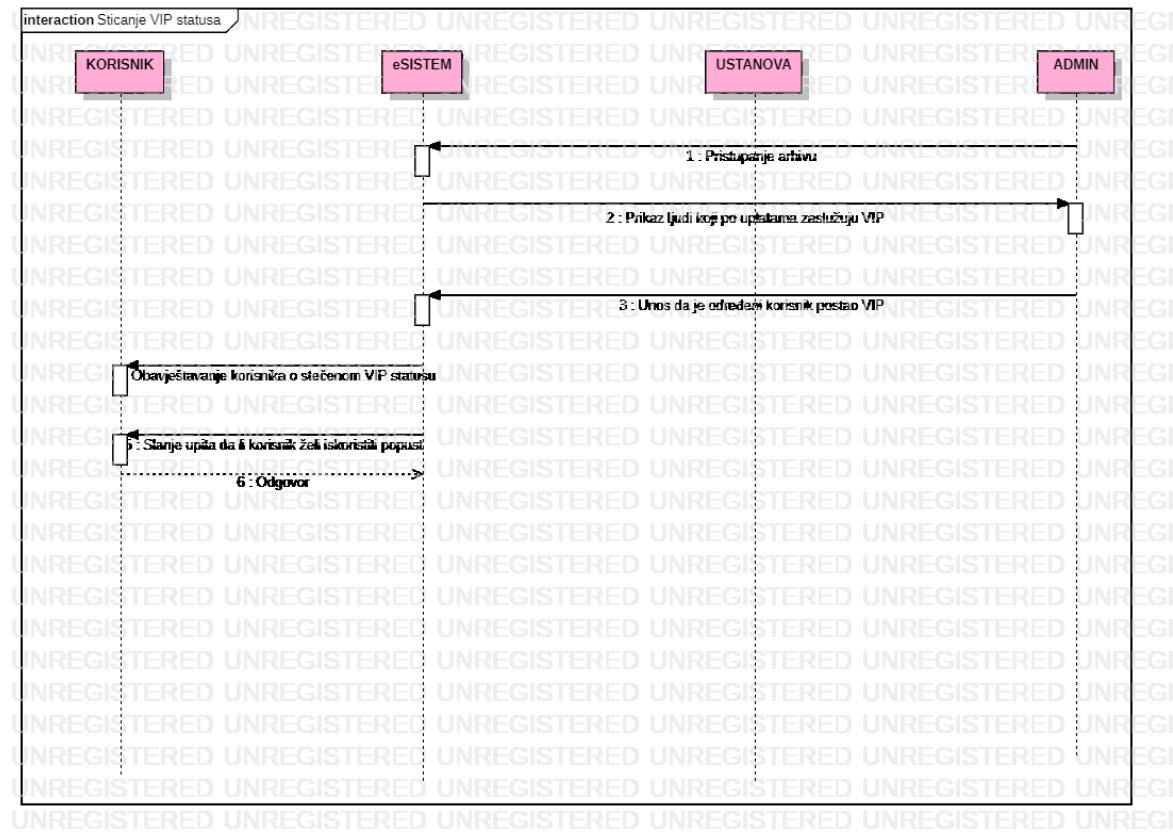
Pristup sistemu



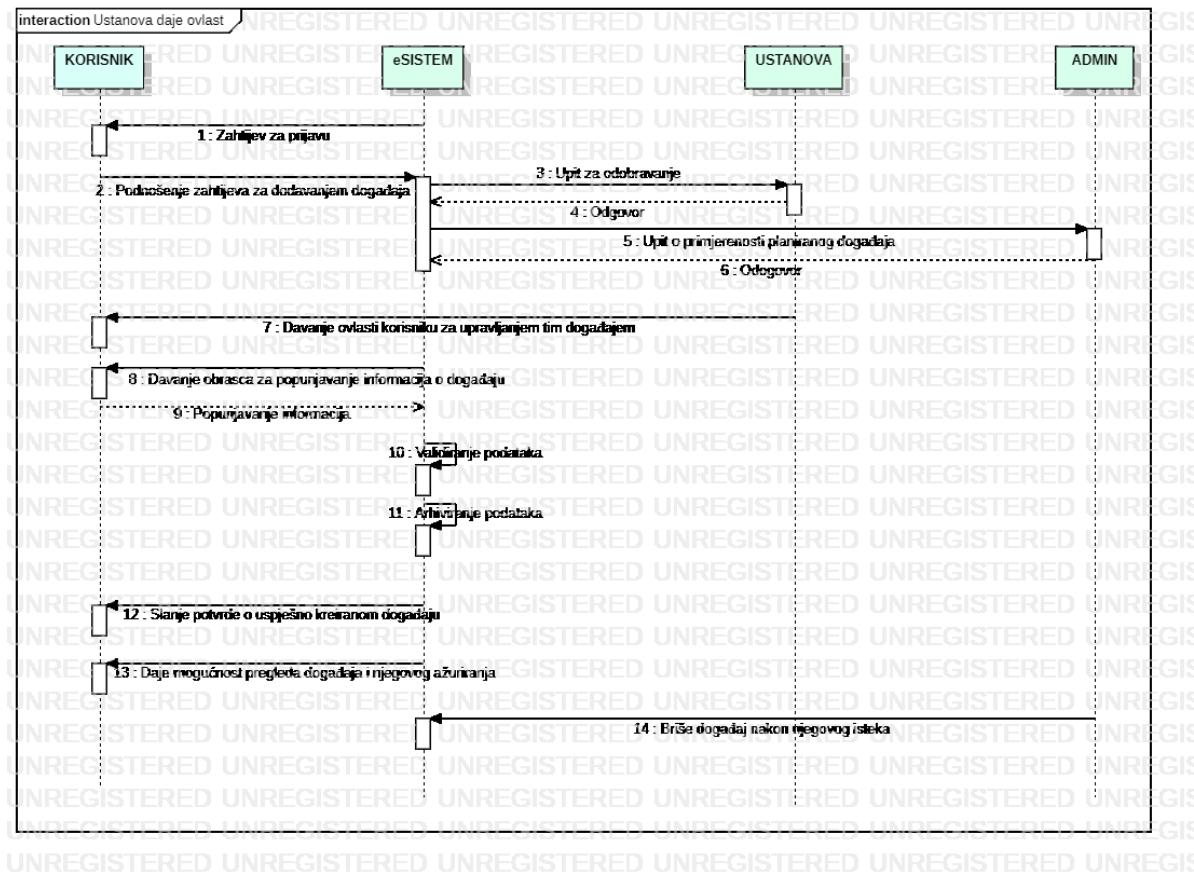
Rezervacija karata-validacija podataka



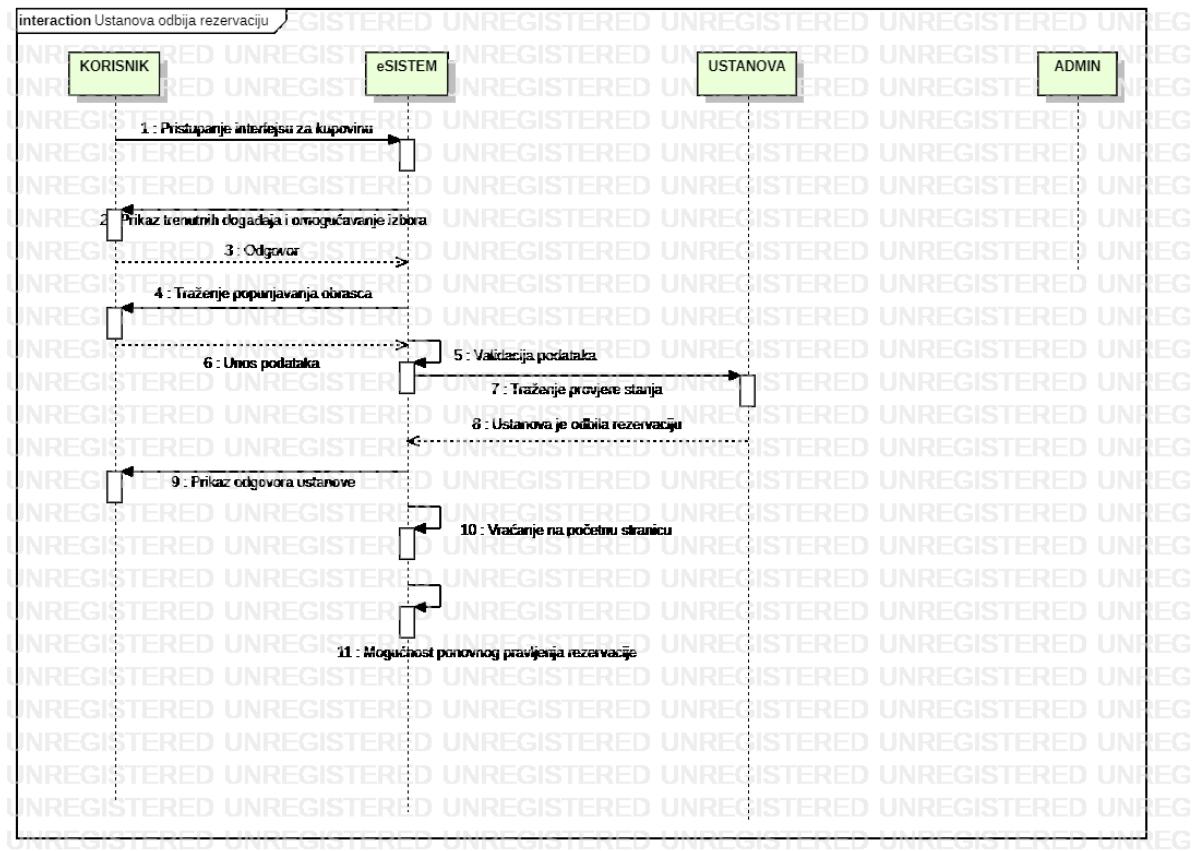
Sticanje VIP statusa



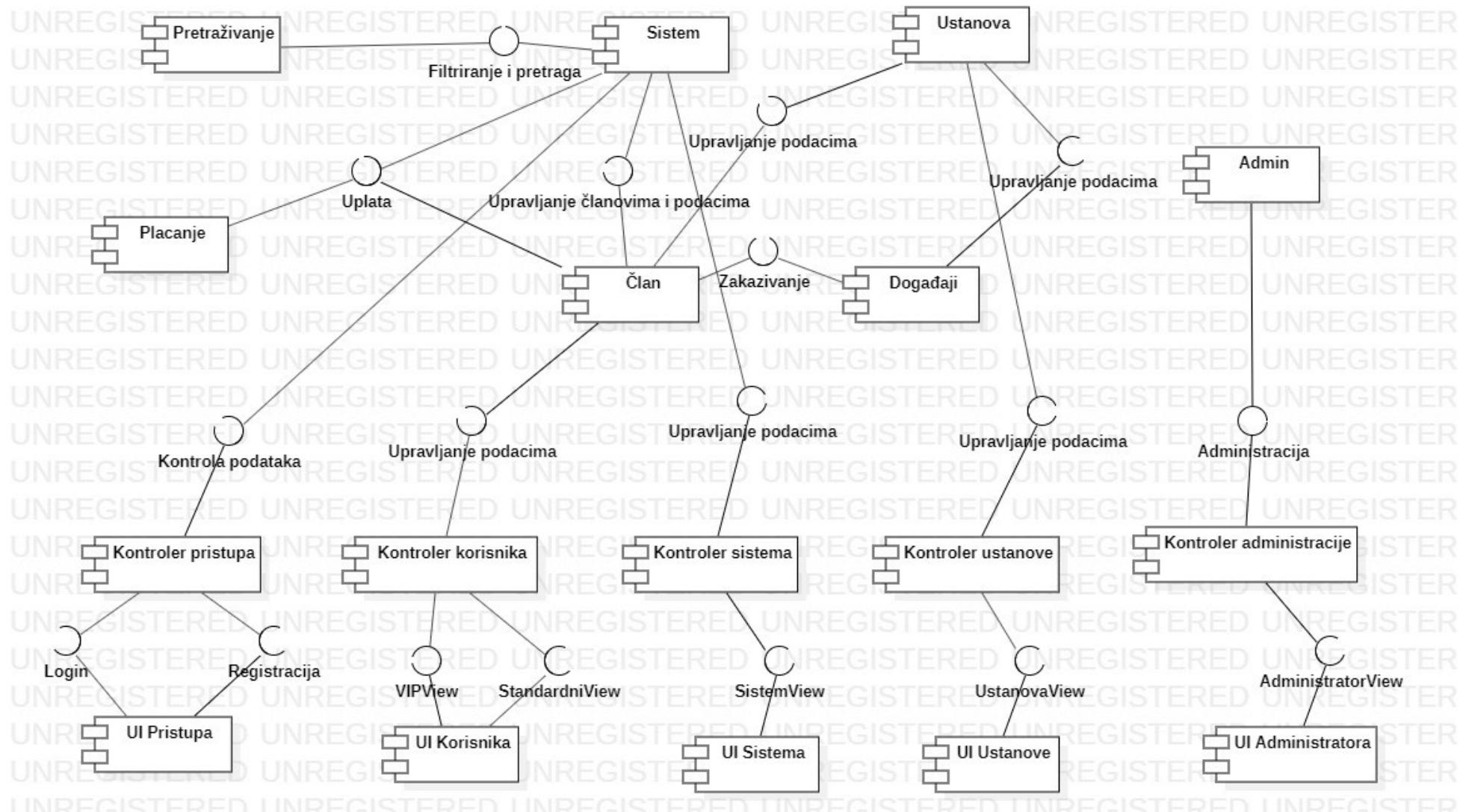
Ustanova daje ovlast



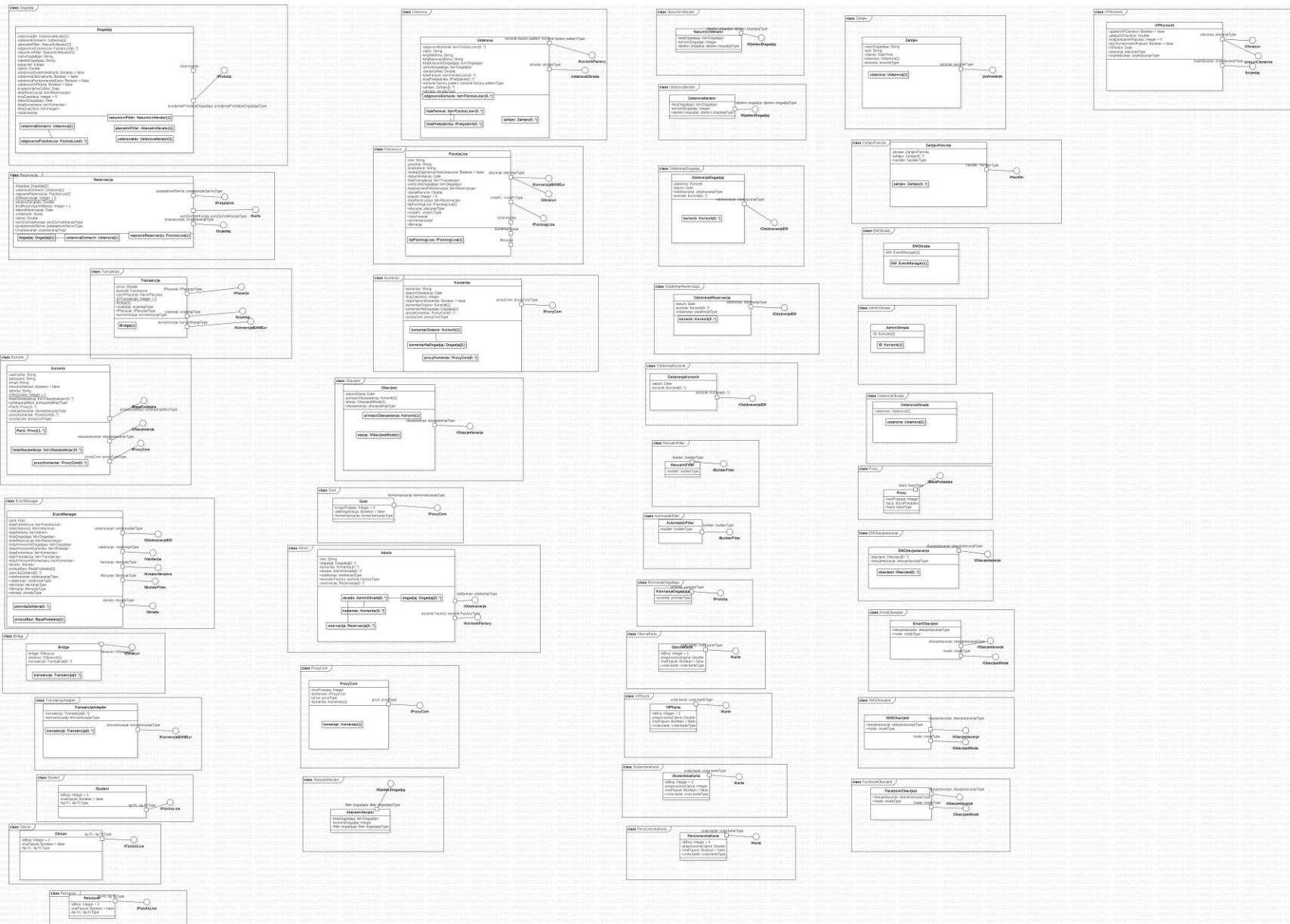
