

Specifikacija projekta

1. Osnovne informacije o sistemu

Naziv teme: Društvena mreža

Logo:



Naziv tima: Team8

Nastavna grupa: RS3

Link na repozitorij tima: <https://github.com/OOAD-2023-2024/Team8.git>

Članovi tima:

1. Amir Hastor, 19829
2. Fuad Jaganjac, 19830
3. Amer Mehmedić, 19832
4. Amina Frljak, 19828

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).

Radi se o web aplikaciji koja omogućava korisnicima kreiranje profila(registraciju), prijavu, kao i editovanje svog profila. Aplikacija omogućava jednostavno postavljanje objava koje su vidljive ostalim korisnicima. Ostali korisnici mogu interaktivirati sa tim objavama na način da ostave lajk ili komentar koji su vidljiv svim ostalim korisnicima.

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:** Kreiranje korisničkog računa i prijava

Vrsta funkcionalnosti: Perzistencija podataka (CRUD operacije)

Opis funkcionalnosti:

Korisnik mora moći registrirati svoj profil na aplikaciji. Obrazac za registraciju mora tražiti korisničko ime, e-mail i lozinku. Korisničko ime mora biti jedinstveno, e-mail mora imati valjani domen i lozinka mora sadržavati između 5 i 10 znakova. Korisnik mora moći prijaviti se sa e-mailom i lozinkom, ako korisnik unese neispravne podatke za prijavu, ne smije se autentificirati i autorizirati, a ukoliko korisnik unese ispravne podatke za prijavu, pravi se korisnička sesija (npr. JWT) i aplikacija preusmjerava korisnika na stranicu sa vijestima. Korisnik mora moći se odjaviti iz aplikacije, čime se zatvara sesija.

2) **Naziv funkcionalnosti:** Izmjena postavki na profilu i izgled profila

Vrsta funkcionalnosti: Perzistencija podataka (CRUD operacije)

Opis funkcionalnosti: Korisnik mora moći posjetiti svoj profil klikom na ikonu profila. Korisnik mora moći promijeniti profilnu sliku, prebacivati se između tamne i svjetle UI teme, uključiti ili isključiti obavijesti, te moći vidjeti najnovija tri posta.

3) **Naziv funkcionalnosti:** Prikaz objava sa preporukom korisniku

Vrsta funkcionalnosti: Operacija sa specifičnim algoritmom obrade

Opis funkcionalnosti:

Prijavljeni korisnik mora moći vidjeti blog objave svih drugih korisnika. Svaka objava mora sadržavati sliku posta, opis, broj svidanja i komentara. Objave se prikazuju korisniku tako što se prvo izlistaju objave koje su preporučene tom korisniku, odnosno objave koje sadrže tagove koje korisnik najviše koristi.

4) **Naziv funkcionalnosti:** Pretraživanje sistema

Vrsta funkcionalnosti: Usluga sistema

Opis funkcionalnosti:

Polje za pretragu mora biti na vrhu stranice sa vijestima.

Polje za pretragu se koristi za pronalazak drugih korisnika samo po korisničkom imenu. Ako se pronađe željeni korisnik, pronađeni korisnički zapis je klikabilan. Stranica korisničkog profila mora prikazivati najnovija tri posta, korisničko ime i profilnu sliku. Korisnički profil ne smije se moći uređivati jer to nije profil koji pripada trenutnom korisniku sesije.

5) **Naziv funkcionalnosti:** Verifikacija podataka pomoću 2FA aplikacije (Google Authenticator)

Vrsta funkcionalnosti: Korištenje vanjskog uređaja

Opis funkcionalnosti:

Korisnici mora biti u mogućnosti da registriira svoje uređaje s Google Authenticator aplikacijom putem web aplikacije. Prilikom prijave ili obavljanja osjetljivih radnji, korisnici će morati unijeti jednokratni kôd koji će dobiti putem Google Authenticator aplikacije.

6) **Naziv funkcionalnosti:** Kreiranje nove objave

Vrsta funkcionalnosti: Perzistencija podataka (CRUD operacije) i usluga sistema

Opis funkcionalnosti:

Aplikacija mora imati dugme "Create a Post". Nakon klika na dugme, korisnika treba preusmjeriti na zasebnu stranicu koja mora sadržavati obrazac koji traži sliku objave i tekst objave. Obrazac mora imati dugme "Cancel" i "Save". Dugme "Cancel" otkazuje trenutni zahtjev i preusmjerava korisnika natrag na početnu stranicu. Dugme "Save" sprema trenutni zahtjev i preusmjerava korisnika na početnu stranicu.

7) **Naziv funkcionalnosti:** Pregled notifikacija

Vrsta funkcionalnosti: Asinhrona operacija

Opis funkcionalnosti:

Korisnik mora moći vidjeti dugme za prikaz obavijesti. Nakon klika na dugme, treba se prikazati najnovije tri obavijesti (ako postoje). Postoje dvije vrste obavijesti: kada drugi korisnik ostavi lajk na tvoju objavu, kada drugi korisnik ostavi komentar na tvoju objavu.

8) **Naziv funkcionalnosti:** Interakcija sa objavama

Vrsta funkcionalnosti: Usluga sistema

Opis funkcionalnosti: Korisnici moraju moći ostaviti lajk ili komentar na blog post.

9) **Naziv funkcionalnosti:** Brisanje korisničkog računa

Vrsta funkcionalnosti: Perzistencija podataka (CRUD operacije) i usluga sistema

Opis funkcionalnosti: Jedna od funkcionalnosti administratora je i mogućnost brisanja registrovanih korisnika. Neki od razloga su govor mržnje, ili neprimjereno ponašanje.

10) **Naziv funkcionalnosti:** Dijeljenje objava

Vrsta funkcionalnosti: Usluga sistema

Opis funkcionalnosti:

Korisnici bi trebali imati mogućnost dijeljenja objava sa drugim korisnicima na aplikaciji. To će im se pojaviti kao notifikacija.

11) **Naziv funkcionalnosti:** Dodavanje tag-ova na objave

Vrsta funkcionalnosti: Usluga sistema

Opis funkcionalnosti:

Korisnik bi trebao imati mogućnost dodavanja jednog ili više tag-ova na novu objavu koristeći hashtag simbol (#) uz naziv željene teme kako bi objave bile kategorizirane prema sličnostima ili ključnim riječima s ciljem da se korisnicima uslužuju objave u skladu sa temama koje su naveli kao relevantnim na svom profilu.

12) Naziv funkcionalnosti: Prijava objava

Vrsta funkcionalnosti: Usluga sistema

Opis funkcionalnosti:

Korisnik mora imati mogućnost da prijave sadržaj (objave ili komentare) drugih korisnika koji im se čine neprimjerenim ili da na bilo koji način krše pravila korištenja aplikacije. Prijave se šalju moderatoru i administratoru nakon čega oni imaju opciju da pregledaju objavu koja je prijavljena i uklone je po potrebi.

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:** Korisnik

Vrsta aktera: Korisnik sistema

Funkcionalnosti u kojima akter učestvuje:

Funkcionalnost sistema	Način učešća
1 - Registracija, prijava i odjava	Mogućnost uređivanja
2 - Izmjena postavki na profilu i izgled profila	Mogućnost uređivanja i pregleda
3 - Prikaz objava sa preporukom korisniku	Mogućnost pregleda
4 - Pretraživanje sistema	Mogućnost pregleda
7 - Pregled notifikacija	Mogućnost pregleda

6 - Kreiranje nove objave	Mogućnost uređivanja
8 - Interakcija sa objavama	Mogućnost uređivanja
5 - Verifikacija podataka pomoću 2FA aplikacije (Google Authenticator)	Mogućnost uređivanja
11 - Dodavanje tag-ova na objave	Mogućnost uređivanja
12 - Prijava objava	Mogućnost uređivanja

b) **Naziv aktera:** Moderator

Vrsta aktera: Zaposlenik sistema

Funkcionalnosti u kojima akter učestvuje:

Funkcionalnost sistema	Način učešća
7 - Pregled notifikacija	Mogućnost uređivanja
8 - Interakcija sa objavama	Mogućnost uređivanja
12 - Prijava objava	Mogućnost pregleda

c) **Naziv aktera:** Administrator

Vrsta aktera: Administrator

Funkcionalnosti u kojima akter učestvuje:

Način učešća:

- *Mogućnost pregleda*
- *Mogućnost uređivanja*

Funkcionalnost sistema	Način učešća
1 - Registracija, prijava i odjava	Mogućnost uređivanja
2 - Izmjena postavki na profilu i izgled profila	Mogućnost uređivanja



9 - Brisanje korisničkog računa	Mogućnost uređivanja
8 - Interakcija sa objavama	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

Opis:

Aplikacija mora imati prosječno vrijeme odgovora manje od 1 sekunde za većinu korisničkih zahtjeva. Maksimalno vrijeme prijave korisnika ne smije prelaziti 3 sekunde.

2) **Naziv nefunkcionalnog zahtjeva:** Sigurnost

Opis:

Lozinke korisnika moraju biti pohranjene u sigurnoj bazi podataka sa heširanim vrijednostima. Pristup korisničkim podacima i osobnim informacijama mora biti strogo kontroliran i zaštićen od neovlaštenog pristupa.

3) **Naziv nefunkcionalnog zahtjeva:** Pouzdanost

Opis:

Aplikacija ne smije dozvoliti gubitak podataka. Svi korisnički podaci i objave moraju biti sigurno pohranjeni i dostupni. Ne smije biti čestih pada aplikacije ili grešaka u radu koje bi utjecale na korisničko iskustvo.

4) **Naziv nefunkcionalnog zahtjeva:** User Agreement (Saglasnost Korisnika)

Opis:

Korisnik mora prihvatiti uslove korištenja aplikacije. Dokument predstavlja specifikaciju koja određuje standarde i uvjete koje aplikacija moraju zadovoljiti u pogledu pravnih obveza, pravila privatnosti, sigurnosnih standarda i etičkih smjernica. Zahtjeva se da korisnici jasno razumiju pravila i uvjete korištenja servisa, čime se osigurava usklađenost s relevantnim zakonima i propisima te unapređuje povjerenje korisnika u aplikaciju.

[Dokument](#)

1. KREIRANJE KORISNIČKOG RAČUNA

Naziv slučaja upotrebe	Kreiranje korisničkog računa
Opis	Registracija korisničkog profila na aplikaciji
Vezani zahtjevi	/
Preduslovi	/
Posljedice – uspješan zadatak	Račun je uspješno kreiran
Posljedice – neuspješan zadatak	Račun nije kreiran
Primarni akteri	Korisnik
Ostali akteri	/
Glavni tok	Korisnik unosi tražene podatke za kreiranje novog korisničkog računa na aplikaciji

Tok događaja - uspješan završetak:

Korisnik	Sistem
1. Izbor opcija za kreiranje računa	
2. Unos potrebnih podataka za registraciju	
	3. Validacija podataka
	4. Kreiranje korisničkog računa i dodavanje profila u bazu podataka
5. Pristup aplikaciji	

Tok događaja - neuspješan završetak:

Korisnik	Sistem
1. Izbor opcija za kreiranje računa	
2. Unos potrebnih podataka za registraciju	
	3. Validacija podataka
	4. Obavješćavanje korisnika o nemogućnosti kreiranja računa
5. Ponovni unos ispravnih podataka za registraciju	

2. PRIJAVA

Naziv slučaja upotrebe	Prijava na korisnički račun
Opis	Korisnik se prijavljuje na korisnički račun unoseći e-mail i lozinku
Vezani zahtjevi	/
Preduslovi	Kreiran korisnički račun
Posljedice – uspješan zadatak	Korisnik pristupa profilu
Posljedice – neuspješan zadatak	Korisnik se nije prijavio na profil
Primarni akteri	Korisnik
Ostali akteri	/
Glavni tok	Registrovani korisnik unosi tražene podatke za pristup korisničkom profilu

Tok događaja - uspješan završetak:

Korisnik	Sistem
1. Unos emaila i lozinke	
	2. Validacija podataka
3. Uspješan pristup aplikaciji	

Tok događaja - neuspješan završetak:

Korisnik	Sistem
1. Unos emaila i lozinke	
	2. Validacija podataka
	3. Potvrda o neispravnim podacima
4. Mogućnost ponovnog pristupa aplikaciji	

3. IZMJENA PODATAKA NA KORISNIČKOM PROFILU

Naziv slučaja upotrebe	Izmjena postavki na korisničkom profilu
Opis	Nakon što se korisnik uspješno prijavi na aplikaciju, ima mogućnost da mijenja postavke na vlastitom korisničkom računu uključujući promjenu slike, UI, obavijesti.
Vezani zahtjevi	/
Preduslovi	Uspješna prijava na korisnički račun
Posljedice – uspješan zadatak	Mogućnost izmjene postavki na računu
Posljedice – neuspješan zadatak	Korisnik se nije prijavio na profil
Primarni akteri	Korisnik
Ostali akteri	/
Glavni tok	Registrovani korisnik ima mogućnost da mijenja postavke i podatke na svom korisničkom računu

Tok događaja - uspješan završetak:

Korisnik	Sistem
1. Prijava na sistem	
2. Odabriri opcije za uređivanje profila	
	3. Prikaz formulara za uređivanje formulara
4. Mogućnost mijenjanja postavki i podataka na korisničkom računu	
	5. Provjera ispravnosti podataka
	6. Ažuriranje podataka u bazi
	7. Obavijest o usješno ažuriranim podacima

Tok događaja - neuspješan završetak:

Korisnik	Sistem
1. Prijava na sistem	
2. Odabriri opcije za uređivanje profila	
	3. Prikaz formulara za uređivanje formulara
4. Mogućnost mijenjanja postavki i podataka na korisničkom računu	
	5. Obavijest o neuspješnoj validaciji

4. PRIKAZ OBJAVA SA PREPORUKOM KORISNIKU

Naziv slučaja upotrebe	Pretraživanje sistema
Opis	Prijavljeni korisnik mora moći vidjeti blog objave svih drugih korisnika. Svaka objava mora sadržavati sliku posta, opis, broj sviđanja i komentara. Objave se prikazuju korisniku tako što se prvo izlistaju objave koje su preporučene tom korisniku, odnosno objave koje sadrže tagove koje korisnik najviše koristi.
Vezani zahtjevi	/
Preduslovi	Uspješna prijava na korisnički račun
Posljedice – uspješan zadatak	Uspješan prikaz preporučenih objava korisniku ako ima takvih
Posljedice – neuspješan zadatak	/
Primarni akteri	Korisnik
Ostali akteri	/
Glavni tok	Registrovani korisnik otvara home page i izlistavaju mu se preporučene objave, ako postoje.

