

Specifikacija projekta

1. Osnovne informacije o sistemu

Naziv teme: Modern home (online prodaja namještaja)

Logo:



Naziv tima: Tim 32

Nastavna grupa: RI2

Link na repozitorij tima: https://github.com/OOAD-2023-2024/Tim32_Modern_home

Članovi tima:

1. Ali Ljaljak, 19371
2. Azra Pamuk, 19447
3. Šaban Zolj, 19333
4. Amil Ljaljak, 19207

Namjena sistema:

Opisati sistem i njegovu namjenu sa maksimalno sedam rečenica. U okviru ovog polja potrebno je objasniti šta sistem treba raditi na apstraktnom nivou, bez detaljnog objašnjavanja pojedinačnih funkcionalnosti i načina razlikovanja aktera sistema (što je predmet daljih poglavlja).

Modern home, sistem za online prodaju namještaja, treba omogućiti da korisnik ima uvid u trenutnu ponudu sistema, da može označiti proizvode koje želi kupiti, te da ih može naručiti. Sistem sadrži login/signup formu putem koje se mogu prijaviti potencijalni kupci, administrator i zaposleni u sistemu. Administratori u sistemu imaju mogućnosti uređivanja, dodavanja i uklanjanja artikala. Dostava namještaja funkcioniše preko vanjskog sistema dostave. Plaćanje je moguće izvršiti prilikom preuzimanja namještaja ili kartično. Također, sistem obezbjeđuje maksimalnu sigurnost korisničkih podataka i lozinki. Korisnici mogu ocijenjivati proizvode i na taj način omogućiti da najbolje ocijenjeni proizvodi budu preporučeni prilikom kupovine, a prilikom pretraživanja je moguće filtrirati ponudu po osnovu ocjena artikala i raznim drugim kriterijima.

2. Funkcionalnosti (poslovni procesi) sistema

Opisati 6 do 8 najznačajnijih funkcionalnosti sistema (u zavisnosti od broja članova u timu). Funkcionalnosti sistema predstavljaju usluge koje sistem pruža korisnicima. Sve funkcionalnosti pripadaju nekoj od različitih vrsta:

- Usluga sistema - u svrhu ostvarivanja krajnje usluge sistema,
- Perzistencija podataka (CRUD operacije)
- Asinhrona operacija - operacije koje koriste principe asinhronne obrade zahtjeva
- Operacija sa specifičnim algoritmom obrade - operacije koje koriste specifične algoritme obrade podataka,
- Korištenje vanjskog uređaja - operacije u kojima se vrši korištenje vanjskih uređaja.

Neophodno je navesti barem po jednu funkcionalnost svake od različitih vrsta.

1) **Naziv funkcionalnosti:** Log in/Sign up

Vrsta funkcionalnosti: Usluga sistema

Opis funkcionalnosti:

Opisati način ostvarivanja funkcionalnosti sa maksimalno pet rečenica.

Svi korisnici sistema imaju mogućnost prijave na sistem koristeći svoje pristupne podatke. Na formi za prijavu pojavljuje se nekoliko opcija: log in (za postojeće korisnike), sign up (za nove korisnike), log in as employee (za administratore i uposlene). Administratori imaju posebnu formu za prijavu koja se pojavljuje i koja zahtjeva specifične podatke, za razliku od prijave kupaca. Log in zahtjeva samo korisničko ime i lozinku, dok je sign up potrebna prijava putem emaila.

2) **Naziv funkcionalnosti:** Dodavanje/uređivanje/brisanje artikala i narudžbi

Vrsta funkcionalnosti: Perzistencija podataka (CRUD operacija)

Opis funkcionalnosti:

Opisati način ostvarivanja funkcionalnosti sa maksimalno pet rečenica.

Ovu operaciju može izvršavati samo administratorski korisnik. Ova operacija komunicira sa bazom podataka te omogućava dodavanje novog proizvoda ili narudžbe u bazu podataka (sistem), izmjenu postojećih proizvoda i narudžbi, te njihovo brisanje. Administrator može da specificira sve informacije vezane za proizvod ili narudžbu npr. naziv, dimenzije, boju, slike proizvoda... Na formi će postojati odgovarajući mehanizmi koji neće dozvoljavati unos besmislenih podataka u bazu tj. u sistem.

3) **Naziv funkcionalnosti:** Pretraživanje/filtriranje artikala

Vrsta funkcionalnosti: Perzistencija podataka (CRUD operacija)

Opis funkcionalnosti:

Opisati način ostvarivanja funkcionalnosti sa maksimalno pet rečenica.

Pristup ovoj funkcionalnosti imaju svi korisnici, zaposlenici i administratori, ali se razlikuje za korisnike, te zaposlenike i administratore. Korisnici mogu pretraživati po ključnim riječima, dok se filtriranje vrši na osnovu cijene, boje i sl. Zaposlenici i administratori mogu vršiti pretragu po svim osobinama artikla, a filtriranje se dodatno može vršiti i na osnovu toga da li je artikal na stanju ili ne.

4) **Naziv funkcionalnosti:** Naručivanje namještaja

Vrsta funkcionalnosti: Usluga sistema

Opis funkcionalnosti:

Opisati način ostvarivanja funkcionalnosti sa maksimalno pet rečenica.

Korisnik koji je napravio korisnički račun i unio odgovarajuće podatke ima mogućnost da odabere proizvode/artikle koje želi kupiti, te ih može naručiti. Prilikom odabira artikala, prije ubacivanja u korpu ima mogućnost da odabere karakteristike proizvoda npr. boja, veličina...U trenutku kada kupac označi da želi naručiti proizvod, uposlenik na svom profilu dobija zahtjev za isporuku. Tada se automatski reguliše i količina proizvoda u bazi podataka. Plaćanje se može izvršiti prilikom preuzimanja proizvoda ili kartično preko odgovarajuće forme na stranici sistema.

5) **Naziv funkcionalnosti:** Ocjenjivanje/pregled recenzija artikala

Vrsta funkcionalnosti: Usluga sistema

Opis funkcionalnosti:

Opisati način ostvarivanja funkcionalnosti sa maksimalno pet rečenica.

Svaki registrovani korisnik ima mogućnost ocjenjivanja proizvoda. Ocjenjivanje proizvoda funkcioniše na način da registrovani korisnici mogu ocijeniti samo one proizvode koje su naručivali u prošlosti ili ih trenutno naručuju. Skala ocjenjivanja uključuje ocjene 1 do 5, a korisniku se nudi da označi odgovarajući broj zvjezdica kako bi dao svoj sud o nekom artiklu. Prosječna ocjena svakog artikla će kasnije poslužiti za sistem preporuka. Prilikom pregleda artikala je moguće vidjeti prosječne ocjene artikala.

6) **Naziv funkcionalnosti:** Upravljanje korpom

Vrsta funkcionalnosti: Usluga sistema

Opis funkcionalnosti:

Opisati način ostvarivanja funkcionalnosti sa maksimalno pet rečenica.

Nakon što je korisnik ili gost dodao odgovarajuće artikle u korpu, nudi mu se opcija “pregledaj korpu”. Klikom na tu opciju ima mogućnost da pregleda sve artikle koje je ubacio u korpu, te da uredi informacije vezane za korpu. Može da ukloni određene proizvode iz korpe, da poveća ili smanji količinu nekog artikla, te da upravlja informacijama o proizvodu, poput dimenzija, boje... Kada korisnik izvrši odgovarajuće izmjene u vezi sa korpom, treba da klikne na opciju “sačuvaj promjene” kako bi potvrdio izmjene.

7) **Naziv funkcionalnosti:** Plaćanje

Vrsta funkcionalnosti: Asinhrona operacija

Opis funkcionalnosti:

Opisati način ostvarivanja funkcionalnosti sa maksimalno pet rečenica.

Jednom kada korisnik “ubaci u korpu” sve proizvode koji mu se sviđaju, ima mogućnost da pritisne dugme “Kreni na plaćanje”. Tada mu se pojavljuje odgovarajuća forma. Ukoliko korisnik nije uradio log in ili sign up tj. ukoliko stranicu posjećuje kao gost, od njega se zahtijeva da se prijavi na sistem. Tek tada ima mogućnost da unese svoje podatke, adresu na koju želi da mu se isporuči kupljeni namještaj, način plaćanja, itd. Ukoliko izabere kartični način plaćanja sistem ga vodi na posebnu formu koja služi za online plaćanje.

8) **Naziv funkcionalnosti:** Preporučivanje artikala

Vrsta funkcionalnosti: Operacija sa specifičnim algoritmom obrade

Opis funkcionalnosti:

Opisati način ostvarivanja funkcionalnosti sa maksimalno pet rečenica.

Ova funkcionalnost omogućava da se registrovanom korisniku, prilikom pristupa sistemu, preporučuju određeni proizvodi, za koje algoritam odredi da bi mu se trebali svidjeti. Algoritam koji vrši preporuku radi na osnovu ocjena proizvoda, sličnosti sa proizvodima kojima korisnik trenutno pristupa. Ta sličnost se ogleda u cijeni, sličnom tipu namještaja, vrsti gradivnog materijala, sličnim bojama, itd. Preporuke se pojavljuju u vidu dodatnih elmenata na stranici i omogućavaju korisniku da jednostavno pristupi detaljnim informacijama o preporučenim proizvodima. Administrator također može označiti odgovarajuće proizvode koji će se pojavljivati kao preporučeni.

3. Akteri sistema

Potrebno je navesti najmanje tri aktera sistema.

Vrste aktera:

- Korisnik sistema
- Zaposlenik sistema
- Administrator

Neophodno je navesti barem po jednog aktera za svaku od različitih vrsta.

Korisnici usluga sistema

a) **Naziv aktera: Administrator**

Vrsta aktera: Administrator

Funkcionalnosti u kojima akter učestvuje:

Funkcionalnost sistema	Način učešća
1- Log in/Sign up	Mogućnost uređivanja
2- Dodavanje/uređivanje/brisanje artikala i narudžbi	Mogućnost uređivanja
3- Pretraživanje/filtriranje artikala	Mogućnost uređivanja
8-Preporučivanje artikala	Mogućnost uređivanja

b) **Naziv aktera: Zaposlenik**

Vrsta aktera: Zaposlenik sistema

Funkcionalnosti u kojima akter učestvuje:

Funkcionalnost sistema	Način učešća
1- Log in/Sign up	Mogućnost uređivanja
3- Pretraživanje/filtriranje artikala	Mogućnost pregleda
5-Ocjenjivanje/pregled recenzija artikala	Mogućnost pregleda

c) **Naziv aktera: Registrovani korisnik**

Vrsta aktera: Korisnik sistema

Funkcionalnosti u kojima akter učestvuje:

Funkcionalnost sistema	Način učešća
1- Log in/Sign up	Mogućnost uređivanja
3- Pretraživanje/filtriranje artikala	Mogućnost pregleda
4-Naručivanje namještaja	Mogućnost uređivanja
5-Ocjenjivanje/pregled recenzija artikala	Mogućnost uređivanja
6-Upravljanje korpom	Mogućnost uređivanja
7-Plaćanje	Mogućnost uređivanja
8-Preporučivanje artikala	Mogućnost pregleda

d) **Naziv aktera: Gost**

Vrsta aktera: Korisnik sistema

Funkcionalnosti u kojima akter učestvuje:

Funkcionalnost sistema	Način učešća
3- Pretraživanje/filtriranje artikala	Mogućnost pregleda
5-Ocjenjivanje/pregled recenzija artikala	Mogućnost pregleda
6-Upravljanje korpom	Mogućnost uređivanja

4. Nefunkcionalni zahtjevi sistema

Opisati najmanje tri najznačajnija nefunkcionalna zahtjeva sistema. Nefunkcionalni zahtjevi predstavljaju ograničenja koja sistem mora zadovoljiti kako bi mogao ispravno obavljati svoje funkcionalnosti. Validacije polja za unos vrijednosti ne predstavljaju nefunkcionalne zahtjeve.

1) **Naziv nefunkcionalnog zahtjeva:** Performanse sistema

Opis:

Opisati ograničenje sistema i način na koje se ono ispoljava.

Sistem treba biti sposoban da brzo odgovara na korisničke zahtjeve, kao što su pretraga proizvoda, dodavanje proizvoda u korpu i procesiranje narudžbi. Ovaj zahtjev uključuje specifikacije za brzinu učitavanja stranica, minimalno vrijeme čekanja na odgovor prilikom obrade transakcija i efikasnost baze podataka u obradi upita. Recimo zahtjev da se stranice trebaju učitati unutar 2-3 sekunde osigurava da korisnici neće napustiti stranicu zbog dugog čekanja i da će imati pozitivno iskustvo. Sistem treba odgovoriti na korisnički zahtjev za dodavanjem proizvoda u korpu ili procesiranjem narudžbe unutar određenog vremenskog okvira, recimo, manje od 1 sekunde. Ovo osigurava da korisnici neće doživjeti frustraciju zbog dugog čekanja prilikom obavljanja

2) **Naziv nefunkcionalnog zahtjeva:** Sigurnost podataka

Opis:

Opisati ograničenje sistema i način na koje se ono ispoljava.

Lozinke korisnika su sigurno skladištene kako bi se spriječilo njihovo kompromitovanje u slučaju napada na sistem. Koriste se hash funkcije zajedno s jedinstvenim "salt" vrijednostima prilikom pohranjivanja lozinke korisnika, što osigurava da čak i ako se desi napad na bazu podataka, lozinke neće biti lako provaljene.

3) **Naziv nefunkcionalnog zahtjeva:** Dostupnost i skalabilnost

Opis:

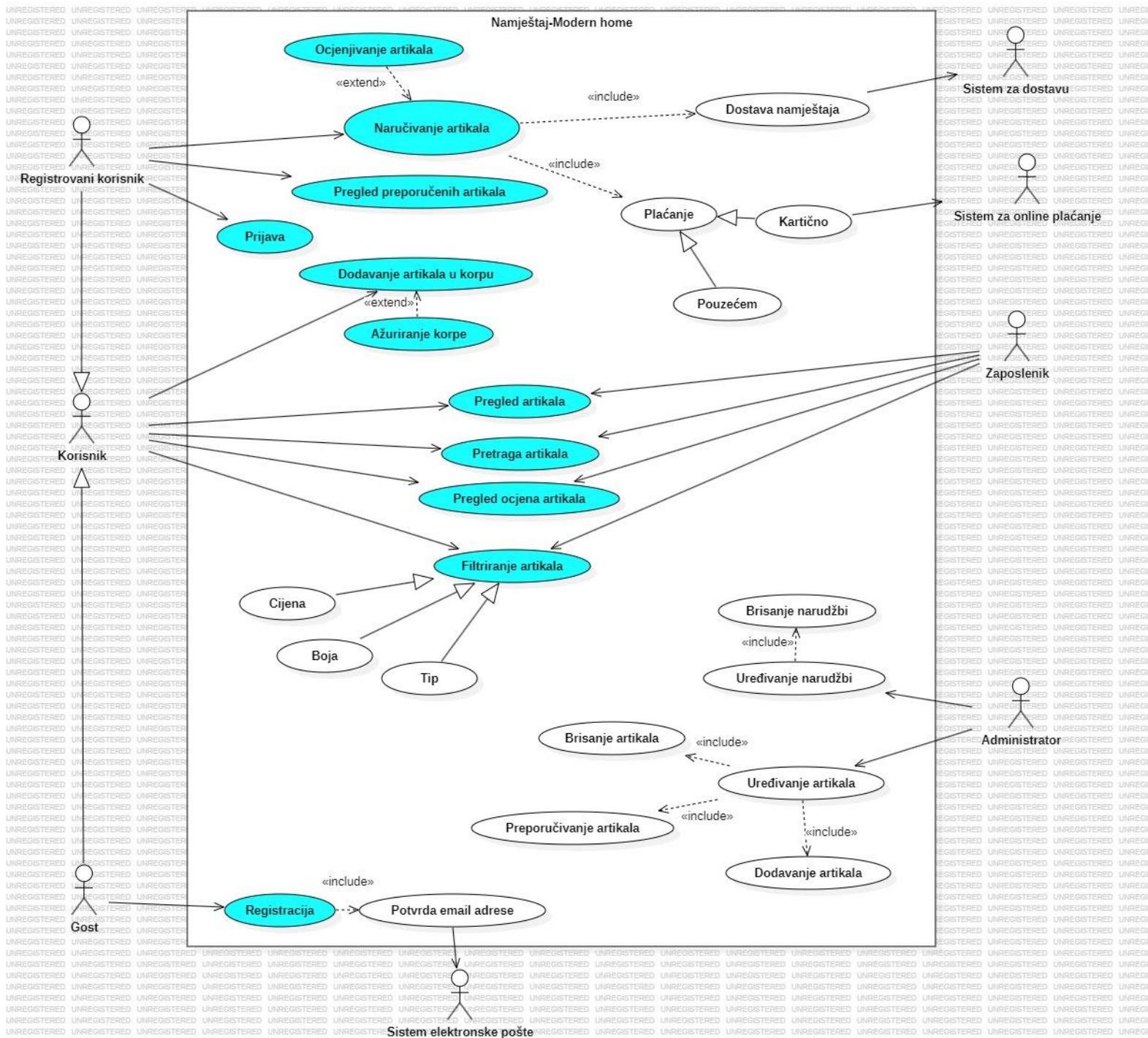
Opisati ograničenje sistema i način na koje se ono ispoljava.

Može se očekivati da sistem bude dostupan 99.9% vremena tokom godine, što osigurava da korisnici mogu pristupiti online shopu sa minimalnim prekidima u radu.