Ustanova odbija rezervaciju



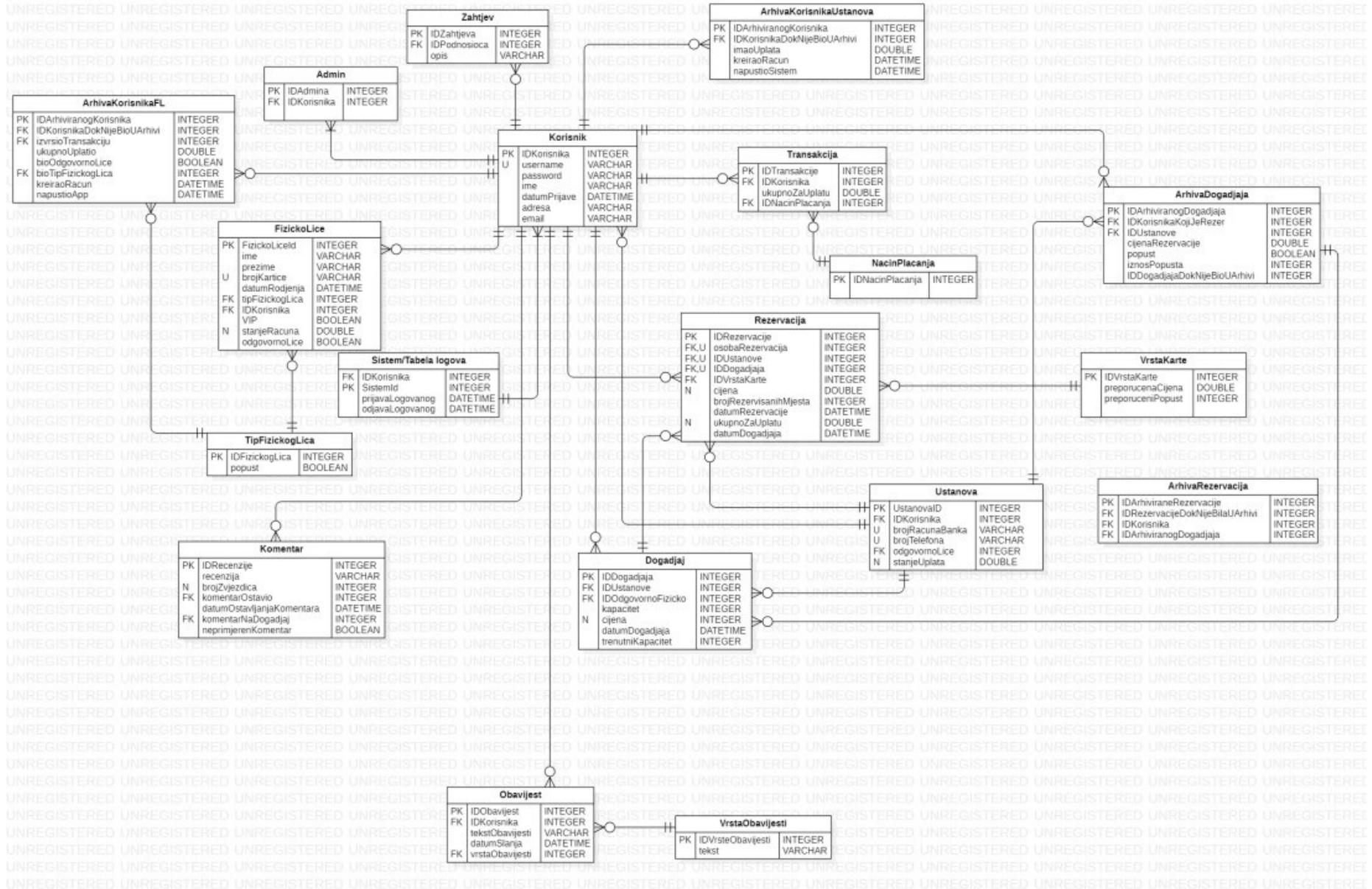
Dijagram komponenti



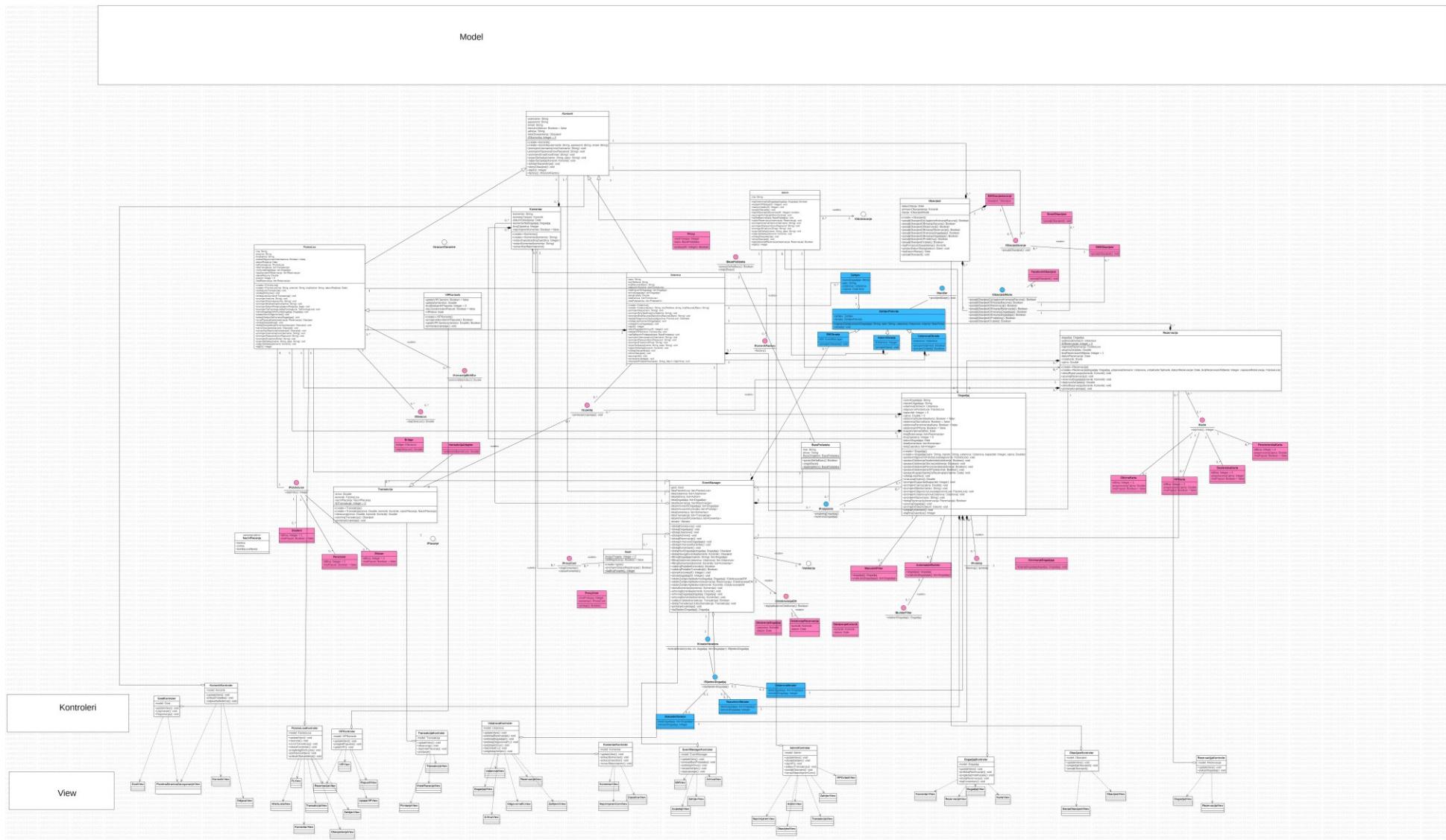
Dijagram složene strukture



ER dijagram baze podataka



MVC patern



WEB servis

Web servis je program koji koristi XML za razmjenu informacija sa drugim programima/softverima