Tok događaja - uspješan završetak:

Korisnik	Sistem
1. Prijava na aplikaciju	
2. Otvaranje home stranice	
	3. Izlistavanje preporučenih objava
4. Pregled preporučenih i ostalih objava	

5. PRETRAŽIVANJE SISTEMA

Naziv slučaja upotrebe	Pretraživanje sistema
Opis	Nakon što se korisnik uspješno prijavi na aplikaciju, ima mogućnost pronalaska drugih korisničkih profila. Nakon što uspješno pronađe korisnika, ima mogućnost prikaza profilne slike i tri posta od drugog korisnika. Također, korisnici moraju moći ostaviti lajk ili komentar na blog post ako postoji
Vezani zahtjevi	/
Preduslovi	Uspješna prijava na korisnički račun
Posljedice – uspješan zadatak	Uspješan pronalazak korisničkog računa i pregled i interakcija sa profilom
Posljedice – neuspješan zadatak	Korisnik nije uspio pronaći korisnički račun
Primarni akteri	Korisnik
Ostali akteri	/
Glavni tok	Registrovani korisnik pronalazi traženog korisnika, gdje ima mogućnost da ostavi lajk ili komentar na post kao i dijeljenje objava

Tok događaja - uspješan završetak:

Korisnik	Sistem
1. Unos emaila i lozinke - prijava na korisnički račun	
2. Pretraživanje drugih korisnika	
	3. Usješan pronalazak drugih korisnika
4. Pregled korisničkog profila	

Tok događaja - neuspješan završetak:

Korisnik	Sistem
1. Unos emaila i lozinke	
2. Pretraživanje drugih korisnika	
	3. Sistem nije pronašao traženog korisnika
4. Mogućnost ponovnog pretraživanja korisnika	

6. 2FA AUTENTIFIKACIJA

Naziv slučaja upotrebe	2FA AUTENTIFIKACIJA
Opis	Ako ima uključenu opciju 2FA autentifikacije, korisnik unosi kod sa telefona za prijavu.
Vezani zahtjevi	/
Preduslovi	Kreiran profil, aktiviran 2FA
Posljedice – uspješan zadatak	Uspješna prijava pomoću 2FA
Posljedice – neuspješan zadatak	Neuspješna prijava
Primarni akteri	Korisnik
Ostali akteri	/
Glavni tok	Korisnik unosi 2FA kod sa telefona za prijavu, ako nije tačan, vraća se poruka da nije ispravan kod.

Tok događaja - uspješan završetak:

Korisnik	Sistem
1. Odabir opcije za 2FA prijavu	
2. Unos 2FA koda sa telefona	
	3. Vršiti se provjera ispravnosti koda
4. Korisnik uspješno prijavljen	

Tok događaja - neuspješan završetak:

Korisnik	Sistem
1. Odabir opcije za 2FA prijavu	
2. Unos 2FA koda sa telefona	
	3. Vršiti se provjera ispravnosti koda
	4. Obavješćavanje korisnika o neispravnom kodu
5. Korisnik neuspješno prijavljen	

7. KREIRANJE NOVE OBJAVE

Naziv slučaja upotrebe	Kreiranje nove objave
Opis	Nakon što se korisnik uspješno prijavi na aplikaciju, ima mogućnost kreiranja nove objave na aplikaciji.
Vezani zahtjevi	/
Preduslovi	Uspješna prijava na korisnički račun
Posljedice – uspješan zadatak	Kreiranje nove objave
Posljedice – neuspješan zadatak	Korisnik nije uspio kreirati objavu
Primarni akteri	Korisnik
Ostali akteri	/
Glavni tok	Registrovani korisnik ima mogućnost da uspješno kreira novu objavu klikom da dugme Save.

Tok događaja - uspješan završetak:

Korisnik	Sistem
1. Prijava na korisnički račun	
2. Mogućnost kreiranja novog posta	
3. Dodavanje i izbor novog posta	
4. Dodavanje tagova na objave	
5. Potvrda o spremanju trenutnog zahtjeva	
	6. Ažuriranje podataka u sistemu
7. Mogućnost pregleda ažuriranog profila	

Tok događaja - neuspješan završetak:

Korisnik	Sistem
1. Prijava na korisnički račun	
2. Mogućnost kreiranja novog posta	
3. Dodavanje i izbor novog posta	
4. Dodavanje tagova na objave	
5. Otkazivanje trenutnog zahtjeva	
	6. Vraćanje korisničkog profila na početni ekran

8. PREGLED NOTIFIKACIJA

Naziv slučaja upotrebe	Pregled notifikacija
Opis	Korisnik dobija obavijest ukoliko dobije lajk ili odgovor na objavu od strane drugog korisnika, kao i ako neko podijeli objavu. Također ukoliko postoje neki neprimjereni komentari ili nešto što krši prava korištenja aplikacije, šalje se obavijest moderatorima ili administratoru aplikacije koji to dalje pregledavaju i uklanjaju ako je to potrebno
Vezani zahtjevi	Interakcija sa objavama
Preduslovi	Uspješna prijava na korisnički račun Davanje komentara od strane drugog korisnika
Posljedice – uspješan zadatak	Korisnik je dobio obavijest
Posljedice – neuspješan zadatak	/
Primarni akteri	Korisnik, Zaposlenik, Administrator
Ostali akteri	/
Glavni tok	Korisnik dobija obavijest ako drugi korisnik ostavi komentar ili lajk na njihovu objavu

Tok događaja - uspješan završetak:

Korisnik	Sistem	Zaposlenik/Administrator
1. Prijava na korisnički račun		
2. Ulazak na korisnički profil i komentarisanje ili ostavljanje lajka na objavu		
	3. Prijem podataka po potrebi	
		4. Prijem obavijesti o prijavama
5. Prijem obavijesti		

9. BRISANJE I UPRAVLJANJE KORISNIČKIM PROFILIMA I OBJAVAMA

Naziv slučaja upotrebe	Brisanje i upravljanje korisničkim profilima
Opis slučaja upotrebe	Mogućnost brisanja i upravljanja korisničkih profila
Vezani zahtjevi	-
Preduvjeti	U slučaju da korisnik ostavi zlonamjeran komentar ili objavu, administrator i moderator imaju mogućnost da brišu komentare i objave. Pored toga, administrator ima mogućnost da briše sve korisnike, a moderator sve obične korisnike
Posljedice - uspješni završetak	Uspješno obrisani komentari, objava ili korisnički profil
Posljedice - neuspješni završetak	-
Primarni akteri	Administrator, Moderator
Ostali akteri	Korisnik
Glavni tok	Ako korisnik ostavi zlonamjeran komentar ili objavu, administrator i moderator imaju mogućnost da izbrišu korisnički profil zbog neprimjerenog sadržaja

Tok događaja - uspješno brisanje korisničkog profila

Korisnik	Administrator/Moderator
1. Slanje prijave o zlonamjernom komentaru ili objavi	
	2. Procjenjuje da li objava ili komentar krše pravila stranice
	3. Briše komentar ili objavu koji ruše pravila
	4. Mogućnost brisanja korisničkog profila koji često krši pravila stranice

10. INTERAKCIJA SA OBJAVAMA

Naziv slučaja upotrebe	Interakcija sa objavama
Opis slučaja upotrebe	Mogućnost ostavljanja lajka i komentara na objavu
Vezani zahtjevi	-
Preduvjeti	Da je korisnik prijavljen
Posljedice - uspješni završetak	Korisnik uspješno ostavi komentar i lajk
Posljedice - neuspješni završetak	-
Primarni akteri	Korisnik, Sistem
Ostali akteri	
Glavni tok	Korisnik ode na home page, i ostavi lajk i komentar na neku objavu

Tok događaja - uspješni završetak

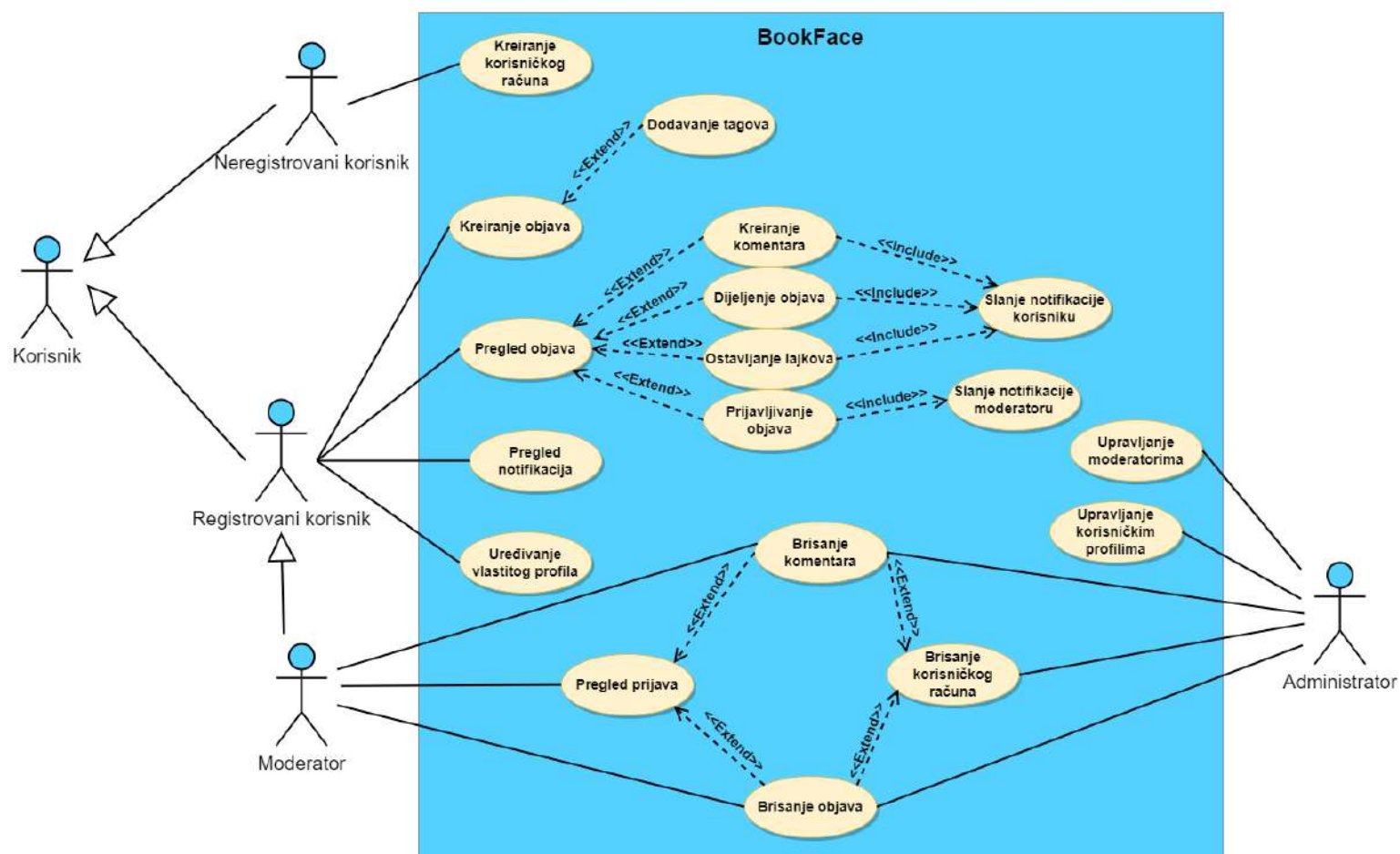
Korisnik	Sistem
1. Otvori home stranicu	
2. Ostavlja like i/ili komentar	
	3. Pohranjiva se komentar ili lajk u sistem