Use Case Diagram

Modern Home

Online Shop Namještaja



Scenariji slučajeva upotrebe

1. Log in

Naziv	Log in
Opis	Učesnici sistema (administrator, uposlenik, registrovani korisnik) imaju mogućnost da se prijave na sistem gdje treba da unesu svoje podatke poput email-a, passworda, itd. Korisnik(gost) koji nije kreirao svoj račun se ne može uspješno prijaviti.
Vezani zahtjevi	/
Preduslovi	Kreiranje korisničkog računa (sign up)
Posljedice-uspješan završetak	Prijava na aplikaciju, mogućnost naručivanja i ocjenjivanja artikala.
Posljedice-neuspješan završetak	Ograničene mogućnosti prilikom korištenja aplikacije.
Primarni akteri	Administrator, uposlenik, registrovani korisnik
Ostali akteri	/
Glavni tok	Otvaranje aplikacije, klik na dugme "Log in", ispunjavanje forme, pristup svim mogućnostima koje nudi korisnički račun
Proširenja/Alternative	Ukoliko korisnik zaboravi šifru može je obnoviti preko email-a. Nakon toga se sve nastavlja kao u glavnom toku.

Tok događaja 1: **Uspješan završetak**

Korisnik	Sistem
1. Korisnik pristupa stranici, te ide na opciju za prijavu	
2. Korisnik popunjava odgovarajuće podatke potrebne za prijavu	
	3. Sistem provjera da li postoji korisnički račun koji odgovara unesenim podacima
	4. Kada sistem ustanovi da su podaci tačni, korisniku biva odobren pristup sa svim benefitima koje registrovani korisnik ima

Tok događaja 2: **Neuspješan završetak**

Korisnik	Sistem
1. Korisnik pristupa stranici, te ide na opciju za prijavu	
2. Korisnik popunjava odgovarajuće podatke potrebne za prijavu	
	3. Sistem provjera da li postoji korisnički račun koji odgovara unesenim podacima
	4. Nakon što sistem utvrdi da su uneseni podaci netačni (nepostojeći korisnik ili pogrešna lozinka), korisniku biva prikazana poruka o tome, te i dalje ima privilegije gosta

2. Sign up

Naziv	Sign up
Opis	Korisnik koji prvi put pristupa sistemu ili pristupa sistemu kao gost može kreirati korisnički račun. Potrebno je da ispuni odgovarajuću formu koja svojim podacima kao što su ime, prezime, e-mail, password, itd.
Vezani zahtjevi	/
Preduslovi	Ne smije postojati račun sa istim pristupnim podacima
Posljedice-uspješan završetak	Kreiranje korisničkog računa, mogućnost naručivanja i ocjenjivanja proizvoda.
Posljedice-neuspješan završetak	Nemogućnost kreiranja korisničkog naloga, prikaz greške, neke mogućnosti sistema ostaju zaključane.
Primarni akteri	Korisnici gosti
Ostali akteri	/
Glavni tok	Pristup sistemu, klik na dugme “sign in”, ispunjavanje forme, prijava na kreirani korisnički račun
Proširenja/Alternative	/

Tok događaja 1: **Uspješan završetak**

Korisnik	Sistem
1. Korisnik pristupa stranici, te ide na opciju za registraciju	
2. Korisnik popunjava odgovarajuće podatke potrebne za registraciju	
	3. Sistem provjera da li već postoji korisnički račun sa unesenim podacima
	4. Nakon što se utvrdi da korisnik sa unesenim korisničkim imenom ne postoji, novi korisnik se registruje

Tok događaja 2: **Neuspješan završetak**

Korisnik	Sistem
1. Korisnik pristupa stranici, te ide na opciju za registraciju	
2. Korisnik popunjava odgovarajuće podatke potrebne za registraciju	
	3. Sistem provjera da li već postoji korisnički račun sa unesenim podacima
	4. Korisnik sa tim korisničkim imenom već postoji, pa neće biti kreiran novi račun, te će se ispisati odgovarajuća poruka o tome

3. Dodavanje/uređivanje/brisanje artikala

Naziv	Dodavanje/uređivanje/brisanje artikala
Opis	Administrator sistema ima mogućnost uređivanja artikala, njihovog brisanja i dodavanja.
Vezani zahtjevi	Dodavanje artikala Brisanje artikala Ažuriranje artikala
Preduslovi	Prijava administratora na aplikaciju
Posljedice-uspješan završetak	Izmjena upisa u bazi podataka, prikaz izmjena prilikom pretraživanja artikala
Posljedice-neuspješan završetak	Nepravilno prikazana lista artikala, podaci u bazi podataka nisu relevantni, nikakve promjene u listi artikala usljed tehničkih poteškoća (prekid veze sa serverom i sl.)
Primarni akteri	Administrator
Ostali akteri	/
Glavni tok	Prijava administratora na sistem, odabir odgovarajuće opcije za uređivanje artikala, izvršenje izmjena artikala, klik na "submit"
Proširenja/Alternative	/

Tok događaja 1: **Uspješan završetak (dodavanje artikala)**

Administrator	Sistem
1. Administrator pristupa stranici, te bira opciju za uređivanje artikala	
2. Bira opciju za dodavanje novog artikla, te unosi potrebne informacije	
	5. Sistem provjera da li postoji konflikt sa novim artiklom i već postojećim u bazi podataka (da li artikal već postoji) i da li su uneseni podaci adekvatni
	6. Nakon što se utvrdi da je sve uredu, artikal biva dodan u bazu podataka, ujedno i na stranicu

Tok događaja 2: **Neuspješan završetak (dodavanje artikala)**

Administrator	Sistem
1. Administrator pristupa stranici, te bira opciju za uređivanje artikala	
2. Bira opciju za dodavanje novog artikla, te unosi potrebne informacije	
	3. Sistem provjera da li postoji konflikt sa novim artiklom i već postojećim u bazi podataka (da li artikal već postoji) i da li su uneseni podaci adekvatni
	4. Nakon što se utvrdi da je artikal već postoji/ili ako su podaci netačni, otkazuje se dodavanje artikla

Tok događaja 1: **Uspješan završetak (uređivanje artikala)**

Administrator	Sistem
1. Administrator pristupa stranici, te bira opciju za uređivanje artikala	
2. Bira proizvod čije podatke želi da promijeni, te vrši odgovarajuće izmjene	
	3. Sistem provjera da li postoji konflikt sa izmijenjenim artiklom i već postojećim u bazi podataka i da li su uneseni podaci adekvatni
	4. Nakon što se utvrdi da je sve uredu, aizmjene se spašavaju

Tok događaja 2: **Neuspješan završetak (uređivanje artikala)**

Administrator	Sistem
1. Administrator pristupa stranici, te bira opciju za uređivanje artikala	
2. Bira proizvod čije podatke želi da promijeni, te vrši odgovarajuće izmjene	
	3. Sistem provjera da li postoji konflikt sa izmijenjenim artiklom i već postojećim u bazi podataka i da li su uneseni podaci adekvatni

	4. Utvrdi se da su izmijenjeni podaci doveli do kompromisa u bazi podataka, pa se izmjene odbacuju
--	--

Tok događaja 1: **Uspješan završetak (brisanje artikala)**

Administrator	Sistem
1. Administrator pristupa stranici, te bira opciju za uređivanje artikala	
2. Bira proizvod koji želi da obriše, nakon čega bira opciju za brisanje pojedinog artikla	
	3. Sistem uklanja podatke vezano za taj proizvod sa stranice i iz baze podataka
	4. Artikal je uspješno obrisao

Tok događaja 2: **Neuspješan završetak (brisanje artikala)**

Administrator	Sistem
1. Administrator pristupa stranici, te bira opciju za uređivanje artikala	
2. Bira proizvod koji želi da obriše, nakon čega bira opciju za brisanje pojedinog artikla	
	3. U toku procesa brisanja artikla, dolazi do greške (prekid veze), pa se proces brisanja prekida i stranica se vraća na stanje prije zahtjeva za brisanjem
	4. Artikal nije obrisao

4. Dodavanje/uređivanje/brisanje narudžbi

Naziv	Dodavanje/uređivanje/brisanje narudžbi
Opis	Administrator sistema ima mogućnost uređivanja parametara narudžbi, njihovog brisanja i dodavanja novih narudžbi.
Vezani zahtjevi	Dodavanje narudžbi Izmjena parametara narudžbi Brisanje narudžbi
Preduslovi	Prijava administratora na aplikaciju
Posljedice-uspješan završetak	Ažurirana lista narudžbi spremna za slanje sistemu za dostavu
Posljedice-neuspješan završetak	Lista narudžbi nije relevantna, neuspješno ažuriranje u bazi podataka
Primarni akteri	Administrator
Ostali akteri	Sistem za dostavu
Glavni tok	Administrator se prijavljuje na aplikaciju, bira opciju manipulacije narudžbama, izvršava odgovarajuće izmjene, spašava izmjene
Proširenja/Alternative	/

Tok događaja 1: **Uspješan završetak (dodavanje narudžbi)**

Administrator	Sistem
1. Administrator pristupa stranici, te bira opciju za uređivanje narudžbi	
2. Bira koju narudžbu želi da uredi, te mijenja odgovarajuće podatke	
	3. Sistem provjera da li postoji konflikt sa izmijenjenom narudžbom i već postojećim u bazi podataka, i da li su uneseni podaci adekvatni
	4. Nakon što se utvrdi da je sve uredu, izmjene se spašavaju

Tok događaja 2: **Neuspješan završetak (dodavanje narudžbi)**

Administrator	Sistem
1. Administrator pristupa stranici, te bira opciju za uređivanje narudžbi	
2. Bira koju narudžbu želi da uredi, te mijenja odgovarajuće podatke	
	3. Sistem provjera da li postoji konflikt sa izmijenjenom narudžbom i već postojećim u bazi podataka, i da li su uneseni podaci adekvatni
	4. Utvrđi se da je došlo do konflikta, ili da su podaci netačni pa se izmjene odbacuju

Tok događaja 1: **Uspješan završetak (uređivanje narudžbi)**

Administrator	Sistem
1. Administrator pristupa stranici, te bira opciju za uređivanje narudžbi	
2. Bira koju narudžbu želi da uredi, te mijenja odgovarajuće podatke	
	3. Sistem provjera da li postoji konflikt sa izmijenjenom narudžbom i već postojećim u bazi podataka, i da li su uneseni podaci adekvatni
	4. Utvrđi se da je došlo do konflikta, ili da su podaci netačni pa se izmjene odbacuju

Tok događaja 2: **Neuspješan završetak (uređivanje narudžbi)**

Administrator	Sistem
1. Administrator pristupa stranici, te bira opciju za uređivanje artikala	
2. Bira proizvod čije podatke želi da promijeni, te vrši odgovarajuće izmjene	
	3. Sistem provjera da li postoji konflikt sa izmijenjenim artiklom i već postojećim u bazi podataka i da li su uneseni podaci adekvatni
	4. Utvrđi se da su izmijenjeni podaci doveli do kompromisa u bazi podataka, pa se izmjene odbacuju

Tok događaja 1: **Uspješan završetak (brisanje narudžbi)**

Administrator	Sistem
1. Administrator pristupa stranici, te bira opciju za uređivanje narudžbi	
2. Bira narudžbu koju želi da obriše, nakon čega bira opciju za brisanje odabrane narudžbe	
	3. Sistem uklanja podatke vezano za tu narudžbu sa stranice i iz baze podataka
	4. Narudžba je uspješno obrisana

Tok događaja 2: **Neuspješan završetak (brisanje narudžbi)**

Administrator	Sistem
1. Administrator pristupa stranici, te bira opciju za uređivanje narudžbi	
2. Bira narudžbu koju želi da obriše, nakon čega bira opciju za brisanje odabrane narudžbe	
	3. U toku procesa brisanja narudžbe, dolazi do greške (prekid veze), pa se proces brisanja prekida i stranica se vraća na stanje prije zahtjeva za brisanjem
	4. Narudžba nije obrisana

5. Pretraživanje/filtriranje artikala

Naziv	Pretraživanje/filtriranje artikla
Opis	Administrator, registrovani korisnici, gosti i uposlenici imaju mogućnost pretrage artikala, te filtriranja po nekim njihovim ključnim osobinama
Vezani zahtjevi	Pretraga artikala Filtriranje artikala na osnovu nekih kriterija
Preduslovi	Pristup sistemu
Posljedice-uspješan završetak	Prikaz artikala od interesa
Posljedice-neuspješan završetak	Nepravilan prikaz liste artikala
Primarni akteri	Administrator, uposlenici, registrovani korisnici i gosti
Ostali akteri	/
Glavni tok	Otvaranje aplikacije, pristup listi proizvoda, navigacija do tražilice, odabir parametara filtriranja i pretrage, ispis artikala od interesa
Proširenja/Alternative	/

Tok događaja 1: **Uspješan završetak (pretraživanje artikala)**

Administrator/Zaposlenik/Korisnik/Gost	Sistem
1. Korisnik pristupa sistemu	
2. U search bar unosi pojmove na osnovu kojih vrši pretragu	
	3. Na osnovu ključnih riječi formira se lista artikala koji odgovaraju pretrazi
	4. Prikaz artikala koji odgovaraju zahtjevu pretrage
5. Korisnik ima pregled proizvoda na osnovu pretrage	

Tok događaja 2: **Neuspješan završetak (pretraživanje artikala)**

Administrator/Zaposlenik/Korisnik/Gost	Sistem
1. Korisnik pristupa sistemu	
2. U search bar unosi pojmove na osnovu kojih vrši pretragu	
	3. Usljed neadekvatnog algoritma za pretragu, formira se lista pogrešnih artikala
	4. Prikaz artikala koji ne odgovaraju zahtjevu pretrage
5. Korisnik ima pregled proizvoda koje nije pretraživao	

Tok događaja 1: **Uspješan završetak (filtriranje artikala)**

Administrator/Zaposlenik/Korisnik/Gost	Sistem
1. Korisnik pristupa sistemu	
2. U search bar unosi pojmove na osnovu kojih vrši pretragu, i zatim ima opciju filtriranja na osnovu boje, tipa i sl.	
	3. Na osnovu ključnih riječi i parametara filtriranja formira se lista artikala koji odgovaraju pretrazi
	4. Prikaz artikala koji odgovaraju zahtjevu pretrage
5. Korisnik ima pregled proizvoda na osnovu pretrage	

Tok događaja 2: **Neuspješan završetak (filtriranje artikala)**

Administrator/Zaposlenik/Korisnik/Gost	Sistem
1. Korisnik pristupa sistemu	
2. U search bar unosi pojmove na osnovu kojih vrši pretragu, i zatim ima opciju filtriranja na osnovu boje, tipa i sl.	
	3. Usljed neadekvatnog algoritma za pretragu i filtriranje, formira se lista pogrešnih artikala
	4. Prikaz artikala koji ne odgovaraju zahtjevu pretrage
5. Korisnik ima pregled proizvoda koje nije pretraživao, niti koji zadovoljavaju parametre filtriranja	

6. Naručivanje namještaja

Naziv	Naručivanje namještaja
Opis	Registrovani korisnici imaju mogućnost dodavanja artikala u korpu, te njihovog naručivanja uz popunjavanje forme gdje se unose podaci potrebni za dostavu i plaćanje
Vezani zahtjevi	Dodavanje artikala u korpu Manipulacija korpom Plaćanje
Preduslovi	Korisnik mora biti registrovan na sistem
Posljedice-uspješan završetak	Narudžba se pridružuje listi narudžbi i prosljeđuje se sistemu dostave
Posljedice-neuspješan završetak	Narudžba odbijena, ne spašava se u sistemu, artikli nisu naručeni, niti dostavljeni
Primarni akteri	Registrovani korisnici
Ostali akteri	Administrator, Sistem za dostavu
Glavni tok	Registrovani korisnik pristupa listi proizvoda, označava artikle od interesa, dodaje je ih u korpu, pregleda korpu, ispunjava odgovarajuću formu, naručuje proizvode
Proširenja/Alternative	Otkazivanje narudžbe nakon pregleda korpe

Tok događaja 1: **Uspješan završetak**

Korisnik	Sistem
1. Korisnik pristupa sistemu, vrši dodavanje artikala u korpu	
2. Klikom na korpu i prelaskom na formu za plaćanje, narudžba se šalje	
	3. Provjeravaju se podaci o dostavi
	4. Ako su podaci ispravni, narudžba se dodaje u bazu

Tok događaja 2: **Neuspješan završetak**

Korisnik	Sistem
1. Korisnik pristupa sistemu, vrši dodavanje artikala u korpu	
2. Klikom na korpu i prelaskom na formu za plaćanje, narudžba se šalje	
	5. Provjeravaju se podaci o dostavi
	6. Podaci su neispravni, narudžba se odbacuje

7. Ocjenjivanje/pregled recenzija artikala

Naziv	Ocjenjivanje/pregled recenzija artikala
Opis	Korisnik koji je naručio artikle ima mogućnost ocjene naručenih artikala. Ocjene se kreću od 1 do 5 i ocjenjivanje se vrši označavanjem odgovarajućeg broja zvjezdica. Zaposlenici, korisnici i gosti imaju mogućnost pregleda ocjena.
Vezani zahtjevi	Pregled recenzija artikala
Preduslovi	Korisnik mora biti registrovan i mora izvršiti kupovinu proizvoda da bi ostavio ocjenu
Posljedice-uspjeh završetak	Ostavljena ocjena proizvoda koja ulazi u prosječnu ocjenu za taj artikal
Posljedice-neuspjeh završetak	Ocjena nije ostavljena
Primarni akteri	Registrovani korisnici
Ostali akteri	/
Glavni tok	Korisnik pregleda artikle, stavlja ih u korpu, izvršava naručivanje i nakon toga ima mogućnost ostavljanja ocjene naručenih proizvoda
Proširenja/Alternative	Ostavljanje komentara

Tok događaja 1: **Uspješan završetak (ocjenjivanje artikala)**

Korisnik	Sistem
1. Korisnik pristupa sistemu, vrši barem jednu uspješnu kupovinu	
2. Nakon toga mu se omogućava opcija da ostavi recenziju na proizvode koje je prethodno kupio	
3. Ostavlja ocjenu od 1 do 5	
	4. Ocjena se spašava i dodaje u prosječnu ocjenu proizvoda