koristeći zajednički internet protokol. Jednostavno rečeno, web servis je vid interakcije sa objektima preko interneta. Primjeri Web servisa u aplikacijama su izvještaju vremenske prognoze, prikazivanje ažuriranja stanja berze, prikaz najsvježijih vijesti koristeći News Headline Web Service, pravljenje neke vrste web portala,... Isto tako, mi možemo biti oni koji pružaju tj. kreiraju web servis i koji daju drugima svoj web servis na korištenje. Možemo napraviti neki web servis i dati drugim kompanijama na korištenje, a da prilikom korištenja tog web servisa, svako ko ga bude koristio, indirektno pravi reklamu našoj kompaniji (našem proizvodu-web servisu).

U našem projektu, mi služimo kao posrednici između ustanova, koje organiziraju događaje, i ljudi koji žele doći na te događaje. Svrha naše aplikcije jeste informisanje o najnovijim vijestima i događajima iz ustanova. Pogodno bi bilo zbog toga napraviti web servis koji će omogućiti prikazivanje brzih, kratkih i najsvježijih novosti iz ustanova koje kreiraju profil na našem sistemu. Aplikacija ima mogućnost obavještavanja svojih korisnika, tako da i to možemo shvatiti kao primjenu web servisa. Preko naše aplikacije mogu i ustanove koje su unešene u bazu podataka da šalju obavijesti korisnicima naše aplikacije, tj. mogu da koriste naš web servis za oglašavanje na taj način. Na neki način možemo i interakciju sa bazom podataka posmatrati kao mogućnost web sistema, jer prilikom svakog logovanja korisnika na aplikaciju, to se dešava preko web sistema i još se na taj način vrši interakcija sa bazom.