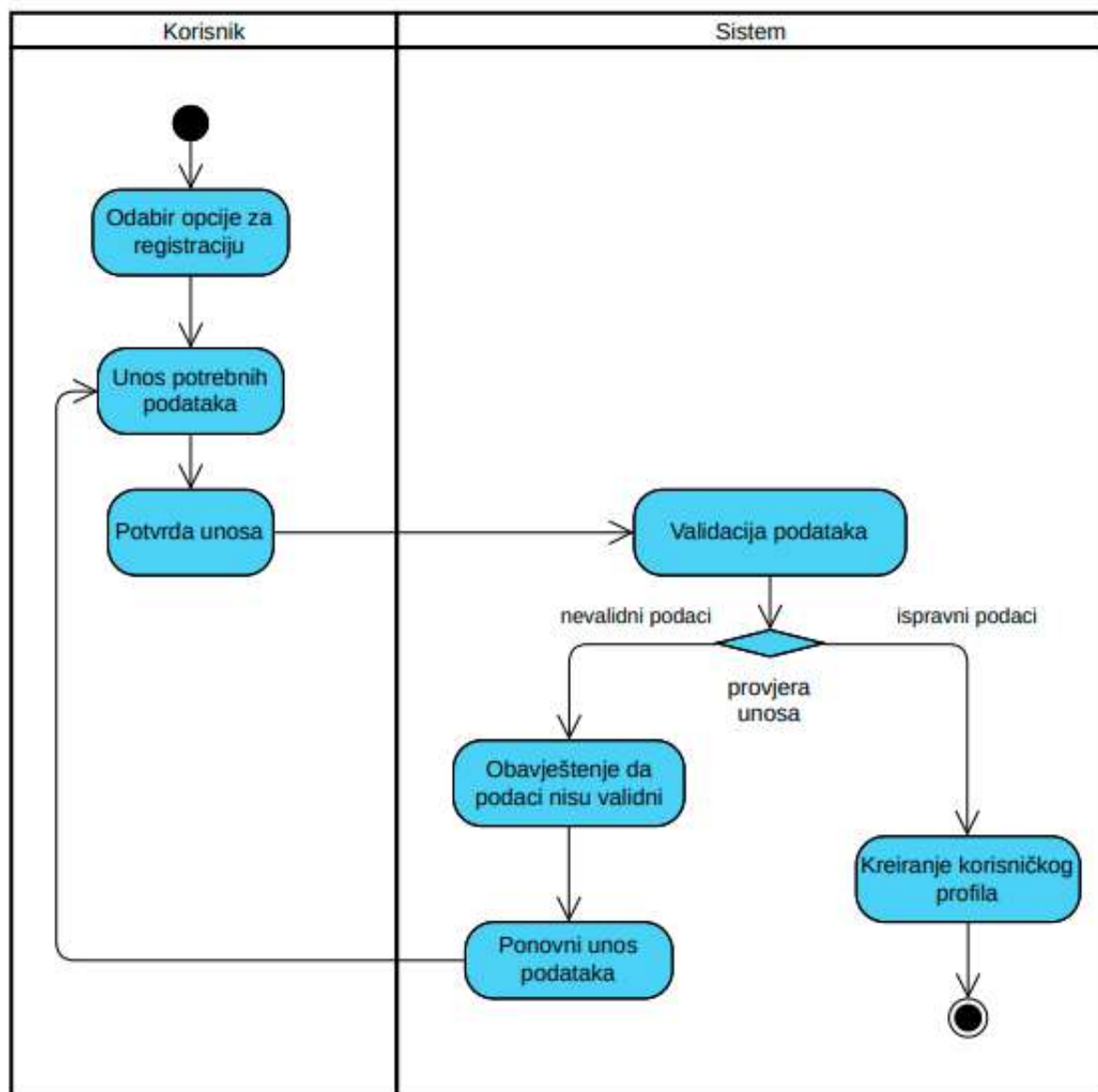
11. PRIJAVA OBJAVE

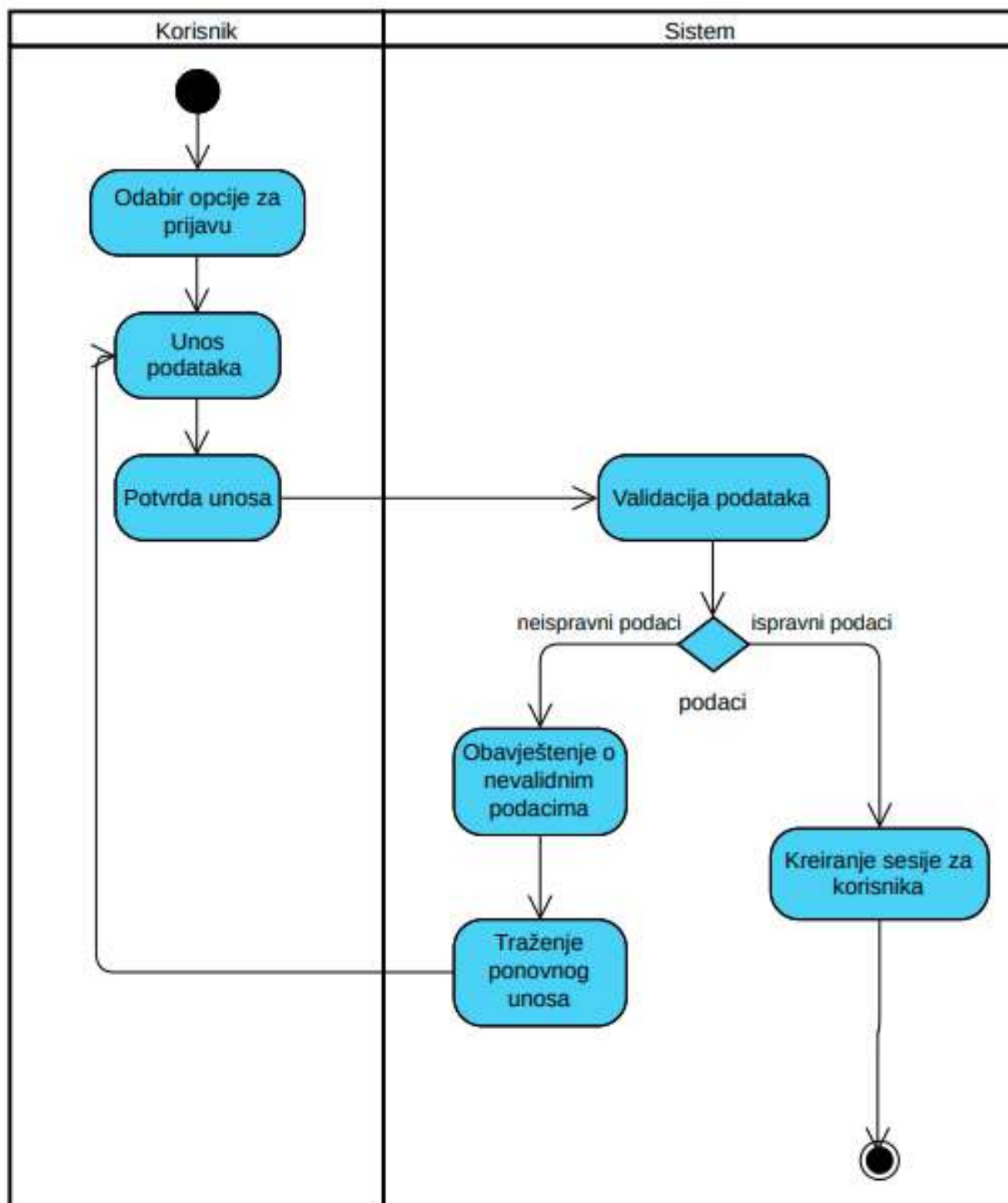
Naziv slučaja upotrebe	Prijava objave ili komentara
Opis slučaja upotrebe	Mogućnost prijavljivanja zloupotrebe komentara ili objave
Vezani zahtjevi	Interkacija sa objavama
Preduvjeti	Da je korisnik prijavljen
Posljedice - uspješni završetak	Korisnik uspješno prijavi komentar ili objavu
Posljedice - neuspješni završetak	-
Primarni akteri	Korisnik, Sistem
Ostali akteri	
Glavni tok	Korisnik odabere opciju za prijavu objave ili komentara koji ruše pravila stranice