Tok događaja 2: **Neuspjeh završetak (ocjenjivanje artikala)**

Korisnik	Sistem
1. Korisnik pristupa sistemu, vrši barem jednu uspješnu kupovinu	
2. Nakon toga mu se omogućava opcija da ostavi recenziju na proizvode koje je prethodno kupio	
3. Ostavlja ocjenu od 1 do 5	
	4. Usljed greške sistema ili neadekvatne ocjene, neće doći do dodavanja

	recenzije u bazu i uračunavanje u prosječnu ocjenu
--	--

Tok događaja 1: **Uspješan završetak (pregled recenzija)**

Zaposlenik/Korisnik/Gost	Sistem
1. Korisnik pristupa sistemu	
2. Bira artikal i otvara se forma koja prikazuje detalje artikla	
	3. Daje podatke o artiklu, uključujući i prosječnu ocjenu
4. Mogućnost pregleda recenzije	

Tok događaja 2: **Neuspješan završetak (pregled recenzija)**

Zaposlenik/Korisnik/Gost	Sistem
1. Korisnik pristupa sistemu	
2. Bira artikal i otvara se forma koja prikazuje detalje artikla	
	3. Zbog pogrešnog proračuna, daju se pogrešni podaci o artiklu, uključujući i prosječnu ocjenu
4. Pregled pogrešnih podataka	

8. Upravljanje korpom

Naziv	Upravljanje korpom
Opis	Registrovani korisnici i gosti imaju mogućnost dodavanja artikala koje žele da naruče u korpu, iz koje isto tako mogu brisati iste ako se predomisle.
Vezani zahtjevi	Dodavanje artikala u korpu Izmjena parametara artikala u korpi Brisanje artikala iz korpe
Preduslovi	Pristup sistemu, mora biti registrovani korisnik ili gost
Posljedice-uspješan završetak	Artikal je dodan u korpu, može biti naručen / artikal je uspješno obrisano iz korpe, neće biti naručen
Posljedice-neuspješan završetak	Artikal nije dodan u korpu/artikal nije obrisano iz korpe
Primarni akteri	Registrovani korisnici i gosti
Ostali akteri	/
Glavni tok	Korisnik pristupa listi artikala, odabirom opcije “Dodaj u korpu” dodaje artikal u korpu, klikom na ikonicu korpe uređuje količinu artikala ili ih eventualno u potpunosti uklanja, nakon čega nastavlja na potvrdu narudžbe
Proširenja/Alternative	/

Tok događaja 1: **Uspješan završetak (dodavanje artikala u korpu)**

Korisnik/Gost	Sistem
1. Korisnik pristupa sistemu	
2. Bira artikal i dodaje ga u korpu	
	3. Dodaje artikal u korpu, spašava u bazu podataka

Tok događaja 2: **Neuspješan završetak (dodavanje artikala u korpu)**

Korisnik/Gost	Sistem
1. Korisnik pristupa sistemu	
2. Bira artikal i dodaje ga u korpu	
	3. Usljed greške u implementaciji ili prekida veze, artikal ne biva dodan u korpu

Tok događaja 1: **Uspješan završetak (uređivanje/brisanje artikala u korpi)**

Korisnik/Gost	Sistem
1. Korisnik pristupa sistemu	
2. Bira artikal i dodaje ga u korpu	
3. Mogućnost uređivanja/brisanja dobija odabirom ikonice korpe, nakon čega ima mogućnost da podesi specifikacije proizvoda (boja, količina i sl.), te da ga u potpunosti ukloni iz korpe	
	4. Artikal u korpi se uspješno ažurira/briše

Tok događaja 2: **Neuspješan završetak (uređivanje/brisanje artikala u korpi)**

Korisnik/Gost	Sistem
1. Korisnik pristupa sistemu	
2. Bira artikal i dodaje ga u korpu	
3. Mogućnost uređivanja/brisanja dobija odabirom ikonice korpe, nakon čega ima mogućnost da podesi specifikacije proizvoda (boja, količina i sl.), te da ga u potpunosti ukloni iz korpe	

	4. Usljed internih grešaka ili konekcionih problema, artikal u korpi ne biva izmijenjen/obrisan
--	---

9. Plaćanje

Naziv	Plaćanje
Opis	Nakon dodavanja artikala korpu, registrovani korisnik nastavlja na potvrdu narudžbe i odabira način plaćanja iste.
Vezani zahtjevi	Upravljanje korpom Narudžba artikala
Preduslovi	Korisnik mora biti registrovan, mora dodati barem jedan artikal u korpu, te kliknuti na dugme “Nastavi na plaćanje”
Posljedice-uspješan završetak	Korisnik uspješno unosi potrebne podatke, artikli će biti naručeni i dostavljeni
Posljedice-neuspješan završetak	Korisnik ne unosi tačne podatke, proces se prekida, artikli se ne naručuju
Primarni akteri	Registrovani korisnik
Ostali akteri	Sistem za online plaćanje
Glavni tok	Dodavanje artikala u korpu, otvaranje forme uređivanja korpe, klik na opciju za prelazak na formu za plaćanje, popunjavanje odgovarajućih podataka, uspješno naručeni artikli i dodavanje narudžbe u listu narudžbi
Proširenja/Alternative	/

Tok događaja 1: **Uspješan završetak**

Korisnik/Gost	Sistem
1. Korisnik pristupa sistemu	
2. Bira artikle, dodaje ih u korpu	
3. Prelazi na formu za plaćanje, gdje unosi potrebne podatke (u slučaju da je u pitanju kartično plaćanje)	
	4. Podaci se provjeravaju, te nakon potvrde da su ispravni, narudžba se dodaje u sistem, artikli se uspješno dostavljaju

Tok događaja 2: **Neuspješan završetak**

Korisnik/Gost	Sistem
1. Korisnik pristupa sistemu	
2. Bira artikle, dodaje ih u korpu	
3. Prelazi na formu za plaćanje, gdje unosi potrebne podatke (u slučaju da je u pitanju kartično plaćanje)	
	5. Podaci se provjeravaju, utvrđuje se da su neispravni, zahtjev za narudžbu se otkazuje

10. Preporučivanje artikala

Naziv	Preporučivanje artikala
Opis	Dok registrovani korisnici vrše pregled pojedinih artikala, u dnu stranice će se nalaziti pregled “Preporučenih artikala”, na osnovu prethodno pregledanih artikala ili nekih sličnih karakteristika sa artiklom koji trenutno gleda.
Vezani zahtjevi	Pregled recenzija artikala
Preduslovi	Mora biti registrovani korisnik
Posljedice-uspjeh završetak	Korisniku će se prikazati odgovarajući artikli
Posljedice-neuspjeh završetak	Korisniku će se prikazivati irelevantni artikli
Primarni akteri	Registrovani korisnik
Ostali akteri	/
Glavni tok	Korisnik pristupa stranici, vrši pregled liste artikala, bira pojedini artikal, otvara se stranica koja će prikazati taj specifični artikal, na dnu prozora ima pristup preporučenim artiklima
Proširenja/Alternative	/

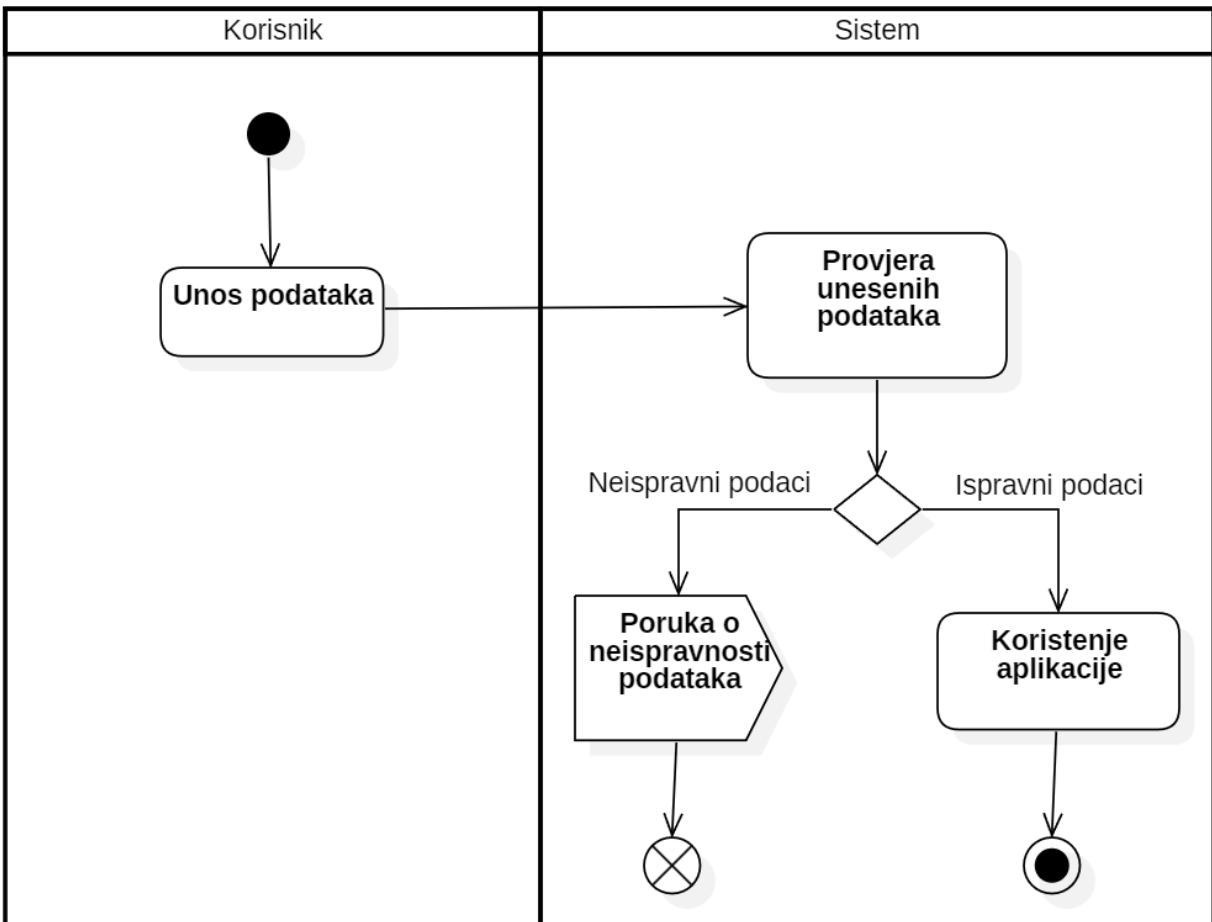
Tok događaja 1: **Uspješan završetak**

Korisnik	Sistem
1. Korisnik pristupa sistemu	
2. Bira artikle, vrši detaljniji pregled	
	3. Sistem vraća listu artikala na osnovu prethodno kupljenih artikala, te artikla koji se trenutno gleda
4. Korisnik ima pregled relevantnih artikala	

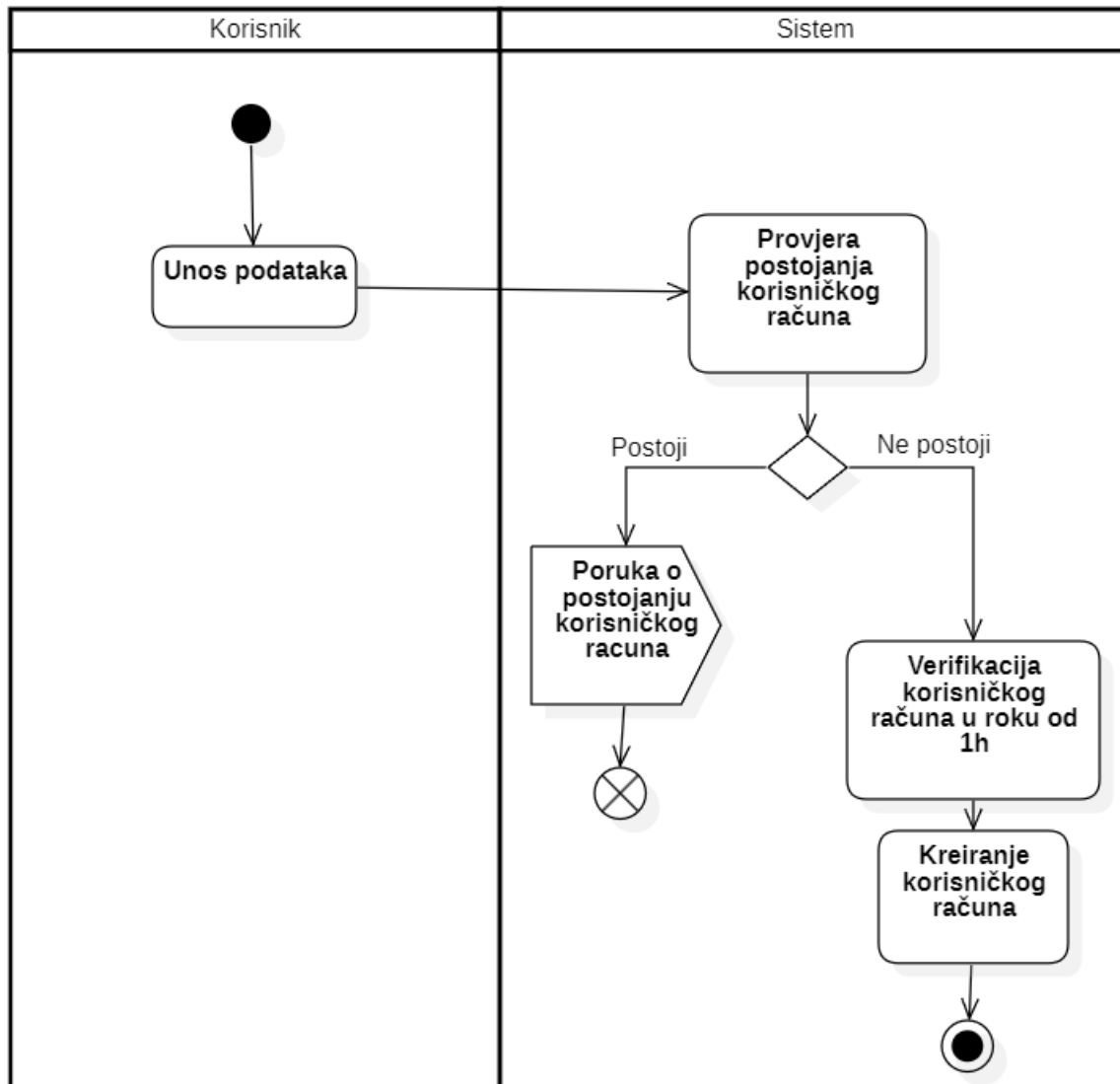
Tok događaja 2: **Neuspješan završetak**

Korisnik	Sistem
1. Korisnik pristupa sistemu	
2. Bira artikle, vrši detaljniji pregled	
	3. Sistem vraća listu pogrešno određenih artikala usljed neadekvatnog algoritma
4. Korisnik ima pregled irelevantnih artikala	

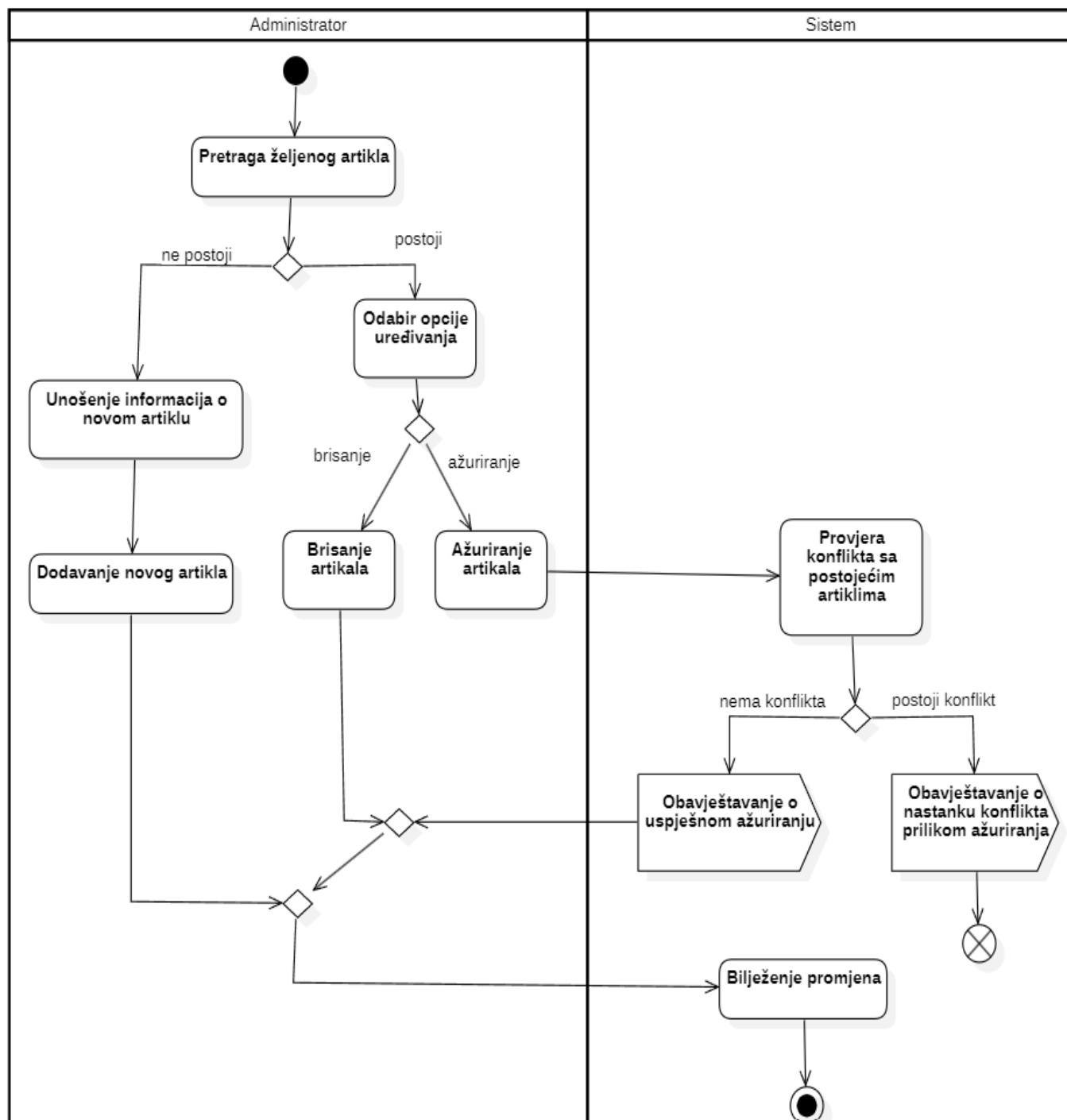
1. Log in



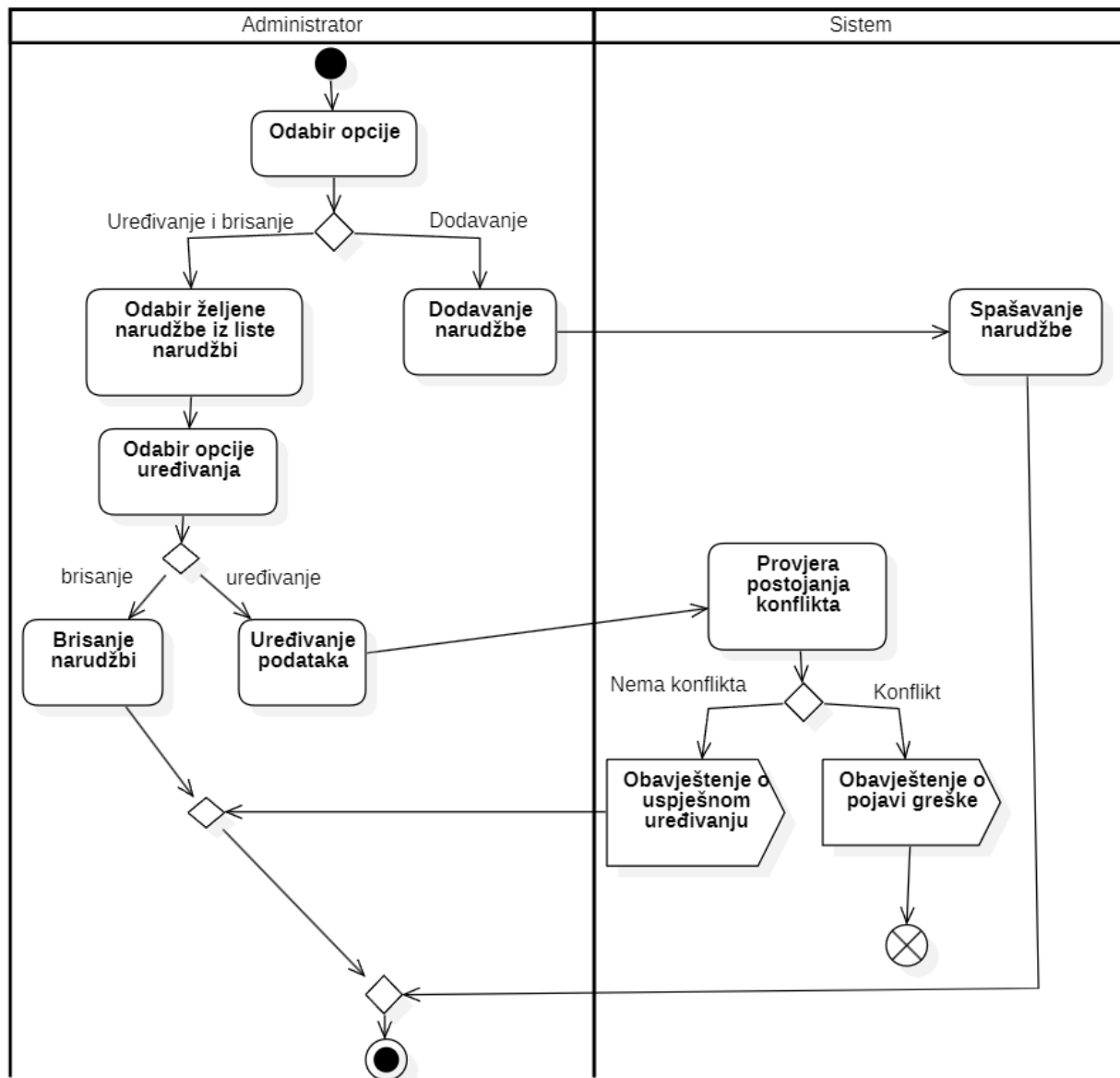
2. Sign up



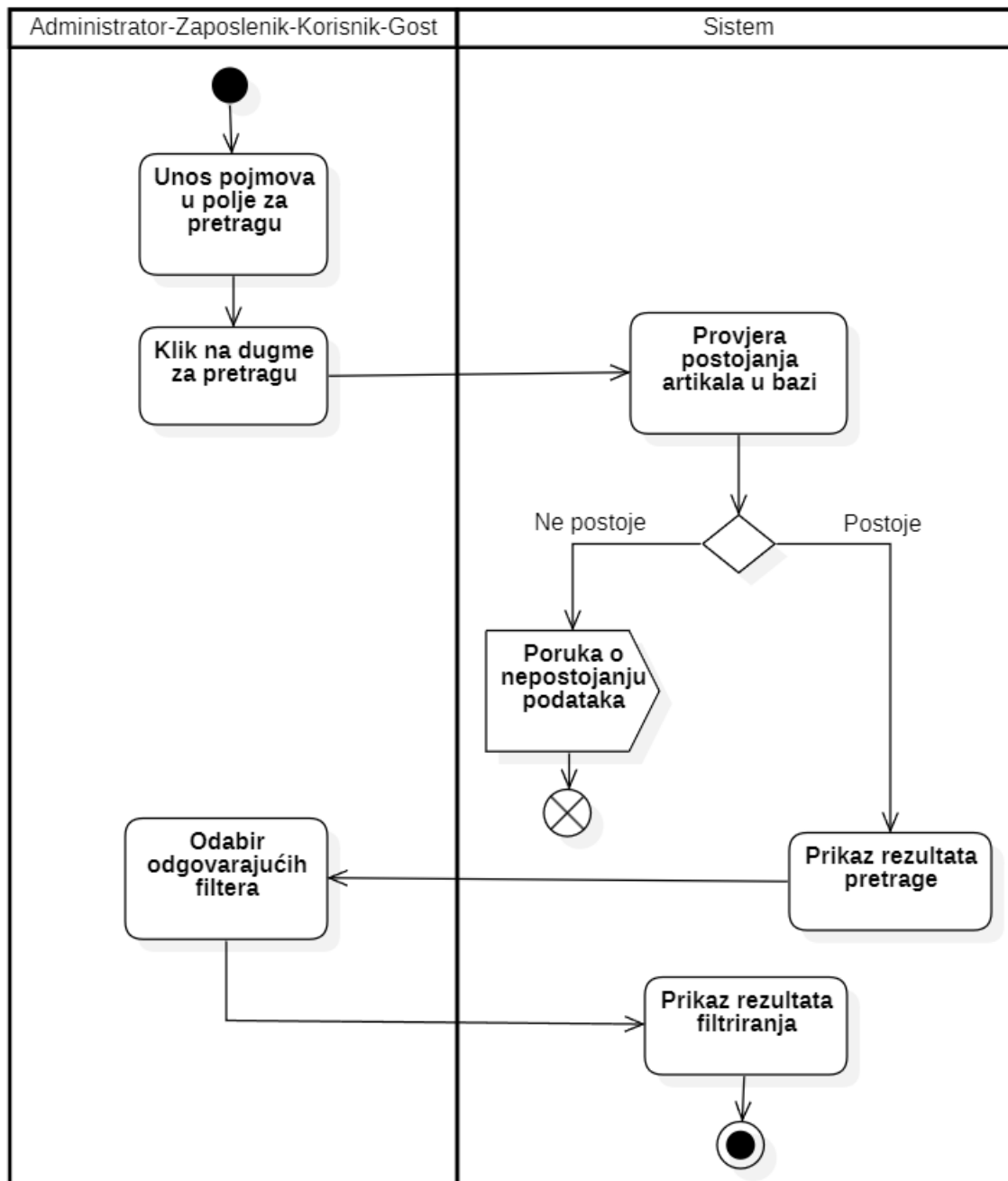
3. Dodavanje/ brisanje/uređivanje artikala



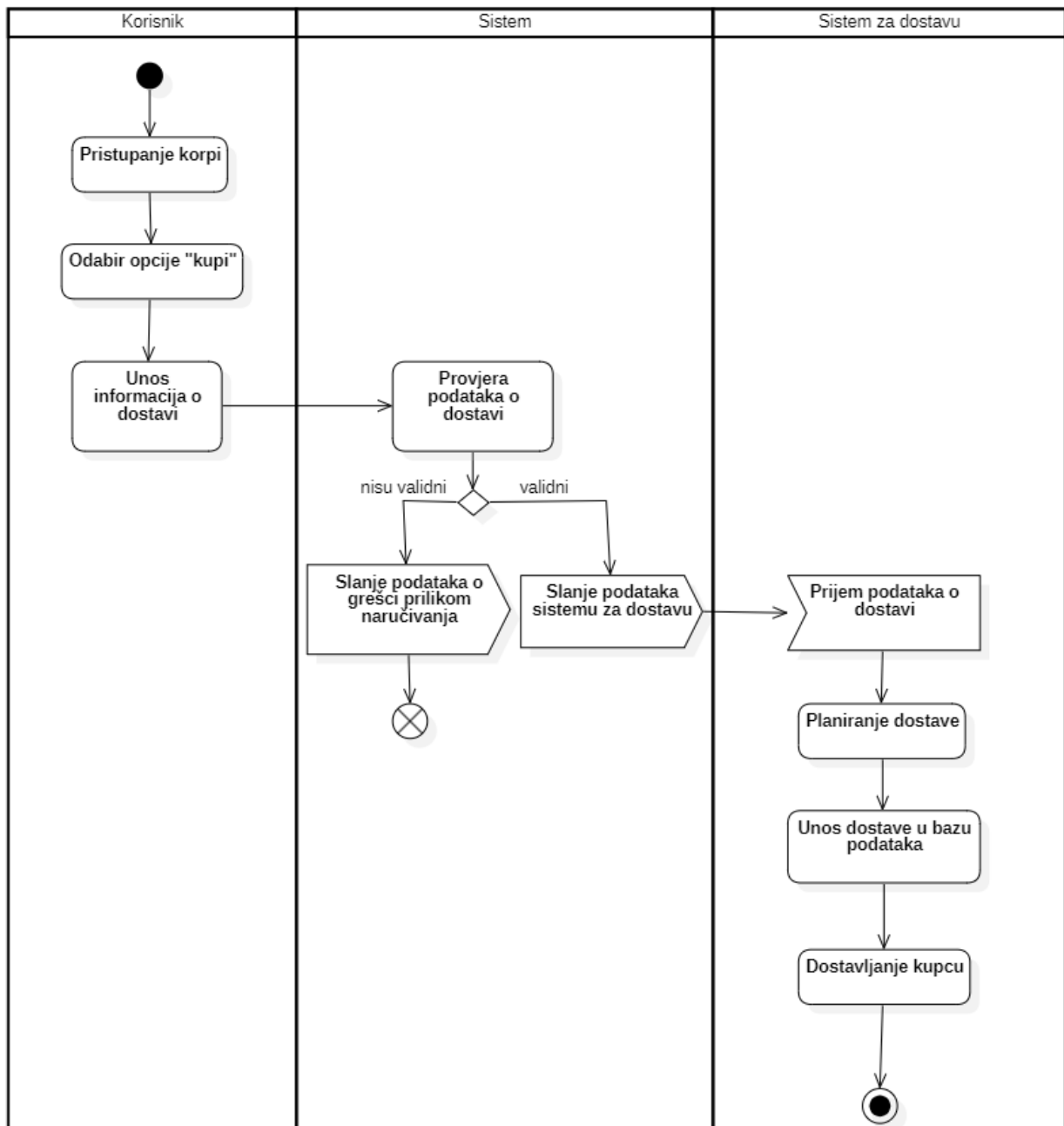
4. Dodavanje/uređivanje/brisanje narudžbi