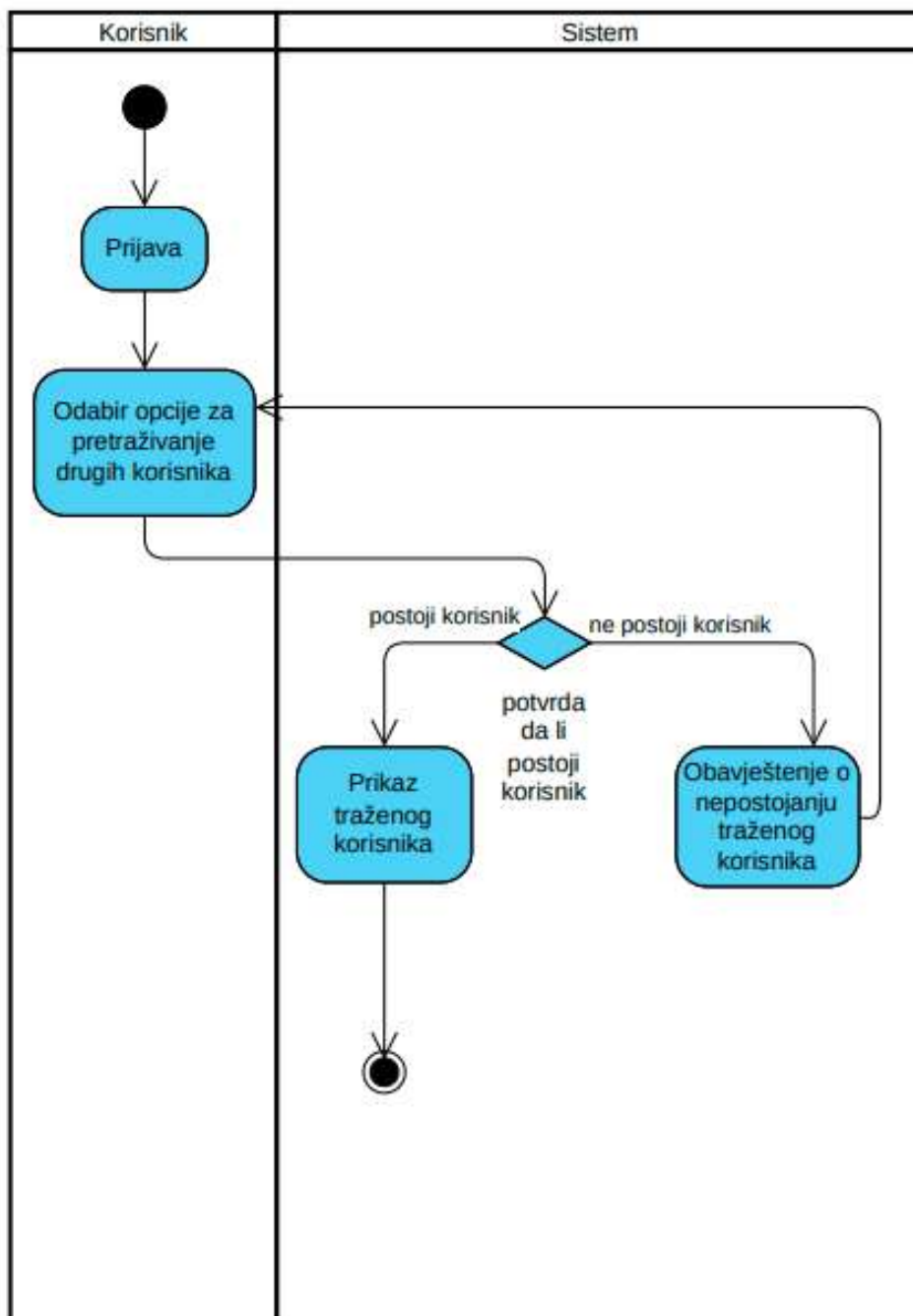
Tok događaja - uspješni završetak

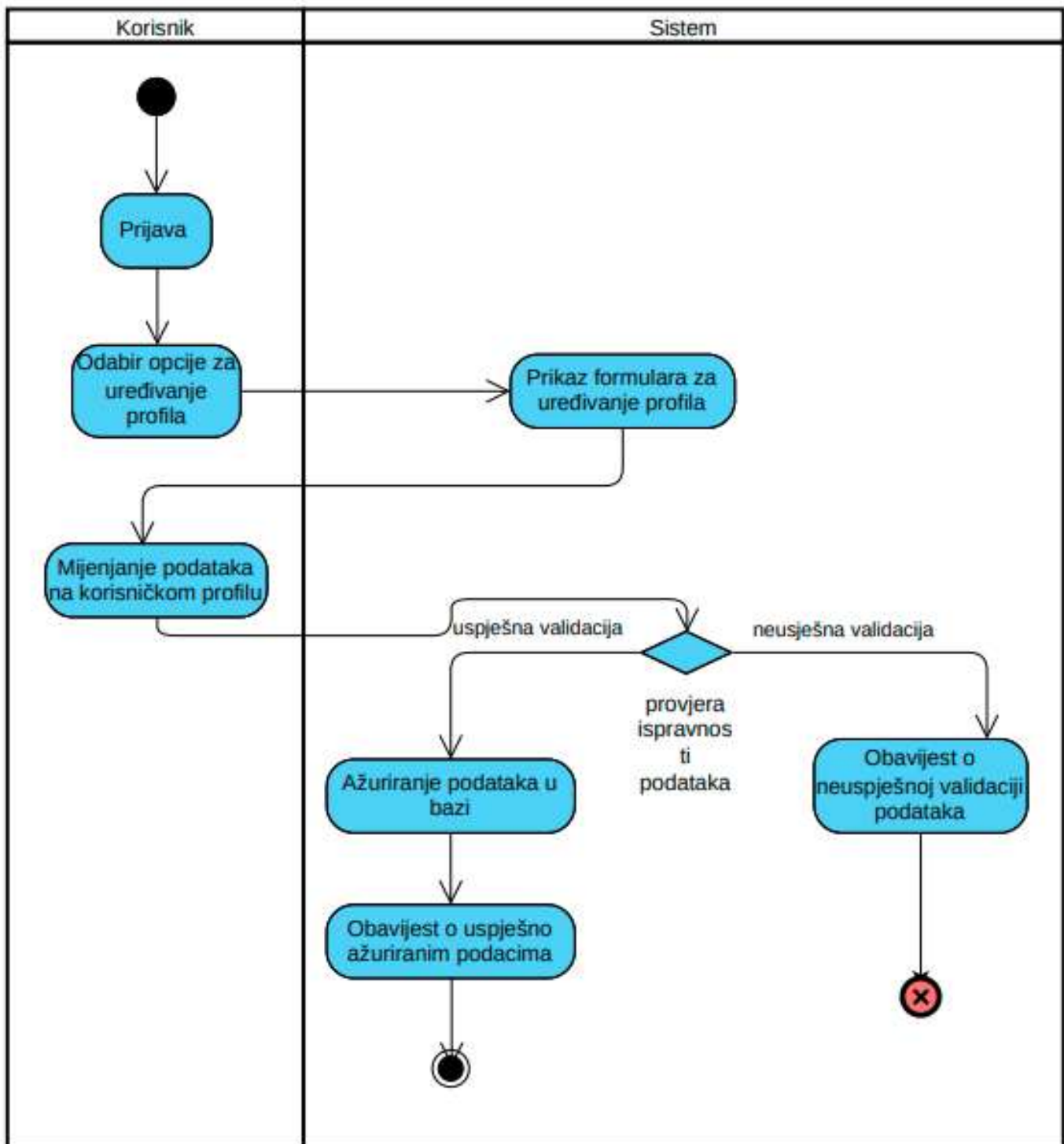
Korisnik	Sistem
1. Korisnik pristupa home stranici	
2. Odabira opciju za prijavu komentara ili objave	
	3. Prijava se dostavlja moderatoru ili adminu

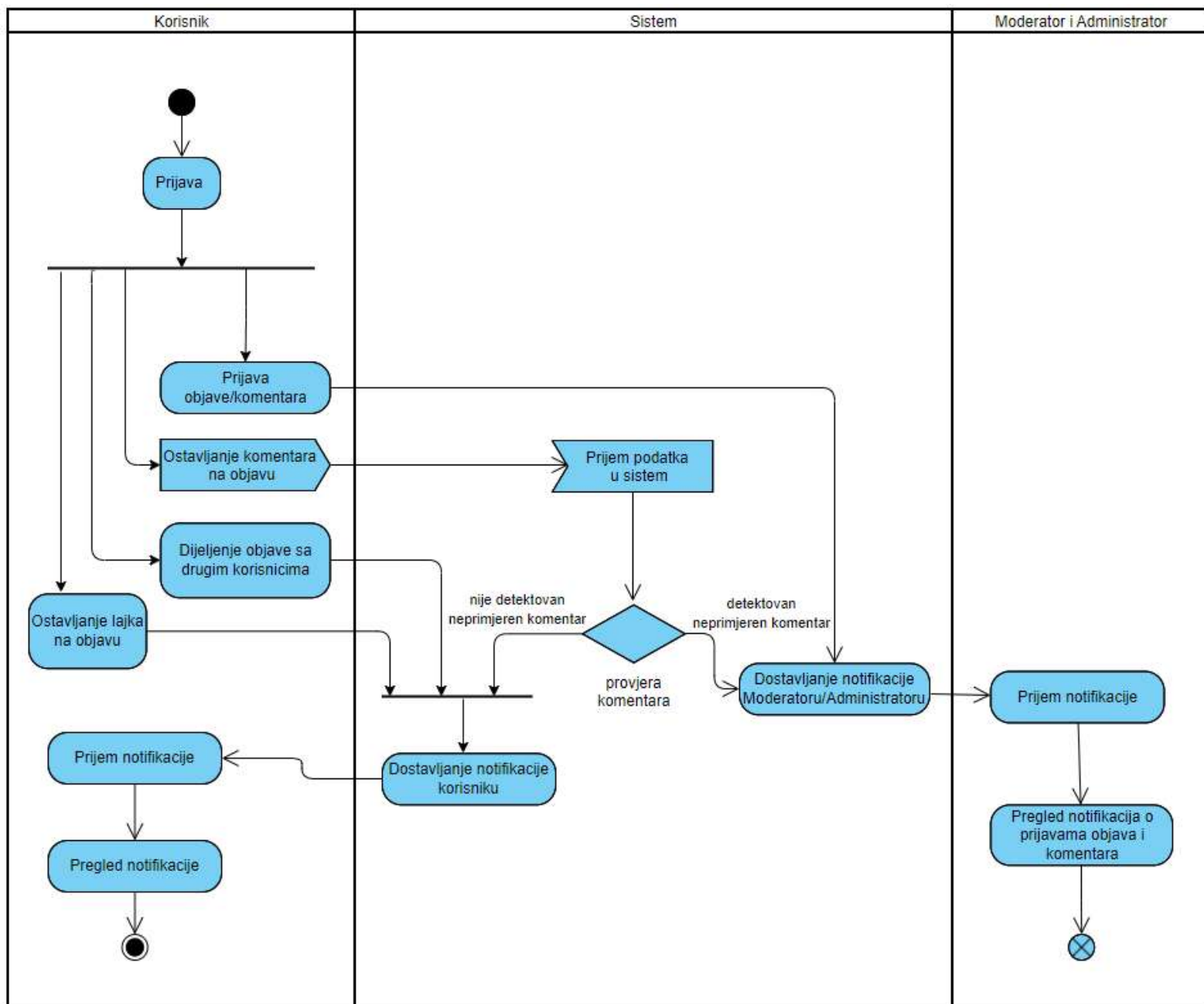


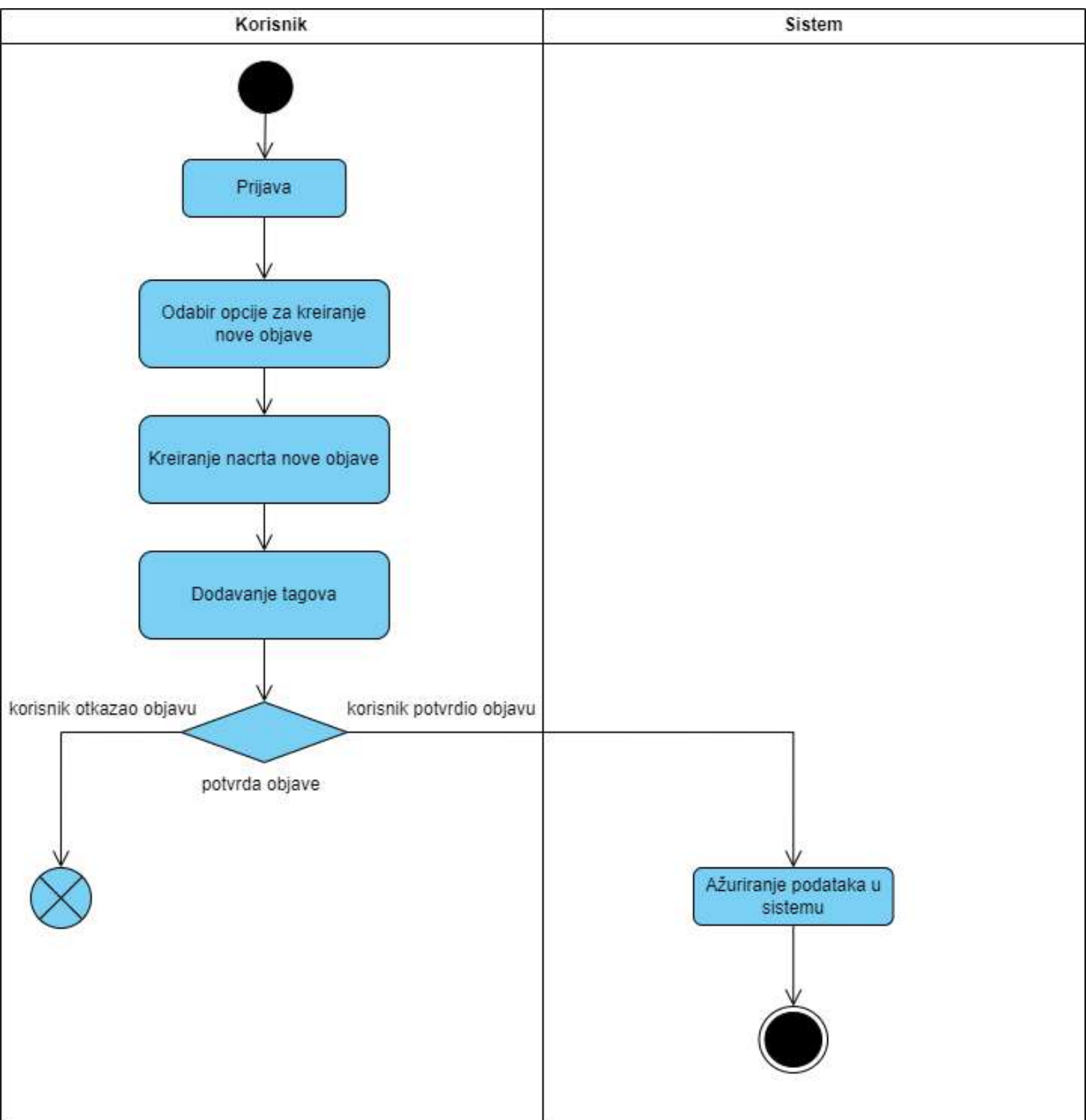


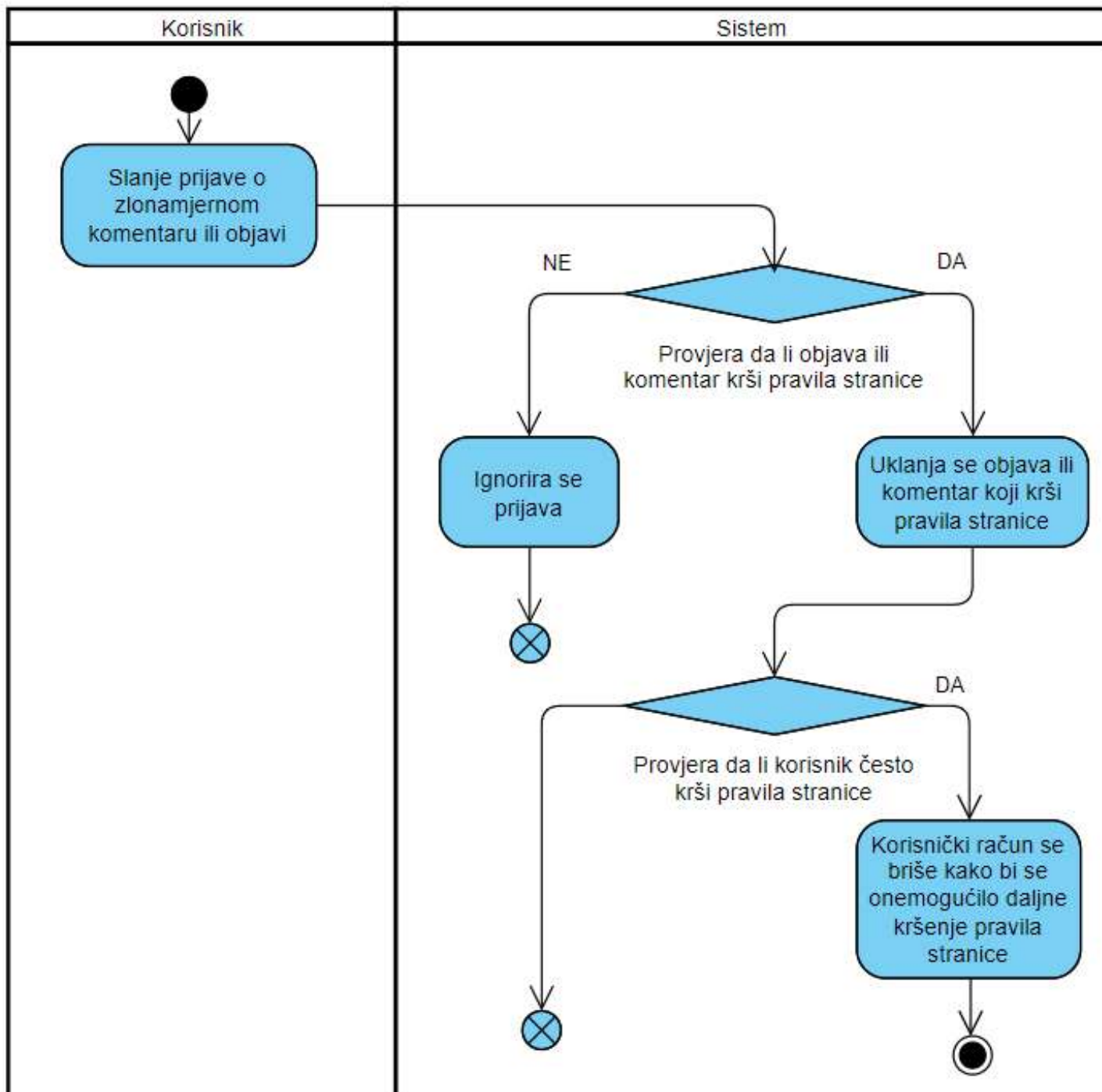
















Log in

Log in by entering your email address and password.