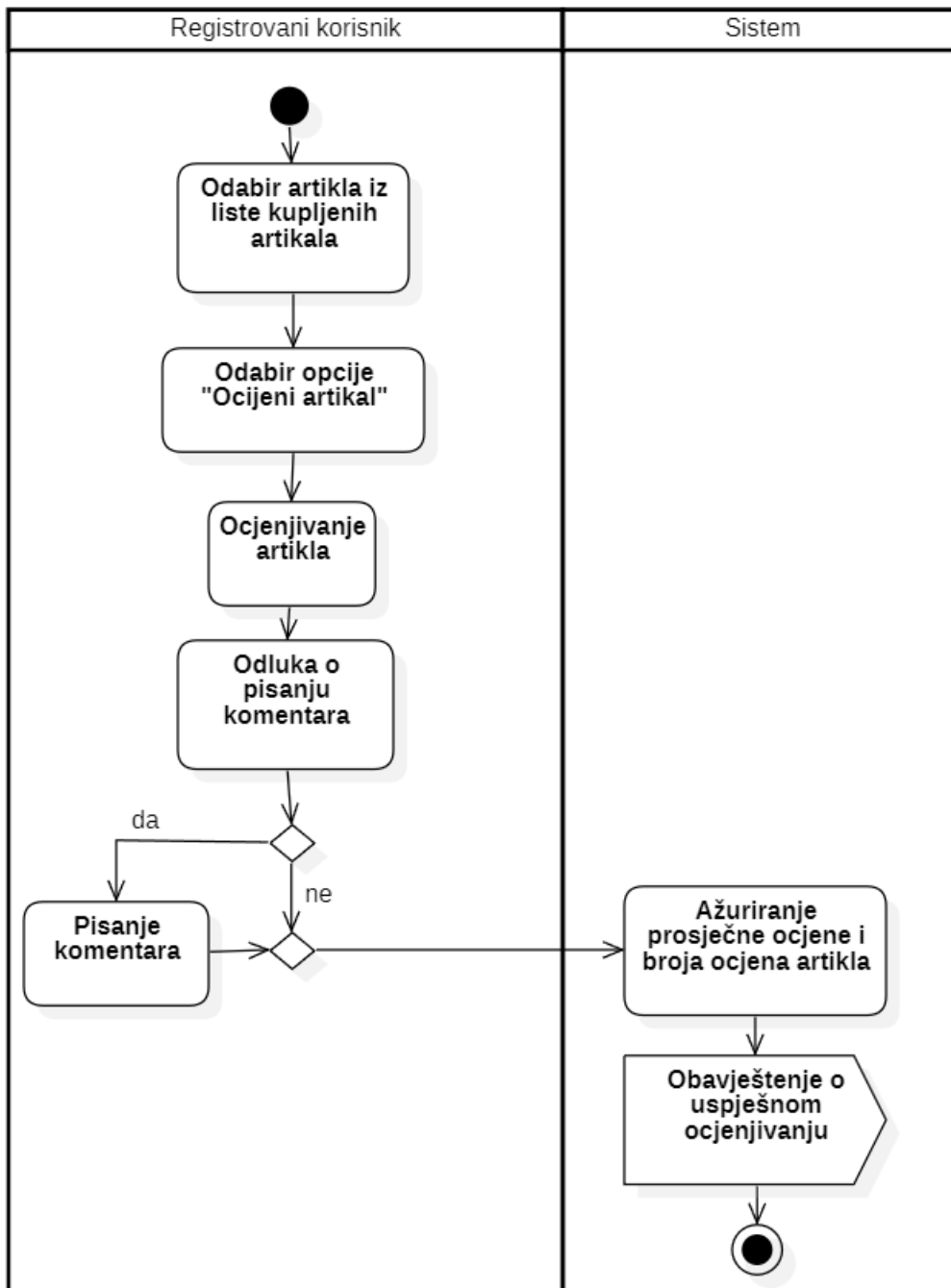
5. Pretraživanje/filtriranje artikala



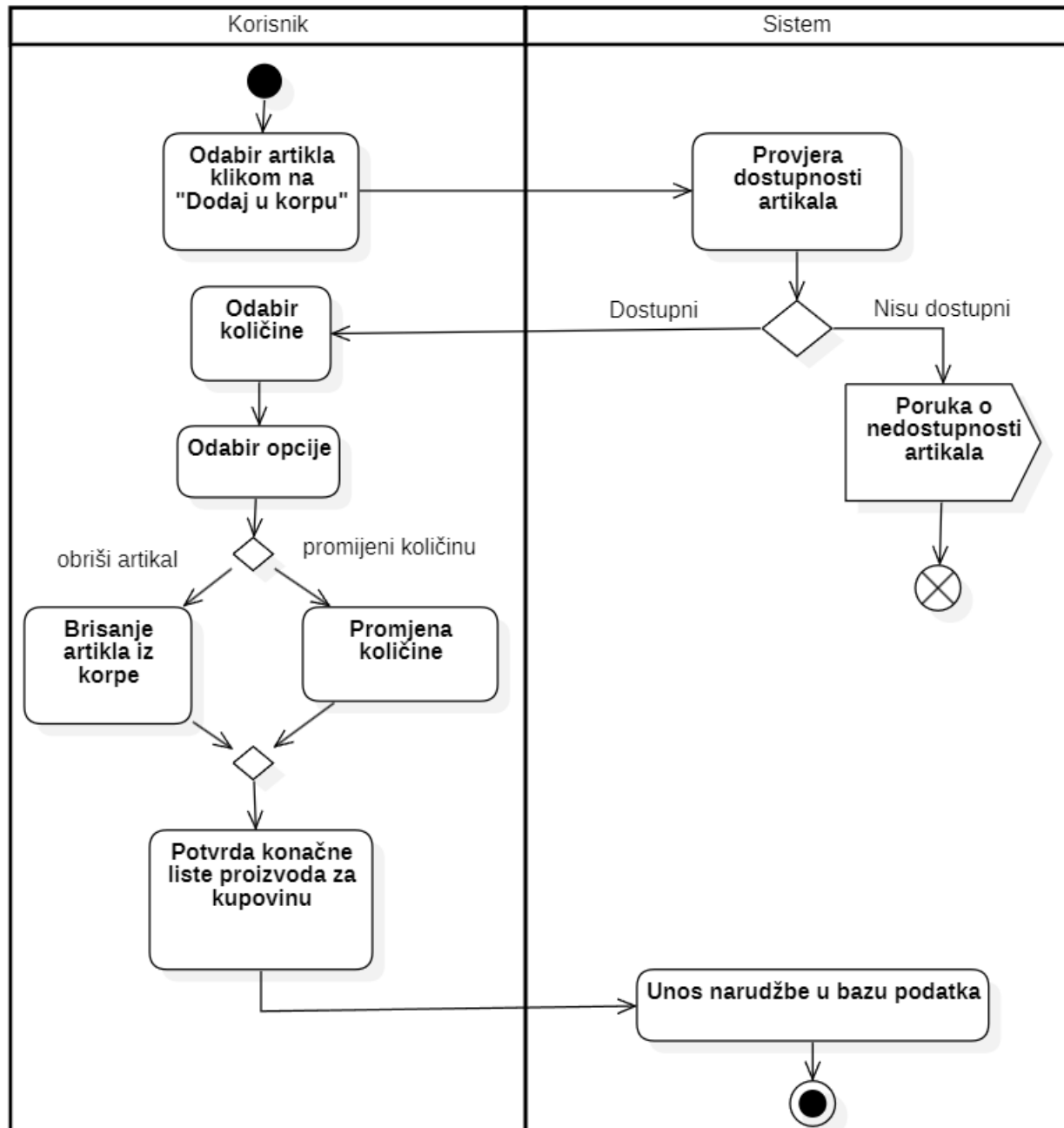
6. Naručivanje namještaja



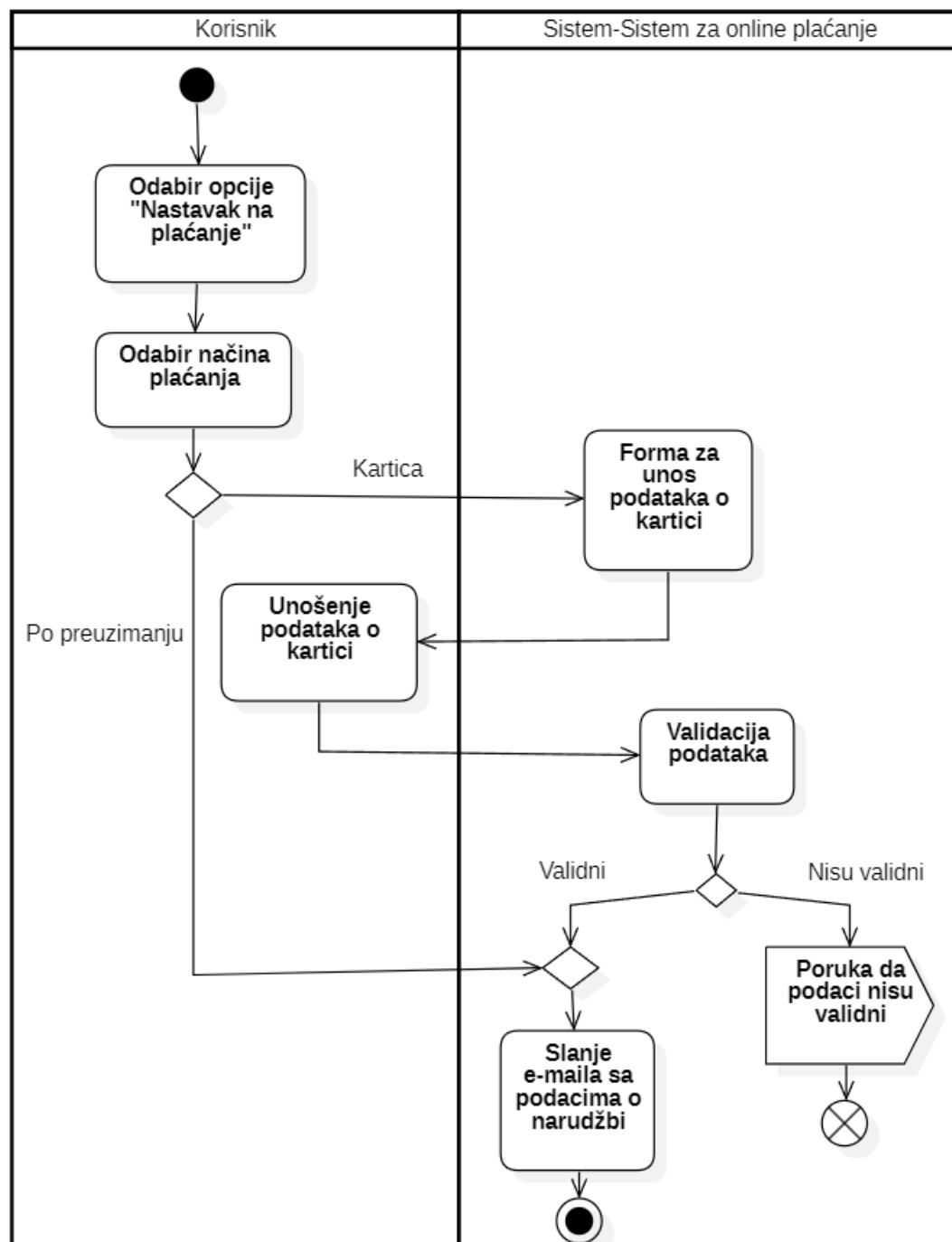
7. Ocjenjivanje/pregled recenzija artikala



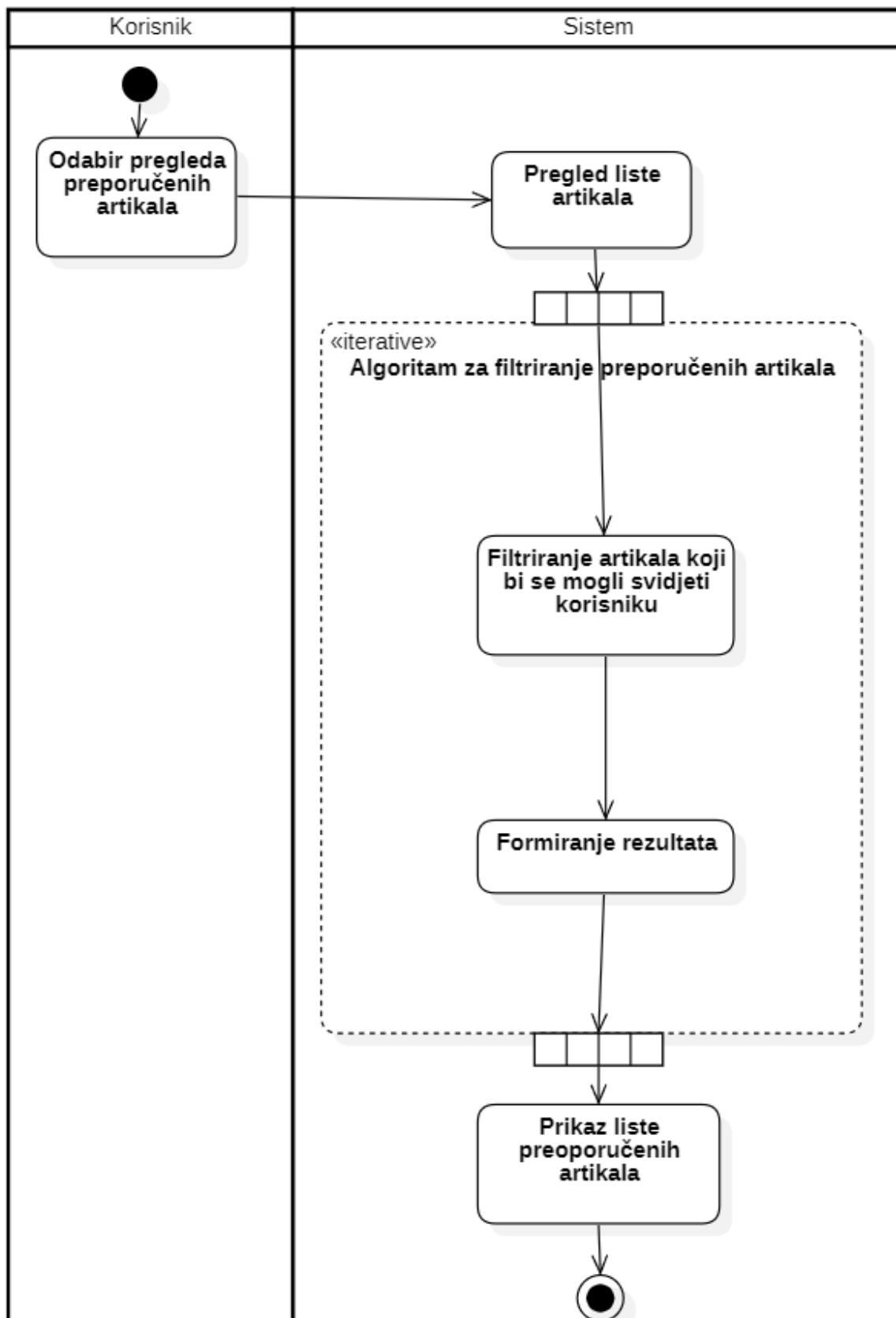
8. Upravljanje korpom



9. Plaćanje



10. Preporučivanje artikala



Analiza i dizajn sistema

Definicija klasa u sistemu

Naziv klase: Gost

Funkcionalni zahtjevi u kojima klasa učestvuje:

(FZ br. 01: Log in / Sign up)

(FZ br. 03: Pretraživanje / filtriranje artikla)

(FZ br. 05: Pregled recenzija artikala)

(FZ br. 06: Upravljanje korpom)

Atributi koje klasa posjeduje:

Naziv atributa	Tip varijable	Dodatne napomene
korpa	List<StavkaNarudzbe>	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
ipAdresa	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>

Naziv klase: Registrovani korisnik
Izveden iz klase Gost

Funkcionalni zahtjevi u kojima klasa učestvuje:

(FZ br. 04: Naručivanje namještaja)
(FZ br. 05: Ocjenjivanje artikala)
(FZ br. 07: Plaćanje)
(FZ br. 08: Preporučivanje artikala)

Atributi koje klasa posjeduje:

Naziv atributa	Tip varijable	Dodatne napomene
idKorisnika	int	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
ime	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
prezime	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
username	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
password	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
email	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
adresa	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
brojTelefona	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
podaciKartica	Kartica	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>

Naziv klase: Zaposlenik

Funkcionalni zahtjevi u kojima klasa učestvuje:

(FZ br. 01: Log in)

(FZ br. 03: Pretraživanje / filtriranje artikala)

(FZ br. 05: Pregled recenzija artikla)

Atributi koje klasa posjeduje:

Naziv atributa	Tip varijable	Dodatne napomene
idZaposlenog	int	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
ime	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
prezime	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
username	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
password	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
email	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>

Naziv klase: Administrator

Funkcionalni zahtjevi u kojima klasa učestvuje:

(FZ br. 01: Log in)

(FZ br. 02: Dodavanje/uređivanje/brisanje artikala i narudžbi)

(FZ br. 03: Pretraživanje / filtriranje artikala)

(FZ br. 08: Preporučivanje artikala)

Atributi koje klasa posjeduje:

Naziv atributa	Tip varijable	Dodatne napomene
username	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
password	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>

email	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
-------	--------	--

Naziv klase: Narudžba

Funkcionalni zahtjevi u kojima klasa učestvuje:

(FZ br. 02: Dodavanje/uređivanje/brisanje artikala i narudžbi)

(FZ br. 04: Naručivanje namještaja)

(FZ br. 06: Upravljanje korpom)

(FZ br. 07: Plaćanje)

Atributi koje klasa posjeduje:

Naziv atributa	Tip varijable	Dodatne napomene
idNarudbe	int	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
artikli	List<StavkaNarudzbe>	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
kupac	RegistrovaniKorisnik	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
datumNarudzbe	Date	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
vrijemeNarudzbe	Time	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
stanjeIsporukeNarudzbe	boolean	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
ukupanIznos	int	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>

Naziv klase: StavkaNarudžbe

Funkcionalni zahtjevi u kojima klasa učestvuje:

(FZ br. 04: Narudžba namještaja)

Atributi koje klasa posjeduje:

Naziv atributa	Tip varijable	Dodatne napomene
idStavke	int	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
idArtikla	int	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>

kolicina	int	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
cijena	int	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>

Naziv klase: Artikal

Funkcionalni zahtjevi u kojima klasa učestvuje:

(FZ br. 02: Dodavanje/uređivanje/brisanje artikala i narudžbi)

(FZ br. 03: Pretraživanje/filtriranje artikala)

(FZ br. 04: Naručivanje namještaja)

(FZ br. 05: Ocjenjivanje/pregled recenzija artikala)

(FZ br. 06: Upravljanje korpom)

(FZ br. 08: Preporučivanje artikala)

Atributi koje klasa posjeduje:

Naziv atributa	Tip varijable	Dodatne napomene
idArtikla	int	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
tip	TipNamještaja	<input type="checkbox"/> Atribut je statički <input checked="" type="checkbox"/> Atribut je <i>enumeration</i>
naziv	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
boja	Boja	<input type="checkbox"/> Atribut je statički <input checked="" type="checkbox"/> Atribut je <i>enumeration</i>
kolicina	int	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
cijena	int	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
ocjene	List<Ocjena>	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>

Naziv klase: Ocjena

Funkcionalni zahtjevi u kojima klasa učestvuje:

(FZ br. 03: Filtriranje artikala)

(FZ br. 05: Ocjenjivanje/pregled recenzija artikala)

Atributi koje klasa posjeduje:

Naziv atributa	Tip varijable	Dodatne napomene
ocjena	int	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
komentar	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
korisnik	Korisnik	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>

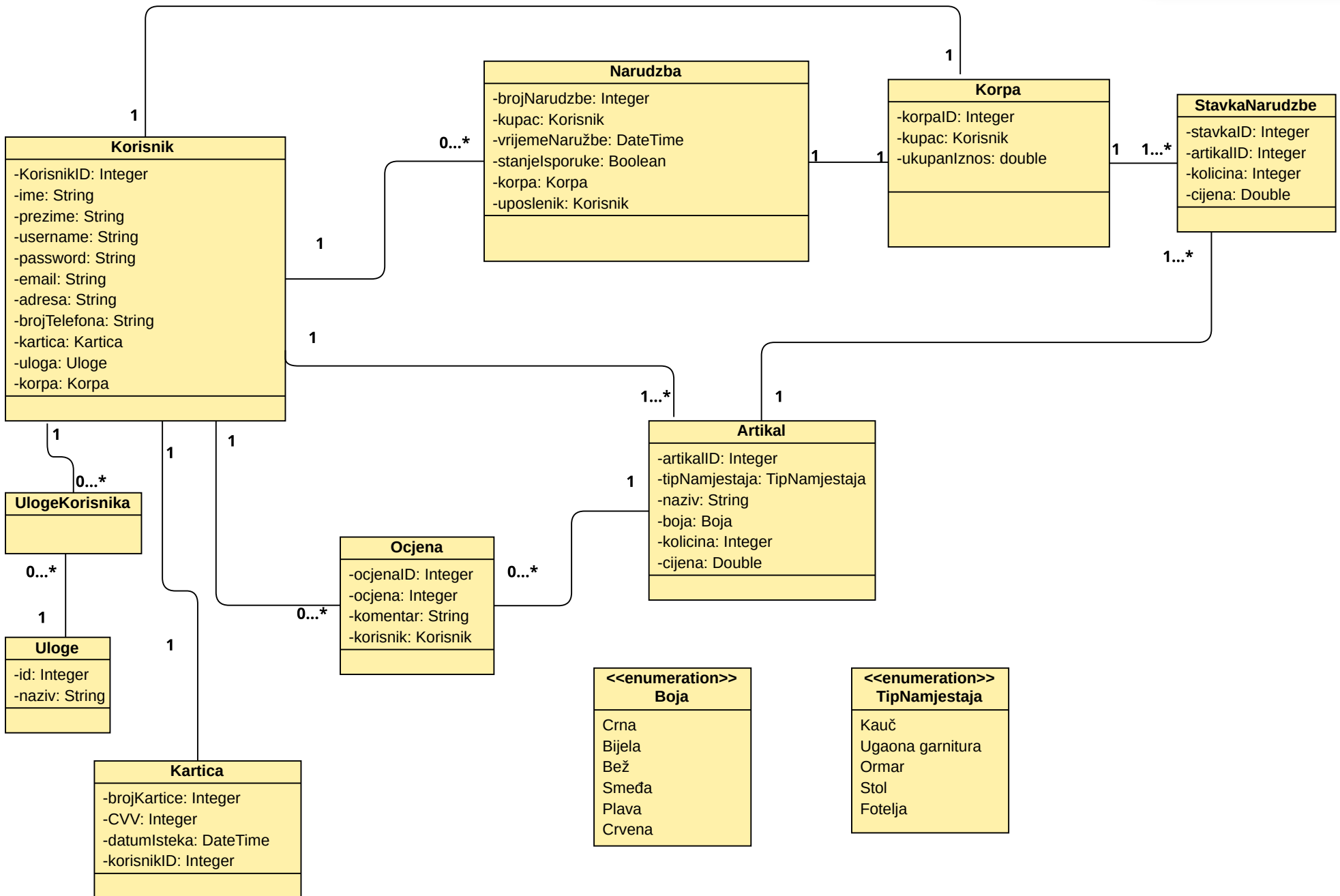
Naziv klase: Kartica

Funkcionalni zahtjevi u kojima klasa učestvuje:

(FZ br. 07: Plaćanje)

Atributi koje klasa posjeduje:

Naziv atributa	Tip varijable	Dodatne napomene
brojKartice	int	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
datumIsteka	Date	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
CVV	int	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
idKorisnika	Korisnik	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>



SOLID principi

1. S – Single Responsibility Principle (SRP)

Kada bi na našem dijagramu klase postojala klasa *Proizvod* u kojoj bismo imali metode:

- *obrisiProizvod*
- *izmijeniCijenu*
- *dajProizvod*

Primijetimo da ovu klasu možemo podijeliti u dvije klase, i to *Artikal* i *StavkaNarudzbe*, budući bi klasa *Proizvod* “znala previše”. Npr. ukoliko bi došlo do promjene u metodi *izmijeniCijenu* to bi uticalo i na već postojeće narudžbe, tj. na već postojeće artikle koje je korisnik naručio prije promjene cijene.

2. O – Open – Closed Principle (OCP)

Da bi bio ispoštovan OCP, nijedna izmjena unutar aplikacije ne smije dovesti do izmjene postojećeg koda već samo do dodavanja novog koda.

U našem slučaju imamo klasu *Artikal* koja sadrži određene attribute i akcije nad artiklima. Ukoliko bi se nakon određenog vremena pojavila potreba da unutar klase *Artikal* dodamo neke nove attribute ili metode, to bi povlačilo da se i vrši izmjena i već postojećih artikala u narudžbi ili korpi (kada bismo unutar njih koristili instance klase *Artikal*).

Ovaj problem rješavamo tako što pravimo dvije različite klase, *Artikal* i *StavkaNarudzbe*, koji će imati različite attribute i metode.

3. L – Liskov Substitution Principle (LSP)

LSP kaže da ukoliko imamo klasu *S* koja je naslijeđena iz klase *T*, onda uvijek mora biti moguće instance klase *T* zamijeniti instancama klase *S*.

U našem slučaju Liskov princip je ispoštovan, budući da na našem dijagramu klasa nemamo naslijeđivanje.

S druge strane, da smo recimo imali klasu *Uposlenik* iz koje bi bile izvedene klase *Zaposlenik* i *Administrator*, ovaj princip ne bi bio ispoštovan. Klasa *Uposlenik* bi imala neke metode koje omogućavaju upravljanje artiklima i narudžbama. Kada bismo pokušali da instancu tako definisane klase *Uposlenik* zamijenimo instancom klase *Zaposlenik* (koja radi samo određene akcije sa artiklima) direktno bismo prekršili LSP.

4. I – Interface – Segregation Principle (ISP)

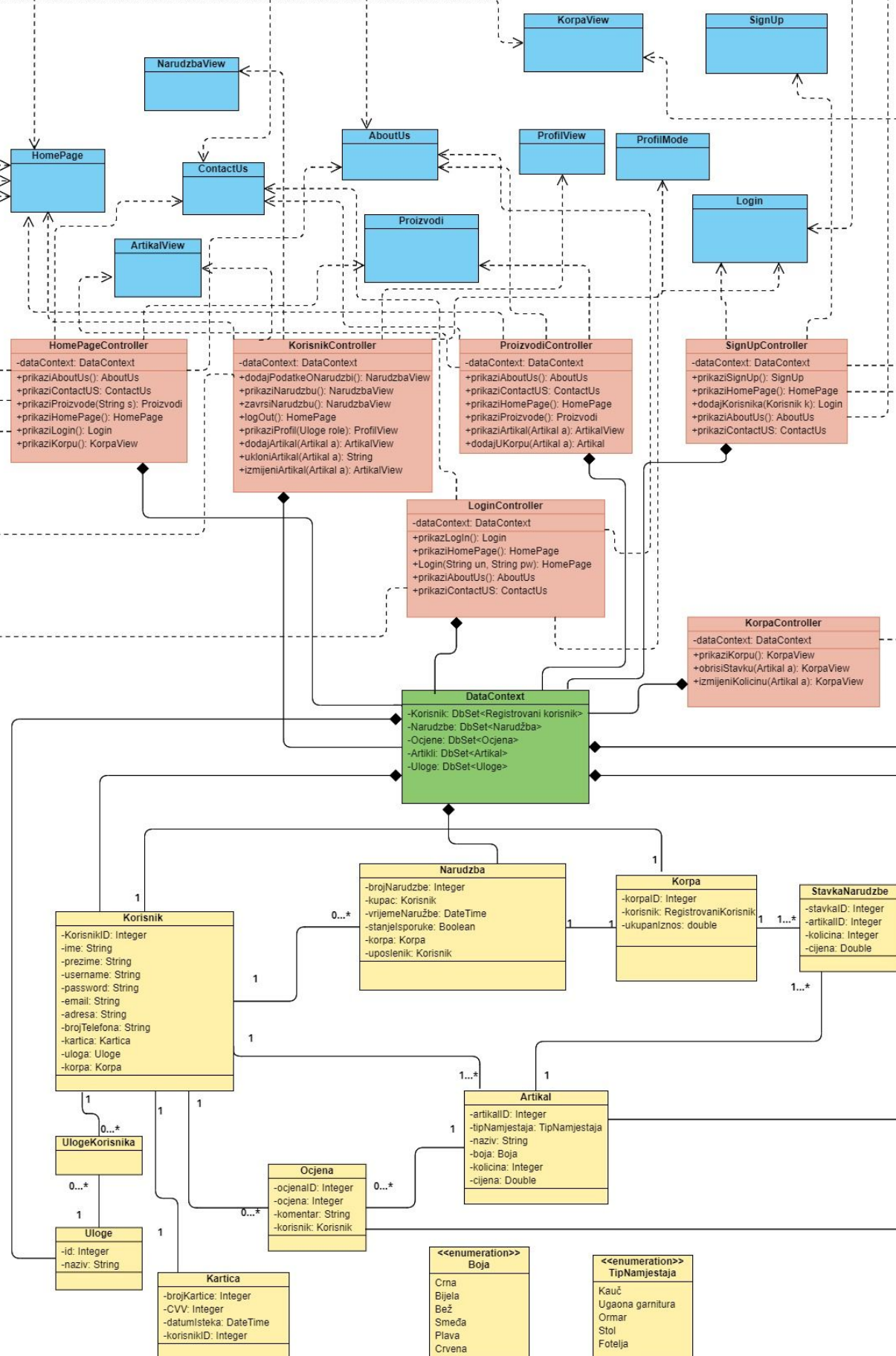
U slučaju da klasa *Korisnik* ne bude interface, postojala bi mogućnost da dodavanjem različitih vrsta korisnika klasa postane preopterećena. Rješenje za tu situaciju je dodavanje interface-a *KorisnikInterface* i da iz njega naslijedimo pojedine tipove korisnika. U tom slučaju je ispoštovan ISP.

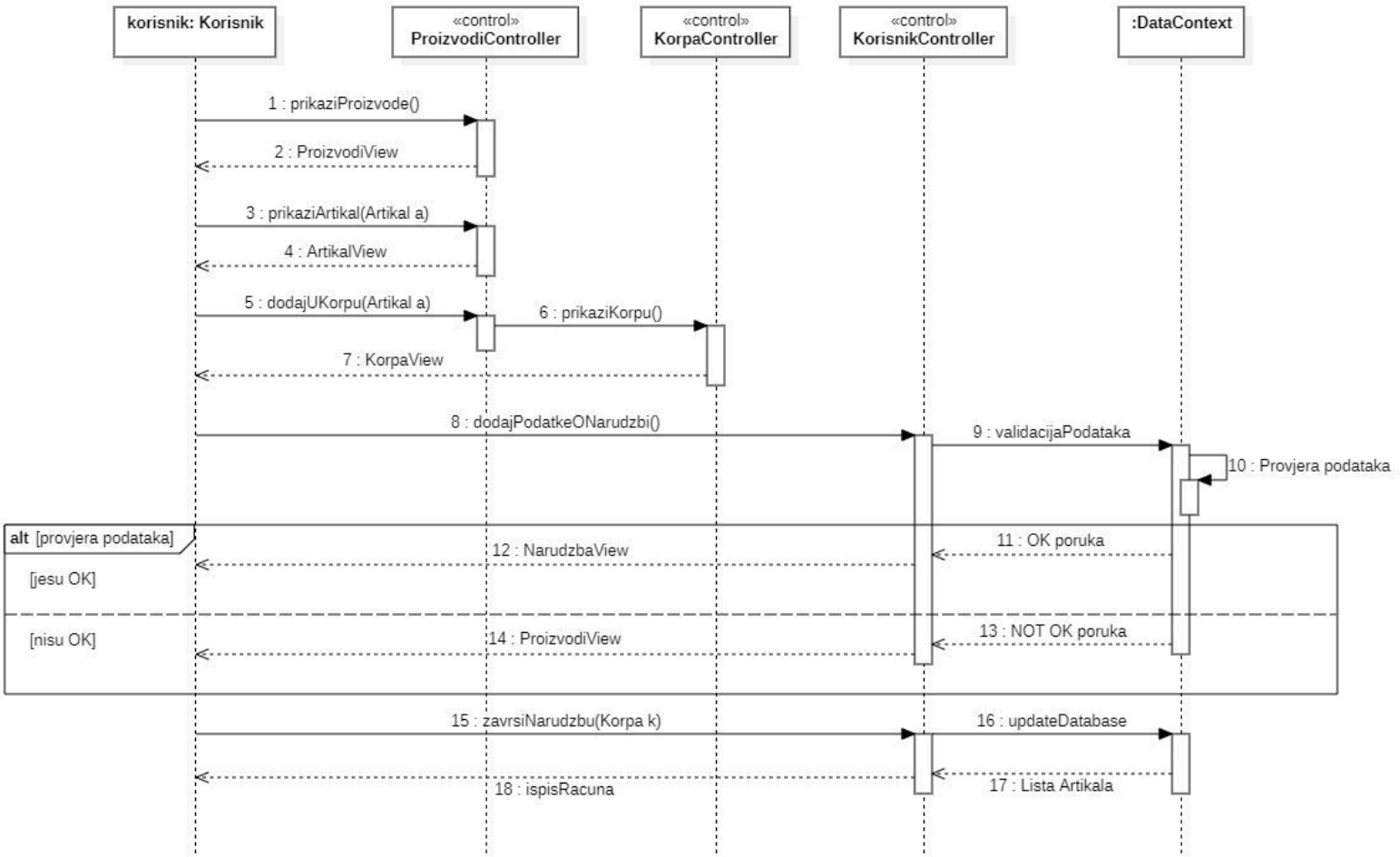
5. D – Dependency – Inversion Principle (DIP)

DIP kaže da moduli visokog nivoa ne bi trebali da zavise od modula niskog nivoa.

Konkretno u našem sistemu možemo primijetiti da klasa *Narudzba* direktno zavisi od *Korisnik* klase, budući da sam korisnik ima mogućnost da obriše i uredi narudžbu, zbog čega se implicira da će *Narudzba* biti afektirana akcijama instance klase *Korisnik*.

Na primjer, umjesto da klasa *Narudzba* direktno koristi klasu *Korisnik*, možemo dizajnirati interfejs ili apstrakciju koji će definisati funkcionalnost koja je potrebna klasi *Narudzba*, a zatim implementirati tu funkcionalnost u klasi *Korisnik*. Na taj način, *Narudzba* neće direktno zavisiti od implementacije *Korisnik* klase, već će zavisiti samo od interfejsa ili apstrakcije.





EFMigrationsHistory	
MigrationId	
ProductVersion	

AspNetRoles	
Id	
Name	
NormalizedName	
ConcurrencyStamp	

AspNetUserRoles	
Id	
UserId	
RoleId	

AspNetUserTokens	
Id	
UserId	
LoginProvider	
Name	
Value	

AspNetUsers	
Id	
UserName	
NormalizedUserName	
Email	
NormalizedEmail	
EmailConfirmed	
PasswordHash	
SecurityStamp	
ConcurrencyStamp	
PhoneNumber	
PhoneNumberConfirmed	
TwoFactorEnabled	
LockoutEnd	
LockoutEnabled	
AccessFailedCount	

AspNetUserClaims	
Id	
UserId	
ClaimType	
ClaimValue	

AspNetRoleClaims	
Id	
RoleId	
ClaimType	
ClaimValue	

Narudzba	
Id	
Idkorisnik	
korisnikId	
vrijemeNarudzbe	
stanjeIsporuke	
Idkorpa	
korpaId	

AspNetUserLogins	
Id	
LoginProvider	
ProviderKey	
ProviderDisplayName	
UserId	

AspNetUserLogins	
Id	
LoginProvider	
ProviderKey	
ProviderDisplayName	
UserId	

Artikal	
Id	
naziv	
tip	
boja	
kolicina	
cijena	
Iddimenzije	
dimenzijId	

Korpa	
Id	
ukupanIznos	
Idkorisnik	
korisnikId	

Korisnik	
Id	
ime	
prezime	
adresa	
brojTelefona	

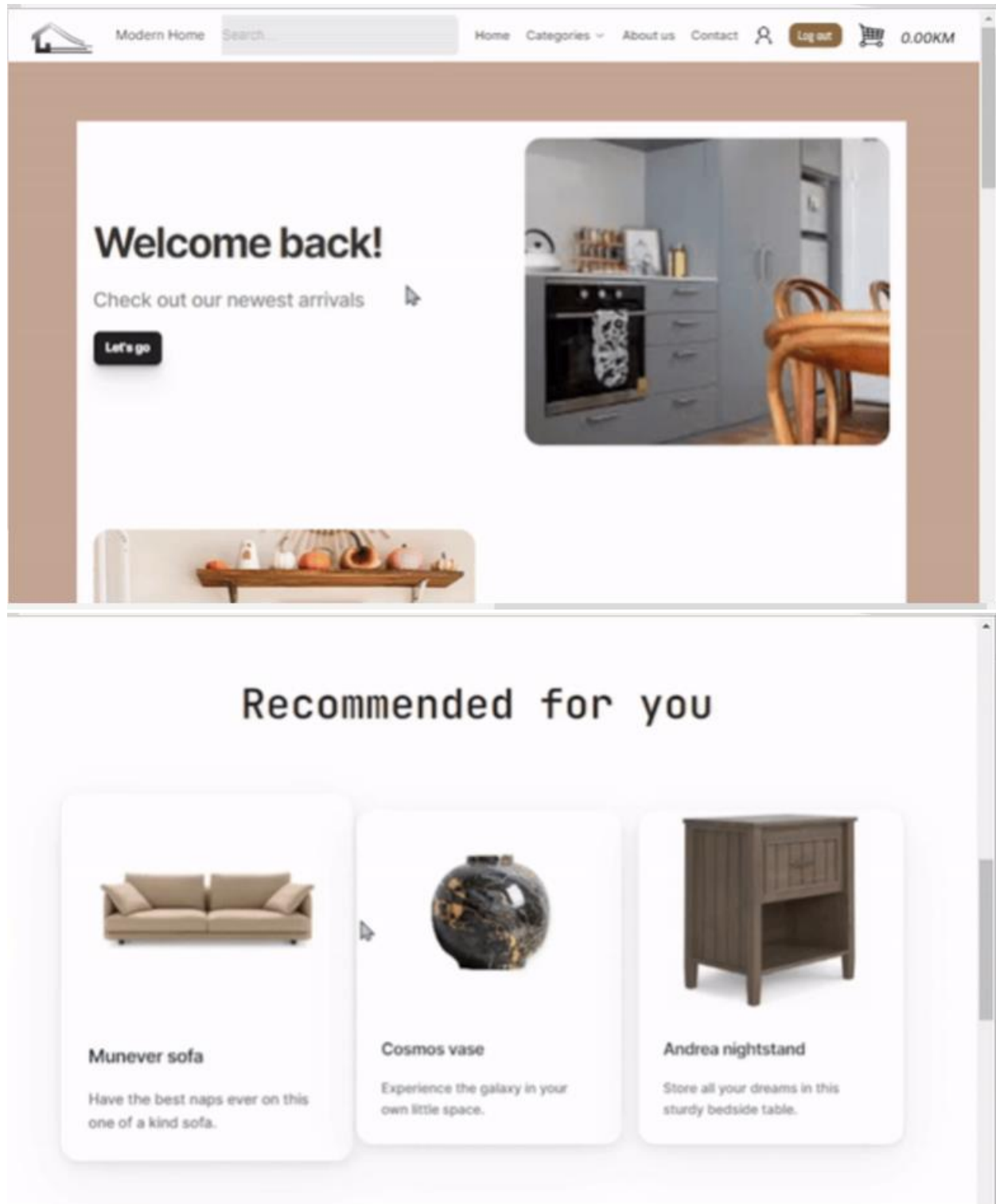
Kartica	
Id	
brojKartice	
datumIsteka	
Idkorisnik	
korisnikId	

Ocjena	
Id	
ocjena	
komentar	
Idkorisnik	
korisnikId	

StavkaNarudzbe	
Id	
Idartikal	
artikalId	
kolicina	
cijena	
Idkorpa	
korpaId	

Dimenzije	
Id	
visina	
sirina	
duzina	

Prototipi



Rate your previously bought products



Contact us

Phone: +(253) 644-2182

Socials



MODERN HOME

Thank you for
visiting our
website.