Email address

 email@address.com

Password

 ●●●●●●●●●● 

Log in

Don't have an account? [Sign up here](#)



Log in

Enter your 2 factor authentication code

--	--	--	--	--	--

Continue



Create New Account

Username

Email address



Password



Create Account



mujo



neko.nekic



3



3



#nature #tree #beautiful

mujo.mujic




3



3









#horizon #meadow #mountains

 neko.nekic



Comments:

- | | | | |
|---|--------------|--------------------|---|
|  | mujo.mujic: | Vau super! |  |
|  | vaso.vasicc: | Ovo je bas lijepo! |  |
|  | mijo.mijic | Meni i nije nesto |  |

Leave a comment:

Write something





seki.sekic reported a post by neko.nekic

[View report](#)


miki.mikic reported a post by neko.nekic


[View report](#)


miki.mikic reported a comment by neko.nekic

[View report](#)




 seki.sekic reported this post by neko.nekic:


 neko.nekic




Comments:


 mujo.mujic:


Vau super!




 vaso.vasicc:

Ovo je bas lijepo!



 mijo.mijic

Meni i nije nesto



**View offender's
profile**

**Deleted reported
content**

Disregard report



neki_e

Edit Profile

3 posts

27 likes

Most Recent posts:





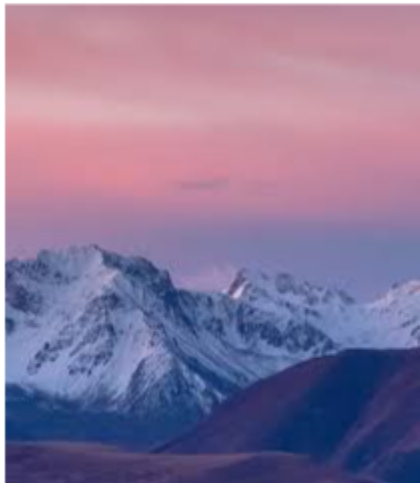
nekic_e

Delete Profile

3 posts

27 likes

Most Recent posts:



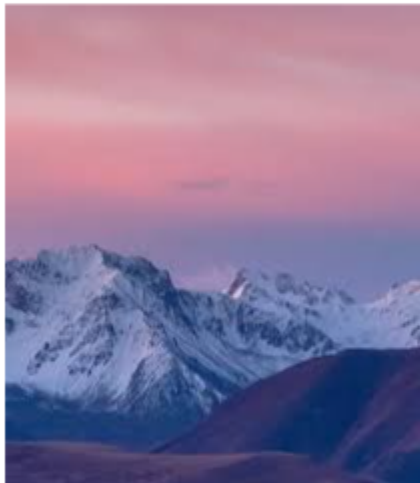


nekic_e

3 posts

27 likes

Most Recent posts:





Edit Profile



neki_e

Change profile photo

Notifications enabled:



Dark mode enabled



2 Factor Authentication



Name

Placeholder

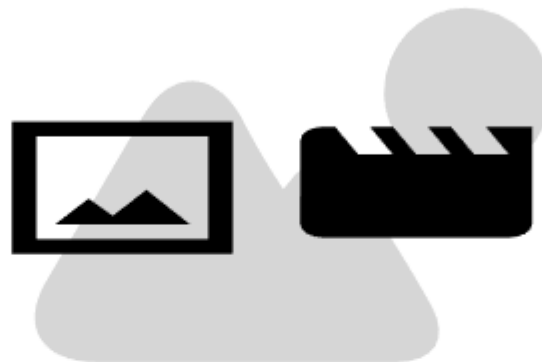
Username

Placeholder

SAVE



Create a Post



UPLOAD PHOTOS OR VIDEOS HERE

Add caption, tags

Cancel

Save

Analiza i dizajn sistema

Definicija klase u sistemu

Naziv klase: Korisnik

Funkcionalni zahtjevi u kojima klasa učestvuje:

FZ br. 1: Kreiranje korisničkog računa i prijave
FZ br. 2: Izmjena postavki na profilu i izgled profila
FZ br. 4: Pretraživanje sistema
FZ br. 5: Verifikacija podataka pomoću 2FA aplikacije (Google Authenticator)
FZ br. 6: Kreiranje nove objave
FZ br. 7: Pregled notifikacija
FZ br. 12: Prijava objava

Atributi koje klasa posjeduje:

Naziv atributa	Tip varijable	Dodatne napomene
korisnikId	Integer	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
nazivKorisnika	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>
sifraHash	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
sifraSalt	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
slikaProfila	Image	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
JWT	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
darkTheme	Bool	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
datumKreiranjaProfila	Datetime	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
2FAEnabled	Bool	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
notifikacijeEnabled	Bool	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>

tipKorisnika	TipKorisnika	<input type="checkbox"/> Atribut je statički <input checked="" type="checkbox"/> Atribut je <i>enumeration</i>
--------------	--------------	---

Naziv klase: Prijava

Funkcionalni zahtjevi u kojima klasa učestvuje:

FZ br. 6 - Prikaz notifikacija

FZ br. 12 - Prijava objava

Atributi koje klasa posjeduje:

Naziv atributa	Tip varijable	Dodatne napomene
prijavaId	Integer	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
prijavljenaObjava	Objava	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
prijavljeniKomentar	Komentar	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
podnosilacPrijava	Korisnik	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
vrsta	VrstaPrijava	<input type="checkbox"/> Atribut je statički <input checked="" type="checkbox"/> Atribut je <i>enumeration</i>
opis	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>

Naziv klase: Objava

Funkcionalni zahtjevi u kojima klasa učestvuje:

FZ br. 6: Kreiranje nove objave

FZ br. 3: Prikaz objava sa preoprukom korisniku

Atributi koje klasa posjeduje:

Naziv atributa	Tip varijable	Dodatne napomene
objavaId	Integer	<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>
objavaTekst	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
objavaMedia	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
objavaTagovi	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
brojLajkova	Integer	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>

Naziv klase: Lajk

Funkcionalni zahtjevi u kojima klasa učestvuje:

FZ br. 8: Interakcija sa objavama

Atributi koje klasa posjeduje:

Naziv atributa	Tip varijable	Dodatne napomene
lajkId	Integer	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
objava	Objava	<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: Komentar

Funkcionalni zahtjevi u kojima klasa učestvuje:

FZ br. 8: Interakcija sa objavama

Atributi koje klasa posjeduje:

Naziv atributa	Tip varijable	Dodatne napomene
komentarId	Integer	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>

objava	Objava	<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>
komentarTekst	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
komentarMedia	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>

Naziv klase: Notifikacija

Funkcionalni zahtjevi u kojima klasa učestvuje:

FZ br. 7: Pregled notifikacija

Atributi koje klasa posjeduje:

Naziv atributa	Tip varijable	Dodatne napomene
notifikacijaId	Integer	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
pošiljalac	Korisnik	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
tipNotifikacije	TipNotifikacije	<input type="checkbox"/> Atribut je statički <input checked="" type="checkbox"/> Atribut je <i>enumeration</i>
objava	Objava	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
komentar	Komentar	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
prijava	Prijava	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
notifikacijaTekst	String	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
vrijemeSlanjaNotifikacije	Datetime	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>
novaNotifikacija	Bool	<input type="checkbox"/> Atribut je statički <input type="checkbox"/> Atribut je <i>enumeration</i>

SOLID PRINCIPI

1. Princip pojedinačne odgovornosti (Single responsibility Principle)

Princip je zadovoljen jer svaka klasa čuva samo njoj bitne podatke, ima samo jednu odgovornost i jedan razlog za promjenu. Nijedna klasa se ne bavi više od jednom svrhom.

2. Open-Closed Principle

Ovaj princip kaže da svaka klasa mora biti otvorena za nadogradnje i zatvorena za modifikacije. Ukoliko promijenimo jednu klasu neće doći do promjene druge klase. Ovaj princip je zadovoljen na našem dijagramu klasa, jer promjena u implementaciji bilo koje klase, ne forsira modifikaciju neke druge klase. Naprimjer, u slučaju da neko ostavi neki lajk ili komentar na neku objavu, korisniku dolazi notifikacija o izvršenoj radnji.

3. Liskov princip zamjene (Liskov Substitution Principle)

Ovaj princip zahtijeva da podklase moraju biti zamjenjive baznoj klasi, tj. da bilo koja upotreba bazne klase omogućava upotrebu i izvedenih klasa, sa istim rezultatom. Naš sistem ne sadrži nasljeđivanja, stoga je ovaj princip zadovoljen.

4. Princip izoliranja interfejsa (Interface Segregation Principle)

Klijenti ne treba da ovise o metodama koje neće upotrebljavati.

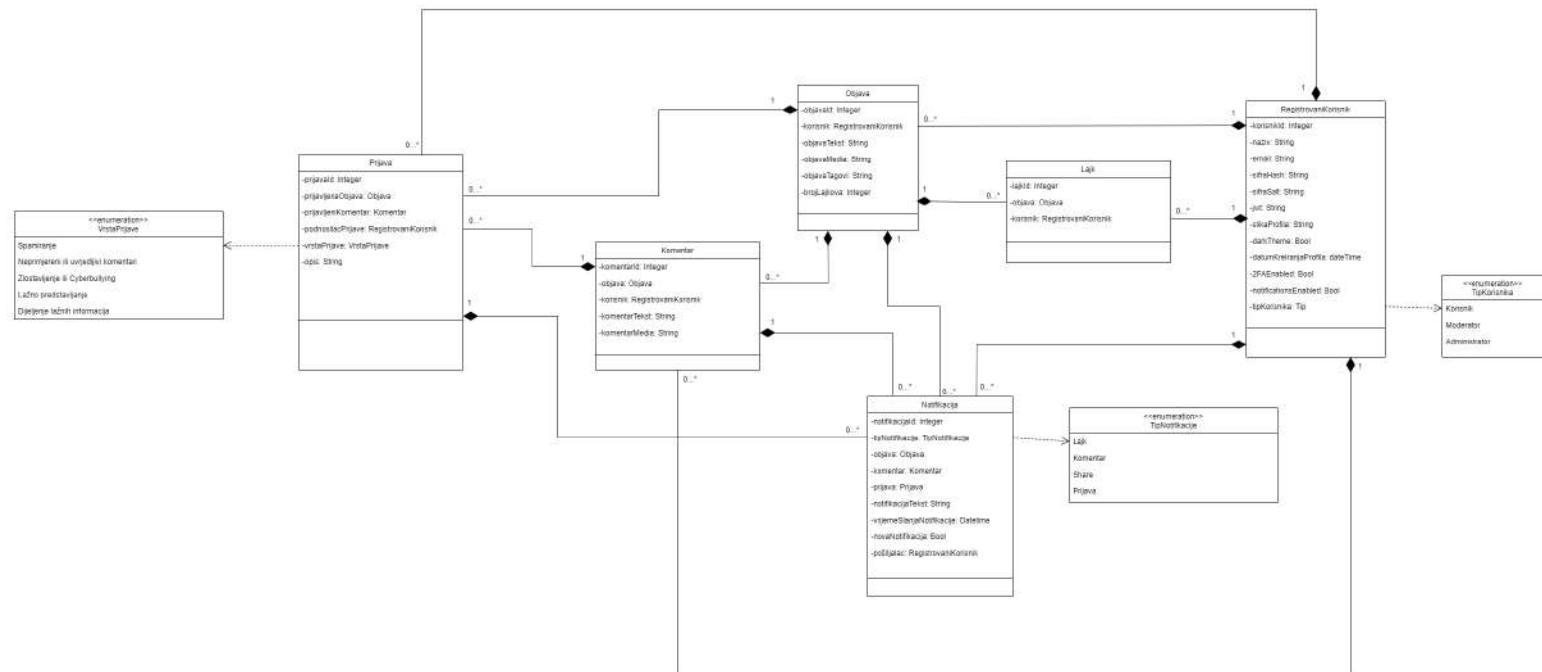
Kako smo dizajnirali model sistema, nismo koristili interfejse, te znamo da ovaj princip nije narušen.