Contact us

Phone: +(253) 644-2182

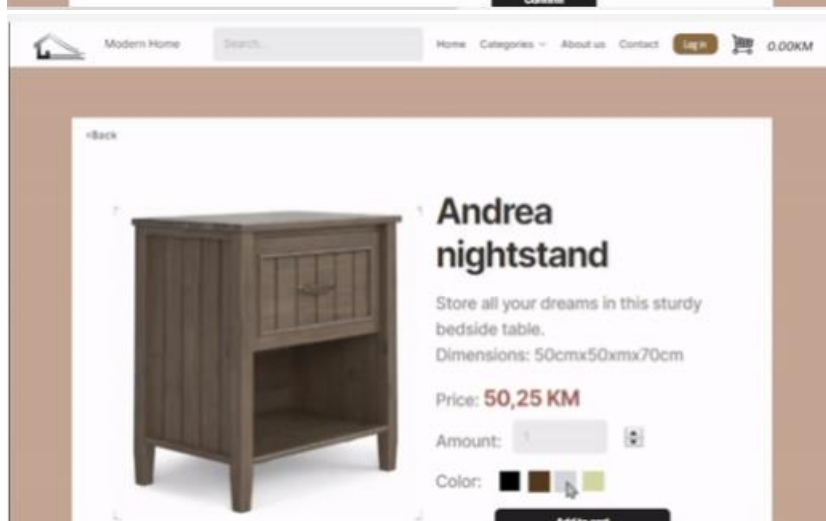
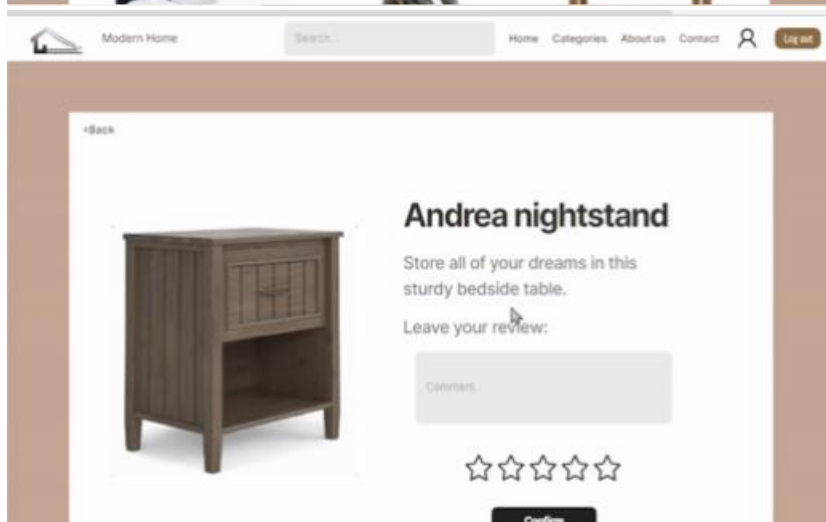
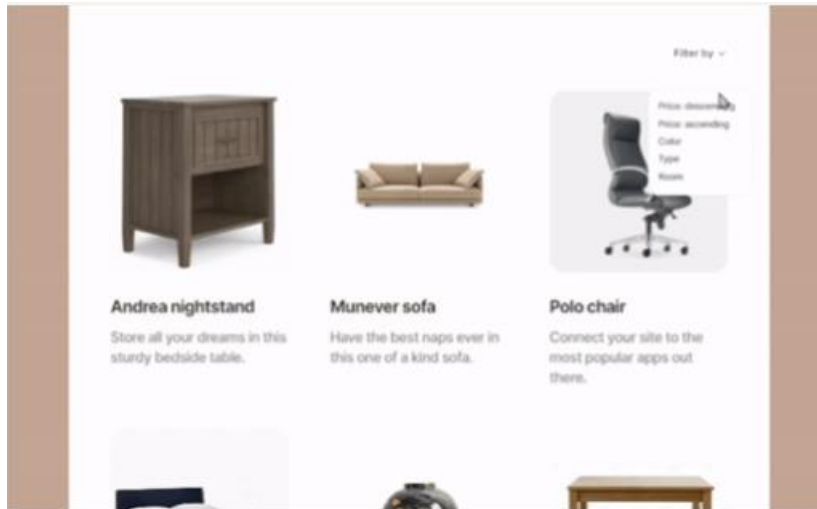
E-mail: info@modernhome.ba

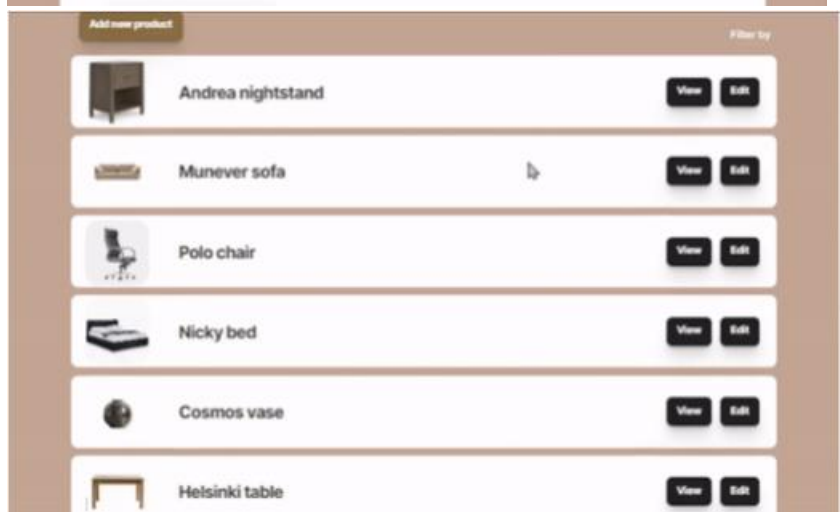
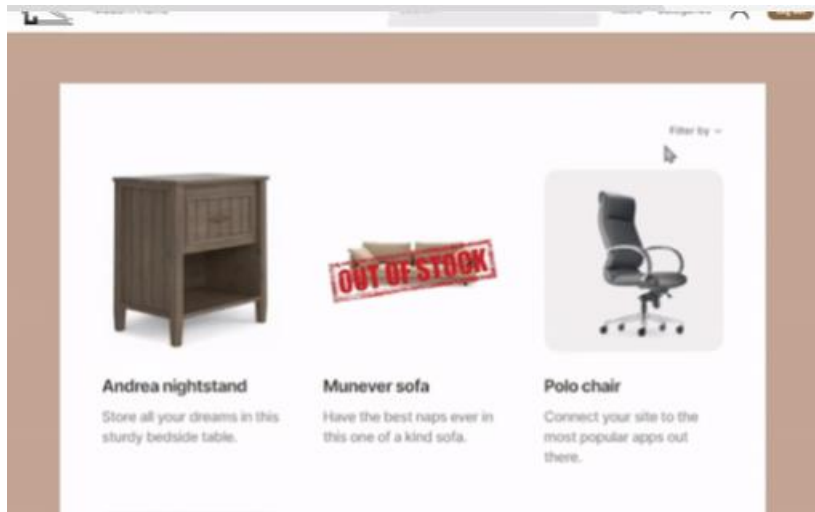
Address: 3381 Stadium Drive,
Dedham, MA


Socials

Keep up with all of the
newest updates and be
the first to find out
about upcoming arrivals.








 Modern Home

Search...

[Home](#) [Products](#) [Orders](#)  [Log out](#)

[Back](#)

add a picture

Name

Description


Dimension

Price


Color

Amount

Confirm

 Modern Home

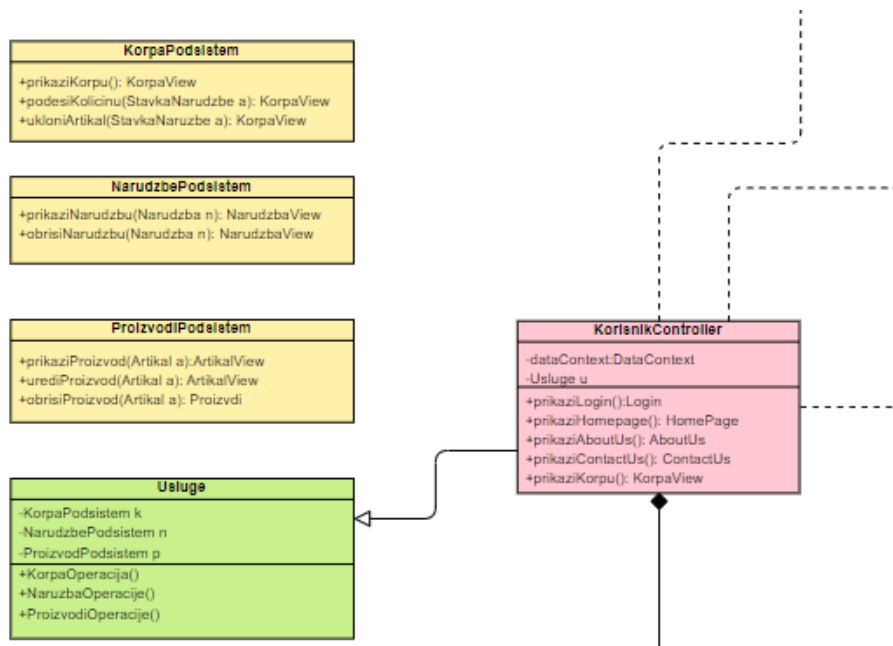
Search...

[Home](#) [Products](#) [Orders](#)  [Log out](#)

	Time of order	Expected delivery	Filter by	
andyshops	April 22nd, 2024. 8:37 PM	April 29th, 2024.	<div><div>View</div><div>Edit</div></div>	
mirthazigler	April 22nd, 2024. 7:55 PM	April 29th, 2024.	<div><div>View</div><div>Edit</div></div>	
sepiadecor	April 21st, 2024. 11:02 PM	April 28th, 2024.	<div><div>View</div><div>Edit</div></div>	
mauveious	April 21st, 2024. 3:25 PM	April 28th, 2024.	<div><div>View</div><div>Edit</div></div>	
tripstudio	April 21st, 2024. 11:49 AM	April 28th, 2024.	<div><div>View</div><div>Edit</div></div>	

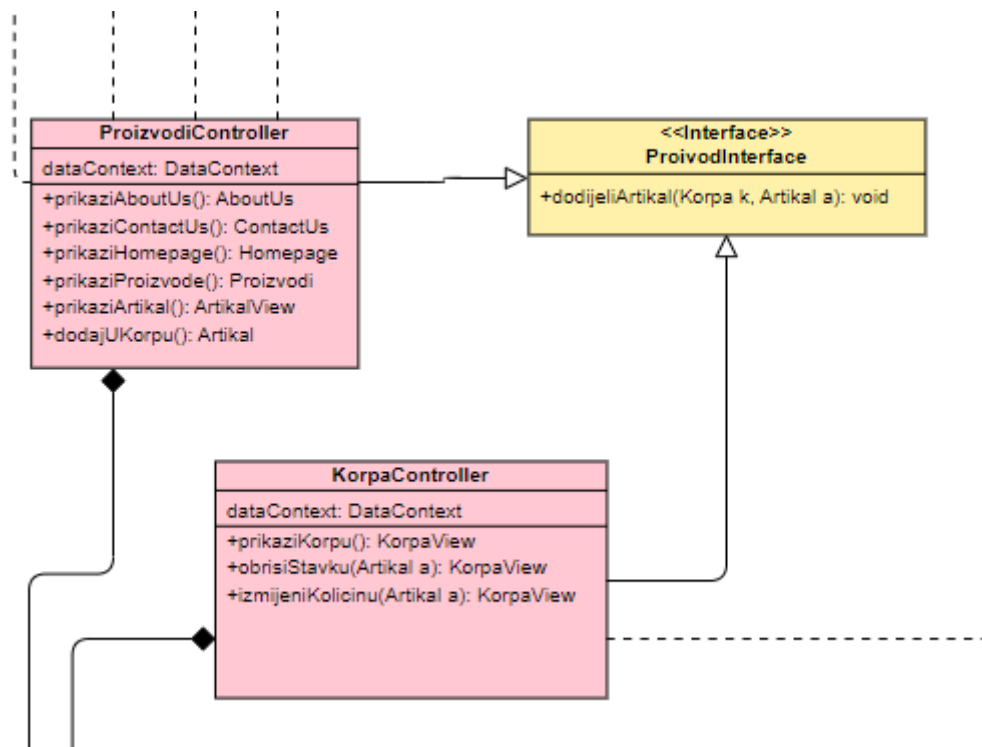
Strukturalni paterni

1. **Adapter pattern** – ovaj tip paterna kreira novu adapter klasu koja služi kao posrednik između originalne klase i željenog interfejsa. Ovime omogućavamo implementaciju neke funkcionalnosti bez ugrožavanja integriteta cijele aplikacije. Adapter pattern bismo mogli primijeniti za obradu kartičnog plaćanja, pošto će se za obradu koristiti neki vanjski servis. Možemo implementirati adapter koji će preuzeti odgovornost za obradu detalja transakcije, tj. kada se unesu podaci vezani za kartično plaćanje, adapter ih prilagođava formatu koji taj vanjski servis zahtijeva.
2. **Facade pattern** – ovaj patern se koristi kada je potrebno da se osigura više pogleda visokog nivoa na podsisteme čija je implementacija sakrivena od korisnika. U našem slučaju, ovaj patern bi nam bio mnogo koristan, budući da su backend sistemi poprilično kompleksni, te je samu implementaciju potrebno sakriti od korisnika. Mogli bismo imati podsistem za upravljanje korpom, narudžbama, te za sam pregled proizvoda, i na osnovu tih podsistema formirati facade interfejs UslugaInterfejs. Ova ideja je prikazana na slici ispod.



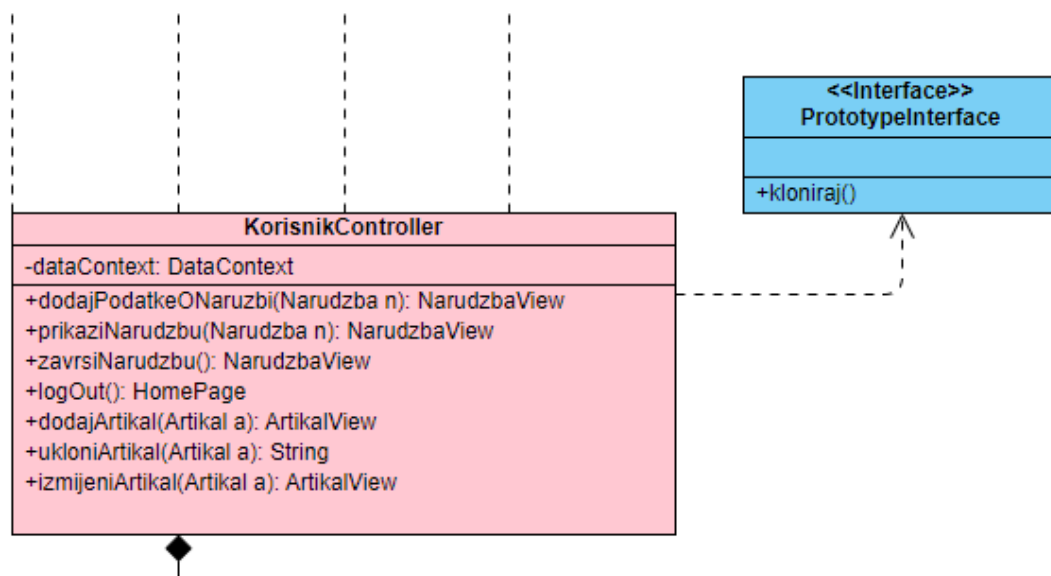
3. **Decorator pattern** – koristi se kada je potrebno da se omogući dinamička dodjela novih elemenata i funkcionalnosti postojećim objektima. Ovaj paterni bismo mogli primijeniti u svrhu proširenja funkcionalnosti prilikom narudžbe artikla, kao npr. mogućnost unosa nekog koda za popust na naručene artikle, ili recimo mogućnost ostavljanja detaljnih naznaka vezanih za način isporuke.
4. **Bridge pattern** – osnovna namjena ovog patern jest da omogući odvajanje apstrakcije i implementacije neke klase tako da ta klasa može posjedovati više različitih apstrakcija i više implementacija za pojedine apstrakcije. Ovaj patern bi nam poslužio kada bismo klasu Artikal implementirali kao apstraktnu klasu, a svaki pojedini tip namještaja (stolice, fotelje, stolovi, kreveti i sl.) naslijedili iz ove klase.
5. **Composite pattern** – omogućava formiranje strukture drveta pomoću klase, u kojoj se individualni objekti (listovi stabla) i kompozicije individualnih objekata (korijeni stabla) jednako tretiraju. U našem slučaju, ovaj patern bismo mogli primijeniti recimo na klase Narudzba, StavkaNarudzbe i Artikal, i to na način da Narudzba predstavlja složeni objekt koji sadrži StavkaNarudzbe, dok StavkaNarudzbe u sebi sadrži informacije o artiklu.
6. **Proxy pattern** – virtualni proxy se koristi kada želimo odgoditi stvaranje stvarnog objekta kada on nije potreban; remote proxy rješava problem kada se objekt ne može instancirati direktno, dok zaštitni proxy omogućava pristup i kontrolu pristupa nad stvarnim objektima. Konkretno za naš slučaj, proxy objekat bismo mogli koristiti za autentifikaciju i autorizaciju administratora za obavljanje određenih zadataka, recimo uređivanje artikala, pregled i brisanje narudžbi i sl. Proxy objekat će različitim korisnicima omogućiti različit nivo pristupa, ne narušavajući integritet same aplikacije. Proxy objekat će biti posrednik između korisnika i stvarne implementacije objekta, osiguravajući da se samo korisnici koji imaju odgovarajuće privilegije mogu vršiti određene akcije.

7. ***Flyweight patern*** – osigurava brzinu dohvaćanja svakog objekta na zahtjev, koristeći činjenicu da postoje situacije u kojima su pojedini dijelovi klase isti za sve instance te klase. Odabrali smo ovaj patern budući da nam optimizira memoriju i brzinu odgovora na pojedine zahtjeve, budući da ćemo imati poprilično mnogo različitih artikala. Smanjujemo zauzeće memorije tako što dijelimo zajedničke podatke. Uzmimo recimo krevete koji su dio našeg asortimana, pa ćemo u bazi imati samo jednu instancu artikla tipa krevet koji ćemo dalje koristiti za kreiranje narudžbi, i na taj način ostvariti značajnu uštedu memorije.