5. Princip inverzije ovisnosti (Dependency Inversion Principle)

Ovaj princip zahtijeva da moduli visokog nivoa ne bi trebali ovisiti od modula niskog nivoa. Oba bi trebalo da ovise od apstrakcija.

B. Moduli ne bi trebali ovisiti od detalja. Detalji bi trebali biti ovisni od apstrakcija.

Kako u našem dijagramu nemamo nasljeđivanje ovaj princip je zadovoljen. Također, s obzirom da u aplikaciji nema tolike kompleksnosti da bi se nešto moglo nazvati apstrakcijom pa da zavisi od detalje i s obzirom da se ne mogu primjetiti high-level i low-level moduli, ovaj princip je zadovoljen.

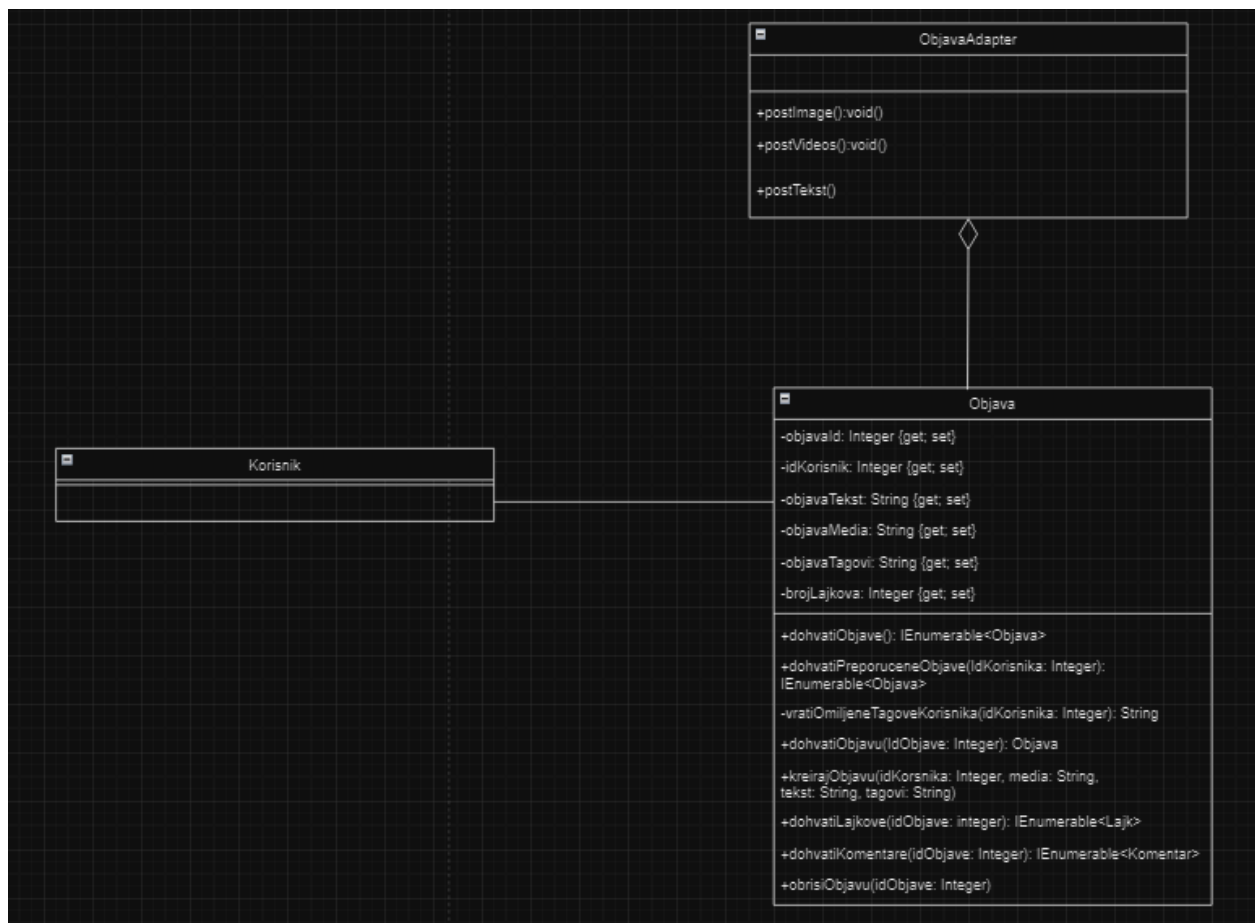


STRUKTURALNI PATERNI

Adapter patern

Osnovna namjena Adapter pattern-a je da omogući širu upotrebu već postojećih klasa. Kada je potreban drugačiji interfejs postojeće klase, a ne želimo da mijenjamo postojeću klasu koristimo Adapter pattern. Tada se kreira nova Adapter klasa koja se koristi kao posrednik između originalne klase i interfejsa. Primjena u sistemu:

U našem sistemu Adapter patern bi mogli iskoristiti za objave. U našem sistemu trenutni tip atributa za tip objavu je string. Taj tip bi mogli zamijeniti uz pomoć metoda konverzija Adapter klase i omogućiti prihvatanje raznih tipova formata objave, bila to slika, tekst ili videozapis. Adapter bi omogućio da se prilikom objavljivanja slike omogući objava iz različitih formata(jpg, pdf) i drugo.



Bridge patern

Osnovna namjena Bridge paternna je da omogući odvajanje apstrakcije i implementacije neke klase tako da ta klasa može posjedovati više različitih apstrakcija i više različitih implementacija za pojedine apstrakcije. Moguće je implementirati i sistem za razmjenu poruka primjenom Bridge paternna.

Ovaj patern bismo mogli primjeniti na tipove različitog slanja notifikacija. Naprimjer mogli bismo implementirati da korisnik može primiti notifikaciju putem aplikacije kao i putem emaila vezano o događajima, notifikacijama i drugo.

Composite patern

Composite patern opisuje grupu objekata koji se tretiraju na isti način kao pojedinačna instanca istog tipa objekta. Namjera kompozita je da "komponira" objekte u strukture stabla koje predstavljaju hijerarhiju dio-cjelina.

U našem sistemu Composite patern bismo mogli iskoristiti tako što bi mogli uvesti još jednu vrstu korisnika – gost, koja će imati sličnu funkcionalnost kao korisnik, koja bi mogla samo pregledati preporučeni sadržaj.

Decorator patern

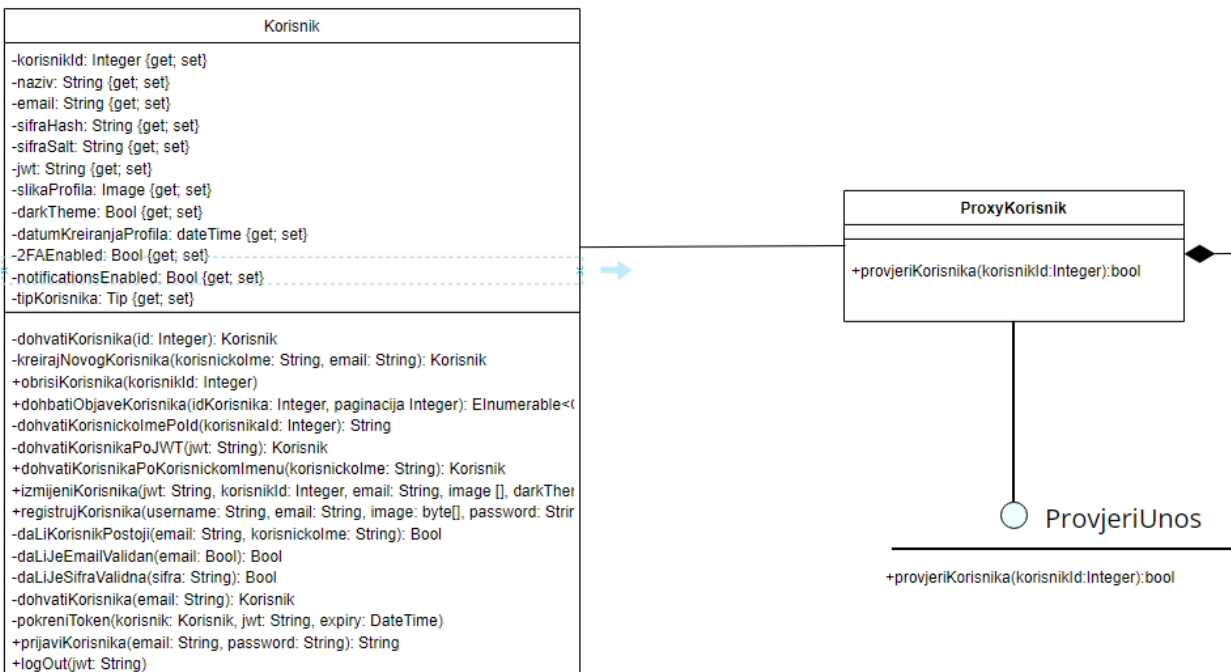
Decorator pattern služi za omogućavanje različitih nadogradnji objektima koji svi u osnovi predstavljaju jednu vrstu objekta. Umjesto da se definiše veliki broj izvedenih klasa, dovoljno je omogućiti različito dekoriranje objekata (tj. dodavanje različitih detalja), te se na taj način pojednostavljuje i rukovanje objektima klijentima, i samo implementiranje modela objekata. U suštini, decorator pattern se koristi u slučajevima kada želimo dodijeliti dodatna ponašanja objektu tokom izvođenja programa bez da mijenjamo kod koji je na neki način u interakciji sa datim objektom.

U našem sistemu decorator patern primjenili smo kao funkcionalnost koja uključuje promjenu postavki i prikaz profila korisnika. Decorator pattern može se koristiti za dodavanje ili promjenu funkcionalnosti na objektima profila korisnika, poput dodavanja mogućnosti promjene profila kao korisničkog imena, profilne slike kao i uređivanje postavki na korisničkom računu.

Proxy pattern

Svrha **Proxy patern-a** je da omogući pristup i kontrolu pristupa stvarnim objektima. Proxy je obično mali javni surogat objekat koji predstavlja kompleksni objekat čija aktivizacija se postiže na osnovu postavljenih pravila.

U našoj aplikaciji ovaj patern ćemo iskoristiti tako što ćemo zabraniti zlonamjerne upotrebe. Recimo ukoliko neki korisnik koji nije registrovan kao administrator ili zaposlenik pokuša da koristi privilegije koje ima administrator, ta akcija će biti zabranjena, te će administrator biti obavješten o tome.



Pored toga, ovaj patern se može se koristiti za optimizaciju prikaza profila korisnika. Prilikom učitavanja korisničkog profila, može se prethodno učitati informacije o korisniku poput imena i objava na profilu, dok se ne smije moći uređivati jer to nije profil koji pripada trenutnom korisniku sesije.

Također, u našem sistemu, administrator često pristupa i mijenja podatke o komentarima i objavama koji su prijavljeni, kao i brisanje korisničkog računa. Kako je autentičnost tih podataka jako bitna, trebamo sa sigurnošću znati da sve izmjene vrši administrator. U tome nam može pomoći ovaj patern.

Façade patern

Fasadni pattern služi kako bi se korisnicima pojednostavilo korištenje kompleksnih sistema, odnosno koristimo kada imamo neki sistem koji ne želimo da razumijemo kako funkcionira "ispod haube". Korisnici vide samo kranji 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.

Primjer korištenja ovog paternu u našem sistemu je primjenjen, s obzirom da korisnik vidi samo određeni dio sistema koji je njemu potreban. Primjer toga je se u pozadini kreiraju operacije kao što su stvaranje nove objave, uređivanje korisničkog profila, brisanje objava

Flyweight patern

Ovaj pattern se koristi kada kreiramo objekte samo po potrebi kada imaju različito specifično stanje, a osnovno stanje je isto za sve objekte.

Korištenjem ovog paternu se onemogućava stvaranje velikog broja instanci objekata koji u suštini predstavljaju jedan objekat. Samo ukoliko postoji potreba za kreiranjem specifičnog objekta sa jedinstvenim karakteristikama (specifično stanje), vrši se njegova instantacija, dok se u svim ostalim slučajevima koristi postojeća opća instanca objekta (bezlično stanje).

Korištenje ovog paternu je veoma korisno u slučajevima kada je potrebno vršiti uštedu memorije.

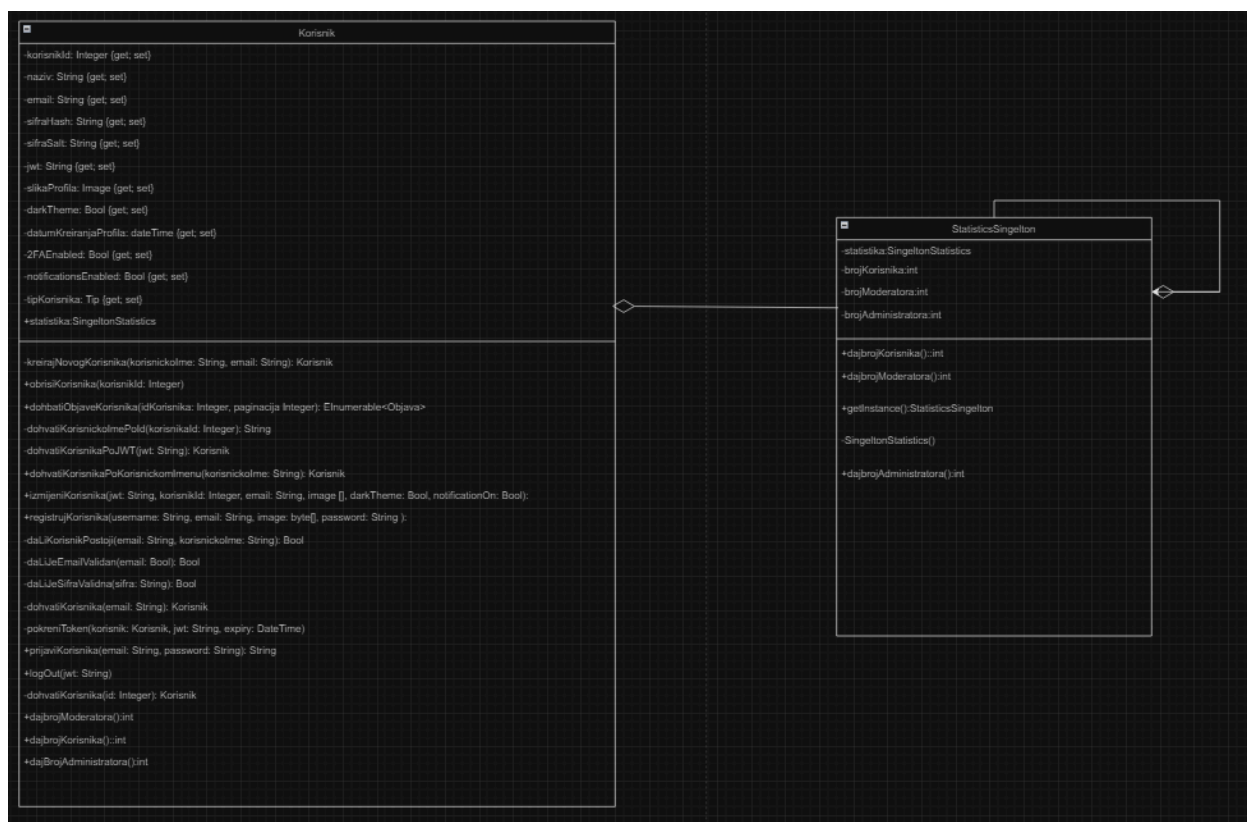
Ovaj patern smo iskoristili u našem sistemu da nakon prikaza prve tri objave, možemo učitati dodatne objave dinamički kako ih korisnik pregledava. Umjesto da odmah učitate sve objave na stranici, učitavanje dodatnih objava može se pokrenuti kada korisnik dosegne određeni dio stranice ili pritisne gumb za učitavanje više objava. Kada se objave dinamički učitavaju, preglednik može bolje upravljati memorijom jer ne mora zadržavati veliku količinu podataka u memoriji odjednom. To može poboljšati performanse aplikacije i spriječiti preopterećenje preglednika.

KREACIJSKI PATERNI

Singleton šablon

Singleton šablon osigurava a da se klasa može instancirati samo jednom i da osigura globalni pristup kreiranoj instanci klase. Postoje objekti čija je samo jedna instanca potrebna i nad kojima je potrebna jedinstvena kontrola pristupa. Instanciranje više nego jednom može prouzrokovati probleme kao što su nekorektno ponašanje programa, neadekvatno korištenje resursa ili nekonzistentan rezultat.

U našem sistemu trebali bismo kreirati klasu SingletonStatistics, koja bi trebala vraćati statistiku za korisnički profil.



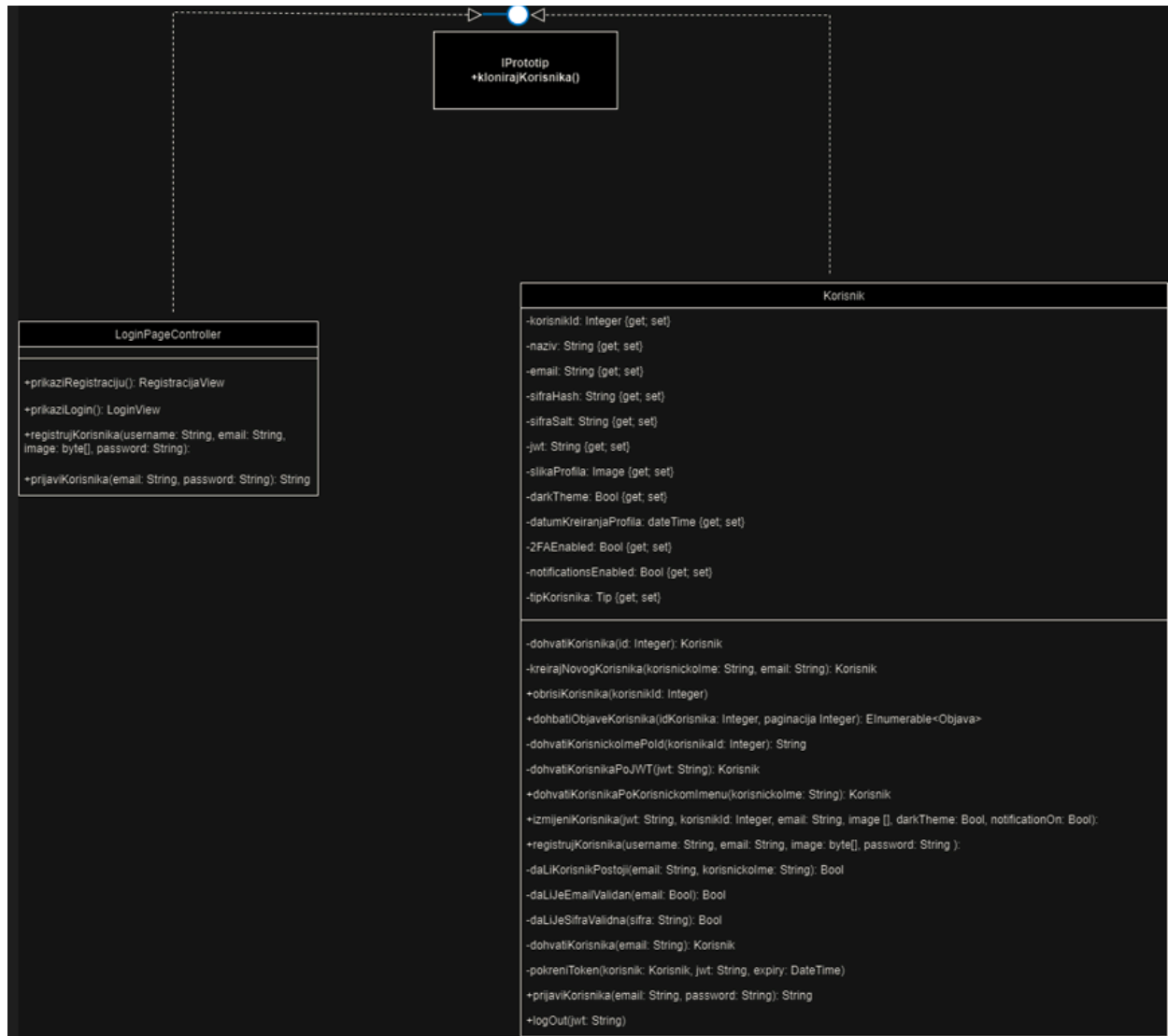
Također, Singleton patern bi mogli primjeniti pri registraciji korisnika na sistem. Mogli bi iskoristiti na način da se bilježe pristupi sistemu korisnika. Napravili bi novu klasu npr. `LoggerClass` koja bi bila singleton klasa. Potrebno je kreirati jednu instancu da se ne bi otvaralo više log fajlova. Sve klase će imati pristup instanci `LoggerClass` klase i dobijati željene informacije.

LoggerClass
-instance: LoggerClass()
+getInstance() -LoggerClass()

Također, Singleton pattern bi se u našem projektu mogao iskoristiti ako bi se odlučili implementovati neku vrstu globalnog tajmera koji bi se ponašao kao logger. Biljezio bi koliko je korisnik bio aktivan na aplikaciji u toku dana...Pošto se vrijeme računa od registracije korisnika, nema smisla da ovih tajmera bude više, već samo jedan

Prototype patern

Osnovna funkcija ovog patern je da olakša kreiranje objekata koristeći postojeću instancu, koja se ponaša kao prototip. Novokreiranom objektu možemo promijeniti određene osobine po potrebi. U našem sistemu, ovaj patern bi mogli iskoristiti nad klasom Korisnik. Ukoliko postoje dvije ili više osoba sa istim imenom i prezimenom, umjesto kreiranja novog objekta, možemo klonirati prvu instancu i izmijeniti odgovarajuće podatke koji se razlikuju (email, username, password). Kreirali bi interfejs IPrototip sa metodom kloniraKorisnikaj koja implementira način kloniranja objekta.



Factory Method patern

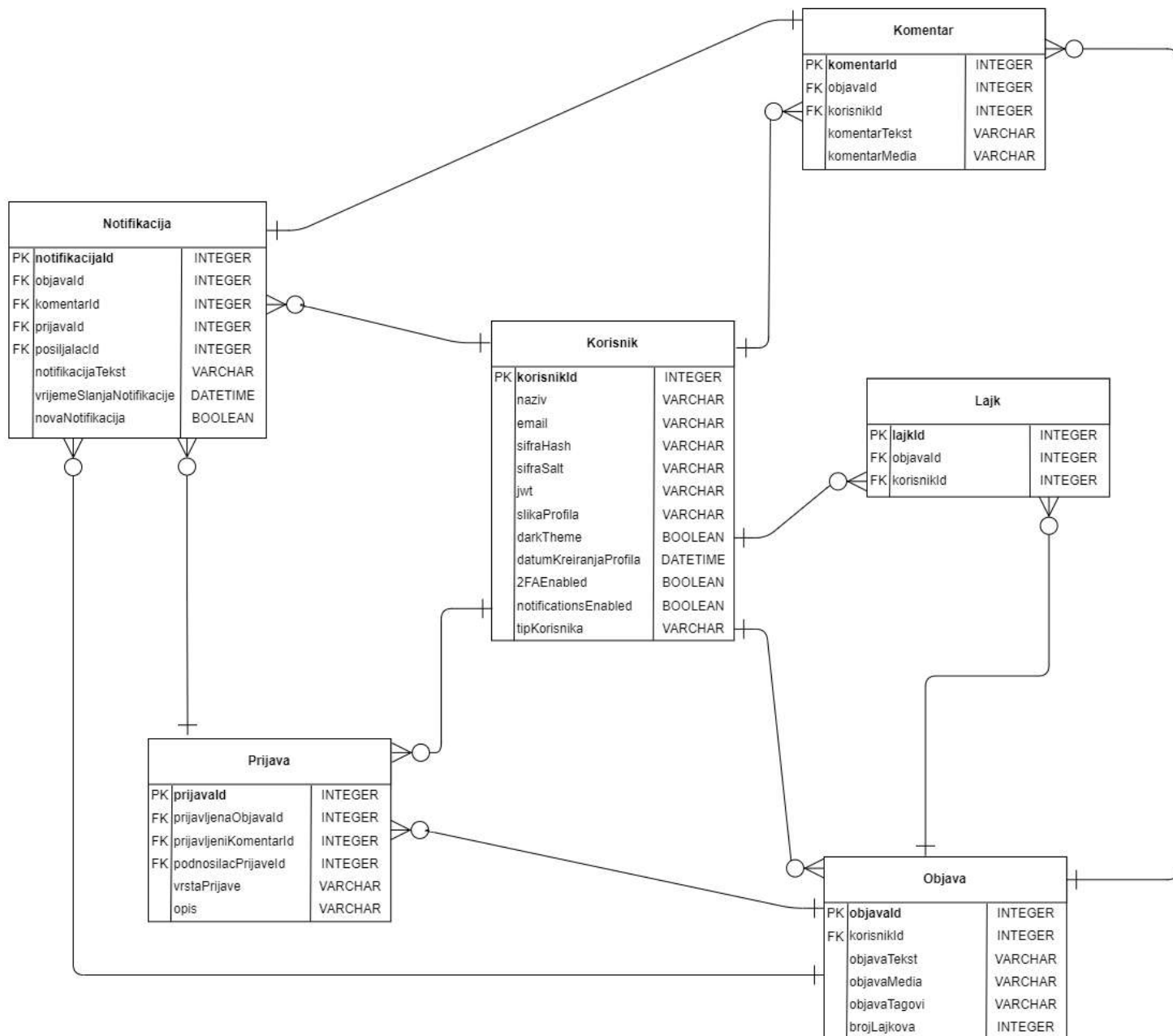
Uloga Factory Method patern je da omogući kreiranje objekata na način da podklase odluče koju klasu instancirati. Različite podklase mogu na različite načine implementirati interfejs. Factory Method patern instancira odgovarajuću podklasu (izvedenu klasu) preko posebne metode na osnovu informacije od strane klijenta ili na osnovu tekućeg stanja. U našem sistemu ovaj patern možemo iskoristiti kod registracije korisnika, tj. instanciramo objekte različitih tipova (Korisnik, Zaposlenik, Administrator). To možemo uraditi na način da kreiramo interfejs IRegistracija, nakon toga klase RegistracijaKorisnik i RegistracijaZaposlenik, RegistracijaAdministrator koje će implementirati interfejs.