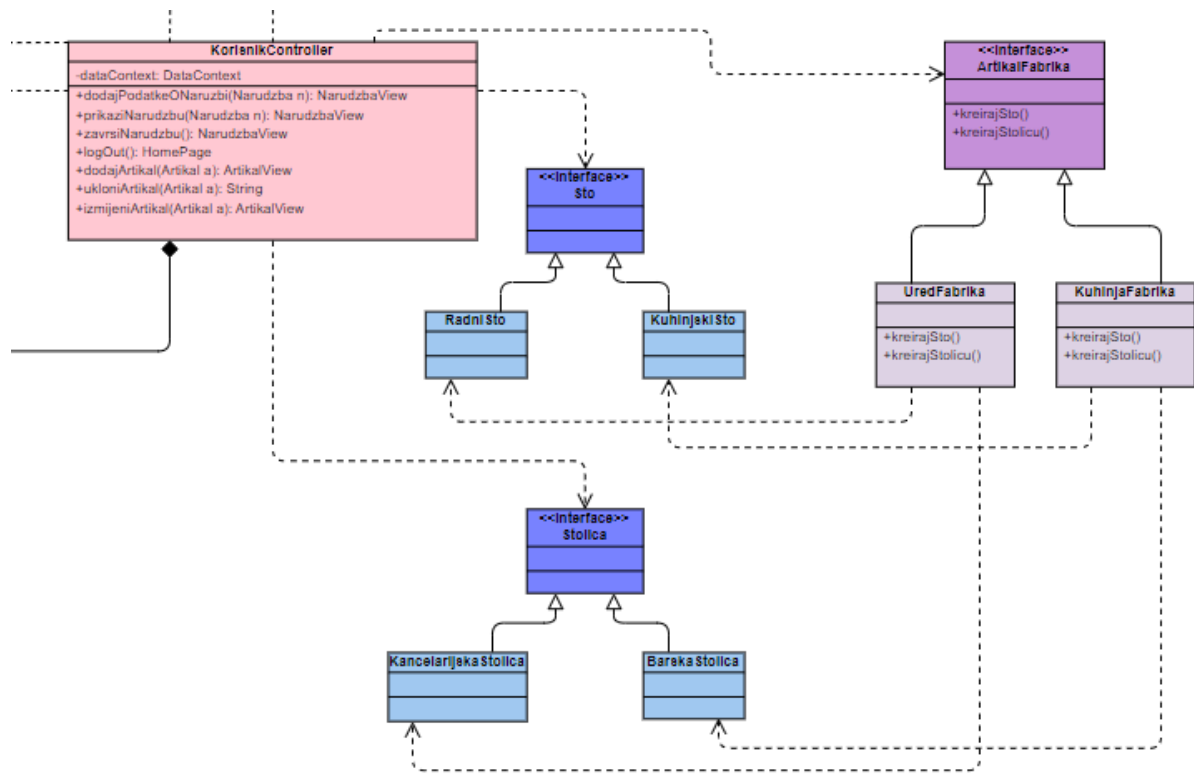


Kreacijski paterni

1. **Singleton patern** – osigurava da se klasa može instancirati samo jednom, te globalni pristup kreiranoj instanci klase. Recimo, ako bismo imali samo jednog administratora sistema, ovaj patern bi nam osigurao da može postojati samo jedna instanca ove klase unutar aplikacije. Sve funkcionalnosti koje administrator ima bile bi ograničene na tu jednu instancu klase, samo bismo unutar klase Administrator morali odvojeno implementirati sve potrebne metode i atribute.
2. **Prototype patern** – dozvoljava da se kreiraju prilagođeni objekti bez poznavanja njihove klase ili detalja kako je objekat kreiran, koristi se ako je kreiranje novog objekta resursno skupo ili kompleksno. Ovaj patern umjesto da kreira objekat iz početka, klonira postojeće objekte i modifikuje ih na zahtjev. Definišemo PrototypeInterface, koji će imati metodu kloniraj(), dok će nam klasa Artikal služiti kao konkretan prototip. U sklopu KorisnikController imamo operaciju za modifikaciju artikla, koja se odvija na način da se klonira originalni prototip, zatim se izvrše odgovarajuće izmjene na kloniranom objektu, nakon čega se vrši spašavanje.

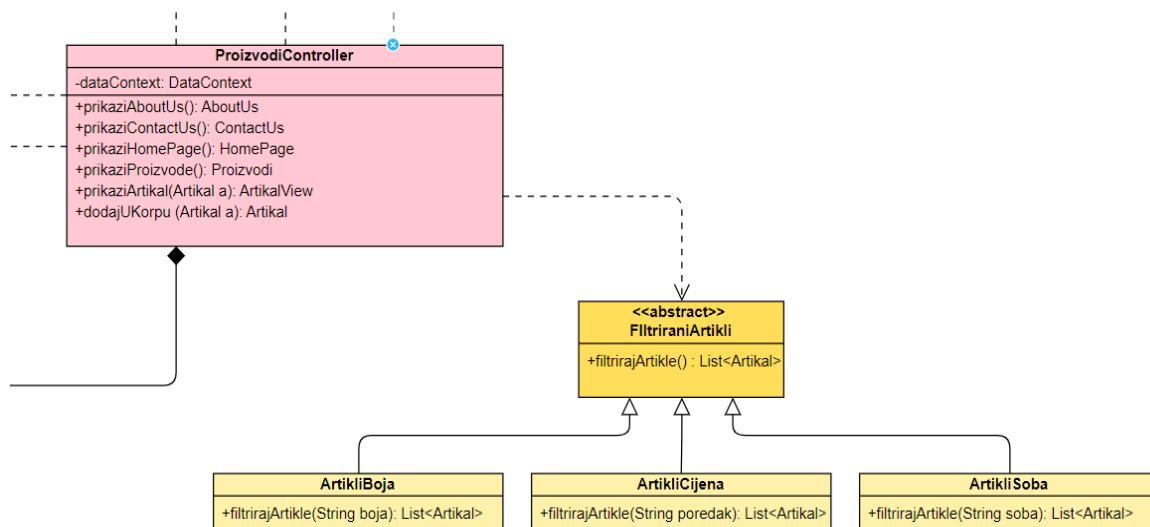


3. ***Factory method patern*** – omogućava kreiranje objekata na način da podklase odlučuju koju će klasu instancirati. Kada bismo imali klasu Zaposlenik, ovaj patern bismo mogli iskoristiti kada bismo razlikovali različite tipove zaposlenika, kao npr. menadžer, skladištar i sl. Ovisno o situaciji, tj. ovisno o tipu zaposlenika, mogli bismo imati ZaposlenikFactory, koja bi bila zadužena za kreiranje objekata tipa Zaposlenik, sa različitim kriterijima. Ovako osiguravamo umjesto da kontroleri ili klase direktno instanciraju objekte klase Zaposlenik, koristili bismo ZaposlenikFactory za dobijanje različitih tipova zaposlenika.
4. ***Builder patern*** – odvaja specifikaciju kompleksnih objekata od njihove stvarne konstrukcije, koristi se ako se objekti mogu podijeliti u skupove objekata koji se razlikuje samo po permutaciji njihovih dijelova. U našem slučaju, ovaj patern bismo mogli iskoristiti za kreiranje objekata klase Artikal, budući da je poprilično efikasan kada su u pitanju kompleksni objekti sa mnogo atributa i metoda, što je kod nas klasa Artikal. Također je pogodan kada u trenutku kreiranja objekta, svi atributi nisu poznati. Builder interface povezujemo sa svim kontrolerima koje i sama klasa Artikal koristi.
5. ***Abstract factory patern*** – odvaja definiciju klase proizvoda od klijenta, u smislu da se na osnovu apstrakne familije proizvoda kreiraju konkretne fabrike produkata različitih tipova i različitih kombinacija. Idealna primjena ovog paternu bi bila za kreiranje različitih tipova namještaja, odnosno artikala, koje su dostupne salonu. Kreiramo apstraktnu fabriku ArtikalFabrika unutar koje definišemo metode za kreiranje različitih tipova artikala. To bi recimo bile metode kreirajSto, kreirajStolicu, kreirajKrevet i sl. Svaka konkretna fabrika (UredFabrika, KuhinjaFabrika...) bi implementirala ove metode i vraćala odgovarajuće objekte. Ova ideja je prikazana na sljedećoj slici.



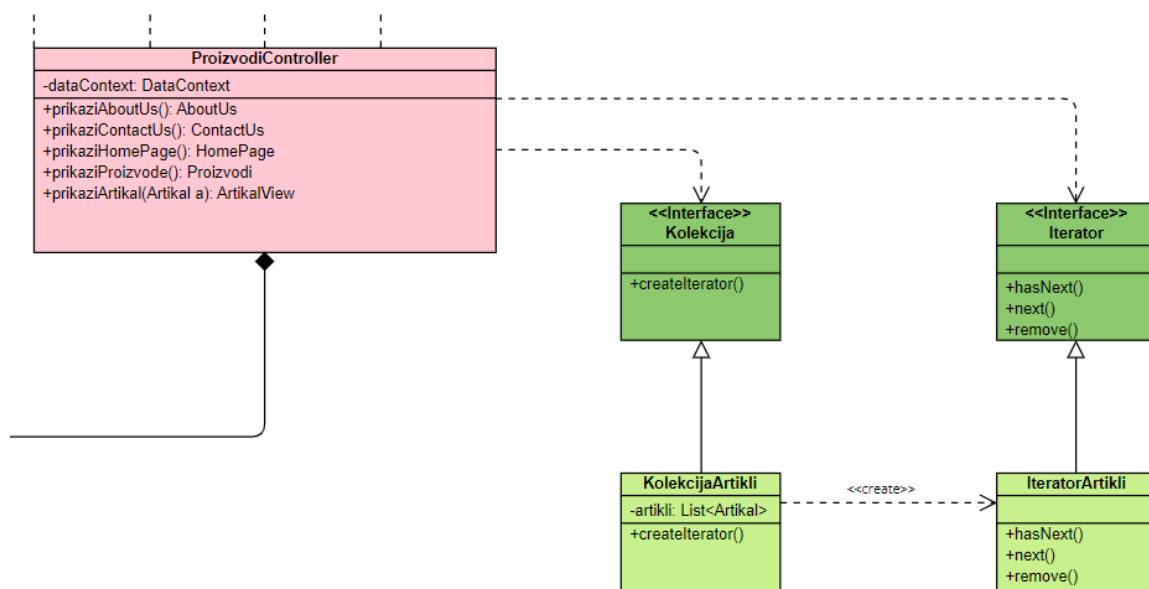
Paterni ponašanja

1. **Strategy patern** – koristi se kada postoje različiti primjenjivi algoritmi, odnosno strategije za neki problem. Ovaj patern bismo mogli iskoristiti kod odabira načina plaćanja narudžbe. Korisnik može odabrati kartično plaćanje ili plaćanje pouzecom. Kreiramo *PlacanjeStrategy* interfejs i definiramo metode *izvrsiPlacanje()* i *izvrsiPovrat()* za različite načine plaćanja. Imali bismo konkretne implementacije ovog interfejsa, i to *KarticaStrategy* i *GotovinaStrategy*, te bismo unutar klase *Narudzba* definisali metodu *postaviMetoduPlacanja()*, koja će postaviti željenu metodu plaćanja. Također bismo unutar klase *Narudzba* definisali i metodu *izvrsiPlacanje()*, koja bi na osnovu odabranog načina plaćanja pozvala odgovarajuću metodu.
2. **State patern** – objektu omogućava promjenu ponašanja na osnovu trenutnog stanja. U kontekstu našeg sistema za prodaju namještaja, state patern možemo primijeniti za upravljanje različitim stanjima narudžbe, npr. “naručeno”, “u obradi”, “isporučeno” i “otkazano”. Ovo bismo postigli definiranjem *State* interfejsa unutar kojeg bismo definirali metode za svako moguće stanje narudžbe, nakon čega bismo imali konkretnu implementaciju interfejsa za svako stanje, npr. *narudzbaNarucena*, *narudzbaUObradi*, *narudzbaIsporucena*, *narudzbaOtkazana*. Interfejs *State* bi povezali sa kontrolerima koji rade sa narudžbama.
3. **TemplateMethod patern** - omogućava izdvajanje određenih koraka algoritma u odvojene podklase. U našem sistemu, ovaj patern bismo mogli primijeniti u svrhu prikaza artikala filtriranih na osnovu boje, cijene ili sobe u kojoj se primarno koristi. Definišemo apstraktnu klasu *FiltriraniArtikli* koja će sadržavati metodu *filtrirajArtikle()*. Biti će povezan sa *ProizvodiCotroller-om*, budući da je on zadužen za prikaz liste proizvoda. Konkretno smo izdvojili dio u kojem se vršilo filtriranje iz funkcije *prikaziProizvode*. Zatim implementiramo konkretne klase koje proširuju ovu klasu i implementiraju metodu prema svojim potrebama. Ova ideja je prikazana na sljedećoj slici.



4. **Chain of Responsibility pattern** - koristi se kada je potrebno da se neke akcije izvršavaju u lancu, u smislu da ako imamo tri radnje, izvršava se prva, pa druga, i naposljetku treća. Dakle, nije moguće nakon izvršenja prve radnje preći na treću, već se mora izvršiti druga nakon prve. Ovaj patern bismo mogli primijeniti na proces dodavanja artikala u korpu i finaliziranja narudžbe. Korisnik prvo bira artikle koje dodaje u korpu, zatim potvrđuje artikle u korpi ili eventualno uređuje količinu tih artikala u korpi, nakon čega unosi podatke o dostavi i plaćanju. Ove radnje se izvršavaju u lancu, odnosno nije moguće unijeti podatke o dostavi i plaćanju ako prije toga uopšte nije bilo artikala u korpi.
5. **Command pattern** – koristi se za enkapsuliranje svih informacija potrebnih za odgođeno izvođenje akcije ili pokretanje događaja. U našem sistemu, ovaj patern bismo mogli primijeniti na slučaj dodavanja artikala u korpu. Prvenstveno bismo implementirali interfejs *Command* koji sadrži samo jednu metodu *execute*, a zatim bismo definisali konkretnu implementaciju ovog interfejsa *DodajUKorpuCommand*. Ova klasa bi primala referencu na *Korpa* objekt, te informaciju o artiklu koji se dodaje. Metoda *execute* bi pozivala metodu *dodajUKorpu(artikal)* nad *Korpa* objektom. *Receiver* objekat (u našem slučaju *Korpa*) izvršava komandu za dodavanje namještaja u korpu.

6. **Iterator pattern** - omogućava pristup elementima kolekcije bez poznavanja kako je kolekcija strukturirana. Iterator patern bismo mogli primijeniti za prolazak kroz listu namještaja. Definiramo interfejs *Iterator*, koji će sadržavati metodu za provjeru postojanja sljedećeg elementa, metodu za dohvaćanje sljedećeg elementa i metodu za uklanjanje elementa. Zatim definiramo interfejs *Kolekcija* koji predstavlja kolekciju namještaja, te ima metodu `createIterator()` koja će vraćati instancu *Iterator*. Implementiramo konkretnu kolekciju namještaja ili narudžbi koja implementira interfejs *Kolekcija*, a nakon toga i konkretni iterator koji prolazi kroz elemente kolekcije. Interfejse povezujemo sa *ProizvodiController*-om, budući da je on “zadužen” za prikaz artikala.



7. **Mediator pattern** - za komunikaciju među objektima bez direktnog povezivanja između njih. Umjesto toga, objekti komuniciraju putem posrednika ili posredničke komponente. Ovaj patern bismo mogli iskoristiti za komunikaciju između zaposlenih. Mediator objekt bi služio kao centralna tačka za razmjenu informacija između zaposlenih unutar sistema. Recimo, kada jedan zaposlenik završi za zahtjevom za naručivanje robe u inventaru, obavijest može poslati mediatoru, koji će tu informaciju dalje proslijediti ostalim uposlenim, kako ne bi došlo do naručivanja viška proizvoda.

8. ***Observer patern*** – uspostavlja relaciju između objekata tako kada jedan objekat promijeni stanje, drugi zainteresovani objekti se obavještavaju. Observer patern primijeniti kako bi pratili promjene u inventaru. Kreiramo klasu *Inventar* koja upravlja artiklima i posmatračima. Posmatrači su klase koje žele biti obavještene o promjenama, kao što su korisnički interfejs, sistem za izvještaje i sistem za obavijesti. Kada se artikli dodaju ili uklone iz inventara, klasa *Inventar* obavještava sve registrovane posmatrače o promjenama. Posmatrači implementiraju interfejs posmatrača i definišu konkretne akcije koje preduzimaju na obavijest. Recimo, korisnički interfejs može osvježiti prikaz, sistem za izvještaje može ažurirati podatke, a sistem za obavijesti može poslati notifikacije korisnicima.
9. ***Visitor patern*** - omogućuje dodavanje novih operacija ili funkcionalnosti objektima bez mijenjanja njihove strukture, tj. koristi se za oblikovanje i manipulaciju objektima u strukturama podataka, bez promjene samih objekata. Vezano za naš sistem, pogodno bi ga bilo primijeniti kada bismo imali opciju određenog popusta na artikle. To bi podrazumijevalo da definiramo *Visitor* interfejs koji bi sadržao metode za svaku vrstu namještaja koju želimo posjetiti (*visitSto*, *visitStolica*...), nakon čega bismo implementirali konkretne “visitore” koji implementiraju *Visitor* interfejs za primjenu popusta. Definiramo *Element* interfejs, koji predstavlja različite tipove namještaja, te konkretne tipove namještaja (*Sto*, *Stolica*, *Sofa*...) koji implementiraju ovaj interfejs, te sadrže metodu *accept* koja prima posjetitelja kao argument i poziva odgovarajuću metodu posjetitelja.
10. ***Interpreter patern*** – koristi se za interpretaciju definicija jezika, te omogućuje interpretaciju rečenica izraženih u tom jeziku. Ovaj patern bismo mogli iskoristiti za interpretaciju upita korisnika. Kada korisnik unese neki upit, npr. “Prikaži sve crvene stolice jeftinije od 200 KM”, sistem bi koristio interpretatore za “crvene”, “stolice” i “jeftinije od 200 KM” kako bi generirao odgovarajući upit prema bazi podataka i rezultate prikazao korisniku.

11. ***Memento patern*** - omogućuje snimanje trenutnog stanja objekta i vraćanje tog stanja u budućnosti, vez otkrivanja detalja njegove implementacije ili narušavanja njegove privatnosti. U slučaju našeg sistema, ovaj patern bismo mogli iskoristiti recimo za pamćenje stanja korpe prije nego što korisnik započne proces plaćanja. Na taj način, ako korisnik prekine proces plaćanja ili se vrati kasnije, može mu se omogućiti povratak u stanje u kojem je bio prije nego što je započeo proces plaćanja.