Builder patern

Ovaj patern bismo mogli iskoristiti kod kreiranja objekata tipa Korisnik jer klasa Korisnik sadrži veći broj atributa koji bi se mogli odvojeno definisati, ali idalje ne prevelik broj atributa tako da nije kompleksno ni naše postojeće rješenje. No to bi se moglo promijeniti ukoliko se klasa Korisnik nadogradi sa mnogo više informacija o korisniku i u tom slučaju bi ovaj patern bio itekako od koristi. U slučaju implementacije Builder patern, postojao bi interfejs IBuilder sa različitim dijelovima kreiranja korisnika, dodajMjestoRodjenja(String drzava, String mjestoRodjenja), dodajDatumRodjenja(int datumRodjenja) itd. Također postojali bi i različiti builderi - BuilderUser bi pozvao samo osnovne metode, dok bi BuilderUseBiznis pozvao i dodatne metode koje bi bile u skladu sa mogućnostima tog korisnika (trenutno takav korisnik nije predviđen, no za potrebe dodavanja ovog patern u naš sistem, ta funkcionalnost bi se mogla uvesti).

Abstract Factory patern

Abstract Factory patern omogućava da se kreiraju familije povezanih objekata/produkata. Na osnovu apstraktne familije produkata kreiraju se konkretne fabrike (factories) produkata različitih tipova i različitih kombinacija. Patern odvaja definiciju (klase) produkata od klijenta. Zbog toga se familije produkata mogu jednostavno izmjenjivati ili ažurirati bez narušavanja strukture klijenta. Ovaj patern bi bio iskoristiv kada bismo željeli ubaciti filtere za objavu(najpopularnije, filteri po tipu sadržaja, vremenski filteri ...). Bilo bi potrebno kreirati interfejs IFactory koji bi na jednostavniji način omogućio dobivanje instance Objava na osnovu datih filtera.

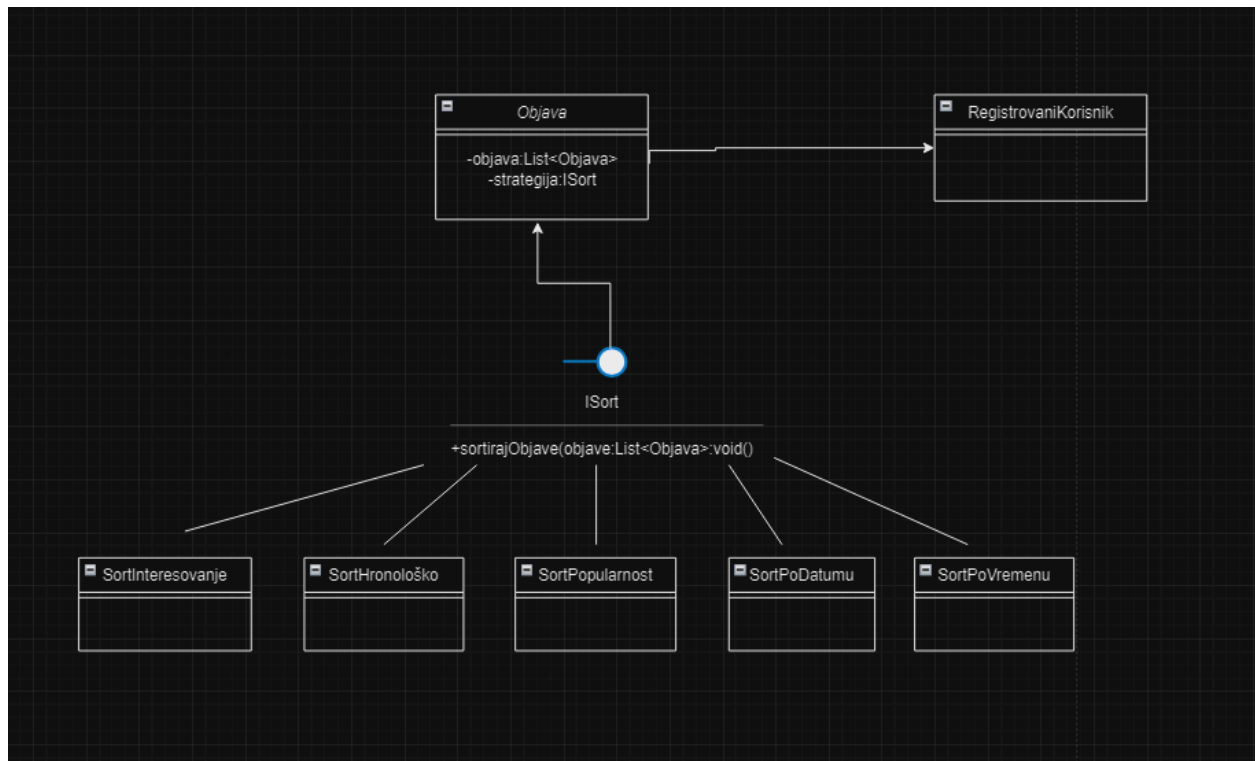


Paterni ponašanja

Strategy patern

Strategy patern izdvaja algoritam iz matične klase i uključuje ga u posebne klase. Pogodan je kada postoje različiti primjenjivi algoritmi (strategije) za neki problem. Strategy patern omogućava klijentu izbor jednog od algoritma iz familije algoritama za korištenje.

U našem sistemu bismo Strategy patern mogli iskoristiti tako da prethodno RegistrovanomKorisniku omogućimo da sortira objave. Objave bi se mogle sortirati na više načina (hronološko sortiranje, sortiranje prema popularnosti ili prilagođeno sortiranje bazirano na interesovanjima korisnika). Conext klasa bi za atribut imala listu objava i strategiju sortiranja.



Također, ovaj patern možemo dodati u naš sistem na sljedeći način. Poznato nam je da unutar našeg sistema imamo 3 vrste korisnika: admin-a, modeatora, te samog klijenta. Stoga prilikom samog prijavljivanja korisnika, imamo sigurno 3 opcije prilikom registracije. Detekcija o kojem od tri nabrojana korisnika se radi, vrši se na osnovu ulaznih podataka. Na osnovu datog tipa klase tj. korisnika mogli bismo realizovati 'setup' nakon registracije u kojem bi za svaki tip korisnika postojao drugačiji prikaz (znamo da ova tri korisnika mogu pristupiti i vidjeti različite stvari), samim tim možemo i drugačije podatke dobiti pomoću api request-a prema našoj bazi.

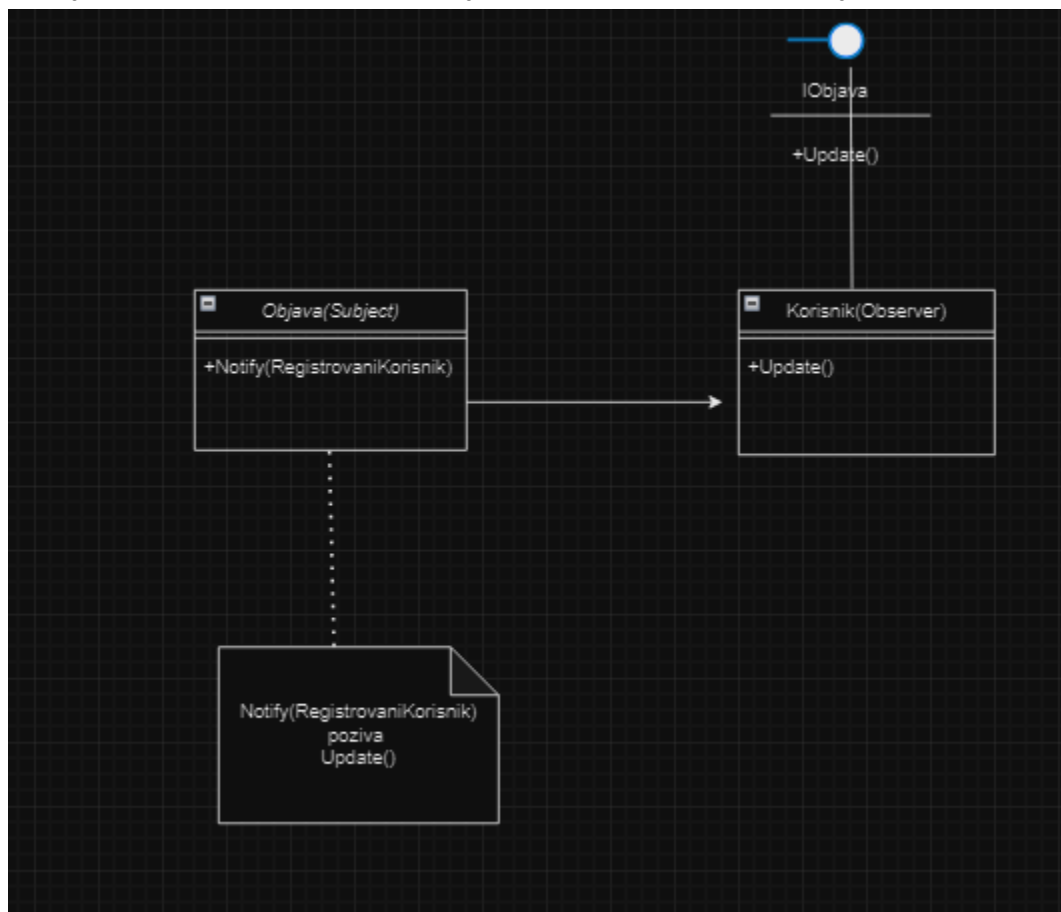
Observer patern

Uloga Observer patern je da uspostavi relaciju između objekata tako kada jedan objekat promijeni stanje drugi zainteresirani objekti se obavještavaju.

Ovaj patern smo primjenili u našem sistemu tako što korisnik dobije notifikaciju (obavještenje) kada drugi korisnik ima neku interakciju sa njegovim profilom.

Također, ovaj patern možemo dodati u naš sistem na sljedeći način. Poznato je da sistem posjeduje funkcionalnost koja omogućava klijentu da ažurira svoj profil. Kada korisnik objavi novi post ili komentar, svi korisnici bivaju obavješteni i njihov feed se ažurira.

Također, obzirom da ćemo observer patern dodati na način da ćemo omogućiti administratoru ili zaposleniku da svakoga puta kada bilo koji klijent ažurira svoj korisnički profil, da on dobije obavijest o tome. Tako će biti u stanju da vidi ažurirane informacije.



State patern

State patern je dinamička verzija Strategy patern i omogućava promjenu načina ponašanja objekta u zavisnosti od stanja u kom se nalazi. Za razliku od Strategy patern, objekat sam mijenja svoje stanje, ono nije izabrano od strane klijenta.

Recimo da želimo omogućiti korisnicima da žele svoj profil učiniti privatnim. Tada bi klasa Korisnik imala dva mode-a: privatni i javni, odnosno ovo bi bile klase koje implementiraju

interface IMode. U zavisnosti od klase, odnosno mode-a bi omogućili drugim korisnicima prikaz profila tog korisnika, odnosno poziv metode prikaziProfil klase PrijavljeniKorisnikController.

Iterator patern

Iterator patern omogućava sekvencijalni pristup elementima kolekcije bez poznavanja kako je kolekcija struktuirana. U našem sistemu ovaj patern možemo primijeniti tako što ćemo omogućiti korisnicima da biraju redoslijed prikaza korisnika koji su im lajkali fotografiju ili koji su komentarisali njihovu objavu.

Medijator patern

Medijator patern enkapsulira protokol za komunikaciju među objektima dozvoljavajući da objekti komuniciraju bez međusobnog poznavanja interne strukture objekta.

Ovaj patern možemo koristiti u našoj aplikaciji, ali možemo proširiti sistem da imamo više vrsta registrovanih korisnika. Mogli bi dodati Premium i VIP korisnike u naš sistem. Korisnici da bi stupili u kontakt mogu koristiti neki vid chat platforme. U našem slučaju bi bez ograničenja komunicirati Premium i VIP korisnici dok standardni korisnici ne bi imali sve mogućnosti.

Chain of responsibility patern

Chain of responsibility omogućava razdvajanje jednog kompleksnog procesa obrade na više objekata koji na različite načine procesiraju primljene podatke. U našem sistemu bismo mogli primijeniti ovaj patern kada bismo omogućili korisnicima da Chain of responsibility da biraju da li žele prihvatiti zahtjev za praćenje.

Template method patern

Omogućava izdvajanje određenih koraka algoritma u odvojene podklase. Struktura algoritma se ne mijenja - mali dijelovi operacija se izdvajaju i ti se dijelovi mogu implementirati različito.

U našem sistemu ovaj patern možemo primijeniti na način da omogućimo primjenu više algoritama pri pretraživanju korisnika. Tako bismo omogućili da se pretraženi korisnici sortiraju na različite načine (dob, broj followera...)

