

*Izjavljujem da tijekom izrade ove zadaće neću od drugoga primiti niti drugome pružiti pomoć, te da se neću koristiti nedopuštenim sredstvima. Ove su radnje teška povreda Kodeksa ponašanja te mogu uzrokovati i trajno isključenje s Fakulteta. Također izjavljujem da mi zdravstveno stanje dozvoljava pisanje ove zadaće.*

---

JMBAG


---

Ime i prezime

---

Vlastoručni potpis

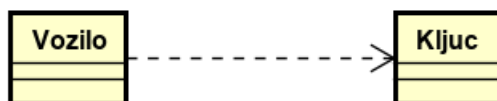
### GRUPA A

1. (1 bod) Navedite barem tri značajke dobrog programskog proizvoda. mora osigurati traženu funkcionalnost i performanse, prihvatljivost za korisnika, pouzdanost, mogućnost lakog održavanja.
2. (1 bod) Navedite barem tri primjera dionika: kupac, klijent, rukovoditelj razvoja, razvojni inženjer.
3. (1 bod) Navedite sve načine zapisivanja zahtjeva sustava. strukturiranim prirodnim jezikom, posebnim jezicima za oblikovanje sustava, dijagramima i matematičkom notacijom.
4. (1 bod) Prilikom utvrđivanja prioriteta zahtjeva, zahtjevi koji se analiziraju u obliku naprednih mogućnosti i koji su pogodni za prikaz mogućnosti budućeg razvoja nazivaju se zahtjevima niskog prioriteta.
5. (1 bod) Veza koja može biti između dva aktera na UML dijagramima obrazaca uporabe naziva se generalizacija.
6. (1 bod) Nacrtajte poruku za uništavanje objekta na UML sekvencijskim dijagramima.  

7. (1 bod) Kod unificiranog procesa razvoja programske potpore, jezgrena aktivnost ispitivanja (engl. *test*) programske potpore provodi se u više faza, ali u najvećoj mjeri tijekom faze izgradnje (engl. construction).
8. (1 bod) Ekstremno programiranje i čisti (engl. *lean*) razvoj neki su od primjera agilnih metoda procesa razvoja programske potpore.
9. (1 bod) Skup stavaka iz projektnog dnevnika zaostataka koji je određen za implementaciju u nekom sprintu naziva se dnevnik sprinta (ili sprint dnevnik) (engl. sprint backlog).

10. (1 bod) Kod klasifikacije modela arhitekture, dinamički procesni model (engl. *component and-connector view type*) prikazuje komponente u izvođenju (engl. runtime).
11. (1 bod) Princip dobrog oblikovanja programske potpore koji predviđa ispunjavanje preduvjeta, završnih uvjeta i invarijanti (engl. *invariants*) prilikom pozivanja neke metode naziva se Oblikuj po ugovoru (engl. Design by Contract).
12. (1 bod) Nacrtajte dijagram razreda za sljedeći kod:

```
public class Kljuc {
    ...
}
public class Vozilo{
    public void otključaj(Kljuc k){
        k.ocitaj();
    }
}
```

Rj.



13. (1 bod) Trenutno se nalazimo na grani master. Osim nje, postoji i grana develop. Obje grane sadrže istu datoteku „popis namirnica.txt“ različitog sadržaja (prikazano na donjoj slici). Napišite kako će izgledati datoteka na trenutnoj grani ukoliko pokrenemo naredbu „git merge develop“.

popis namirnica.txt	popis namirnica.txt
1 Jabuka	1 Jabuka
2 Banana	2 Banana
3 Mandarina	3 Klementina
4 Kruška	4 Kruška
5 Šljiva	5 Šljiva

grana master

grana develop

Rj.

Jabuka

Banana

<<<<<< HEAD

Mandarina

=====

Klementina

>>>>>> develop

Kruška

**Problemski dio - Web aplikacija za neprofitne udruge**

Potrebno je razviti web aplikaciju koju će koristiti neprofitne udruge za organizaciju radionica, dobrotvornih akcija i druženja. Glavna ideja aplikacije je omogućavanje suradnje članova udruge pri organizaciji događaja te prezentacija aktualnosti zainteresiranoj javnosti.

Unutar aplikacije svaka udruga treba stvoriti svoj profil. Registraciju udruge u aplikaciji obavlja njen voditelj i pri tome unosi osnovne podatke o udruzi (naziv, adresa, kontakt telefon i adresa e-pošte) te svoje osobne podatke (OIB, ime, prezime, adresa e-pošte). Na taj način istovremeno se stvara profil udruge i svoj osobni korisnički račun sa statusom "voditelj". Voditelj jedini ima pravo naknadno uređivati profil udruge. Svi ostali članovi udruge također se mogu registrirati na profilu udruge unosom svojih osobnih podataka (OIB, ime, prezime, adresa e-pošte). Njihovu registraciju potvrđuje voditelj unutar aplikacije i dobivaju status "redovni član". Voditelj može redovnim članovima udruge dodijeliti status "administrator", može prenijeti status voditelja na redovnog člana ili administratora te može ukloniti postojeće članove.

Na profilu udruge javno je dostupan popis događaja u zadnjih godinu dana te za svaki događaj bilo tko može postavljati komentare. Administratori i voditelj mogu stvarati nove događaje. Vrste događaja su: radionica, dobrotvorna akcija i druženje. Za svaki događaj su navedeni: naziv, vrsta i kratki opis. Za radionicu i druženje zadaje se lokacija i vrijeme održavanja. Lokaciju je moguće vidjeti na karti, a za prikaz karte aplikacija koristi vanjski servis *GeoKarte*. Za dobrotvornu akciju navodi se što se sve prikuplja (novac, roba, ostalo), razdoblje prikupljanja donacija te broj računa za uplatu novčanih donacija. Svi podaci o događajima su javno vidljivi.

Svi članovi udruge se mogu prijavljivati za sudjelovanje u organizaciji događaja i to sa statusom: „sudjelujem“, „možda“, „ne sudjelujem“. Članovi se mogu i predomisлити pa promijeniti interes. Svim članovima je za svaki događaj vidljiv popis članova koji su postavili svoj status.

*NAPOMENA: Prilikom izrade pojedinih dijagrama uz opći opis uzmite u obzir te poštujujte i sljedeće eksplicitno navedene zahtjeve uz pojedine zadatke.*

**14. (4 boda) Dijagram obrazaca uporabe**

UML dijagramom obrazaca uporabe modelirajte aplikaciju za neprofitne udruge.

Rj.



- prikaz karte
- navodi se što se sve prikuplja
- podaci o događajima su javno vidljivi
- prijavljivati za sudjelovanje
- predomisliti pa promijeniti interes
- vidljiv popis članova

Temeljem toga utvrđeni su obrasci uporabe (engl. *use case*) koji opisuju sekvence akcija (uključujući varijante) koje sustav obavlja u interakciji s aktorima.

Neke od akcija mogu se, ali i ne moraju prikazati kao zaseban obrazac uporabe:

- Prikaz karte

### **3. Povezivanja aktora s obrascima uporabe i određivanje aktivnih i pasivnih aktora.**

- Pasivni aktor je GeoKarta ,
- Aktivni aktori su javni korisnik, član udruge, administrator i voditelj.

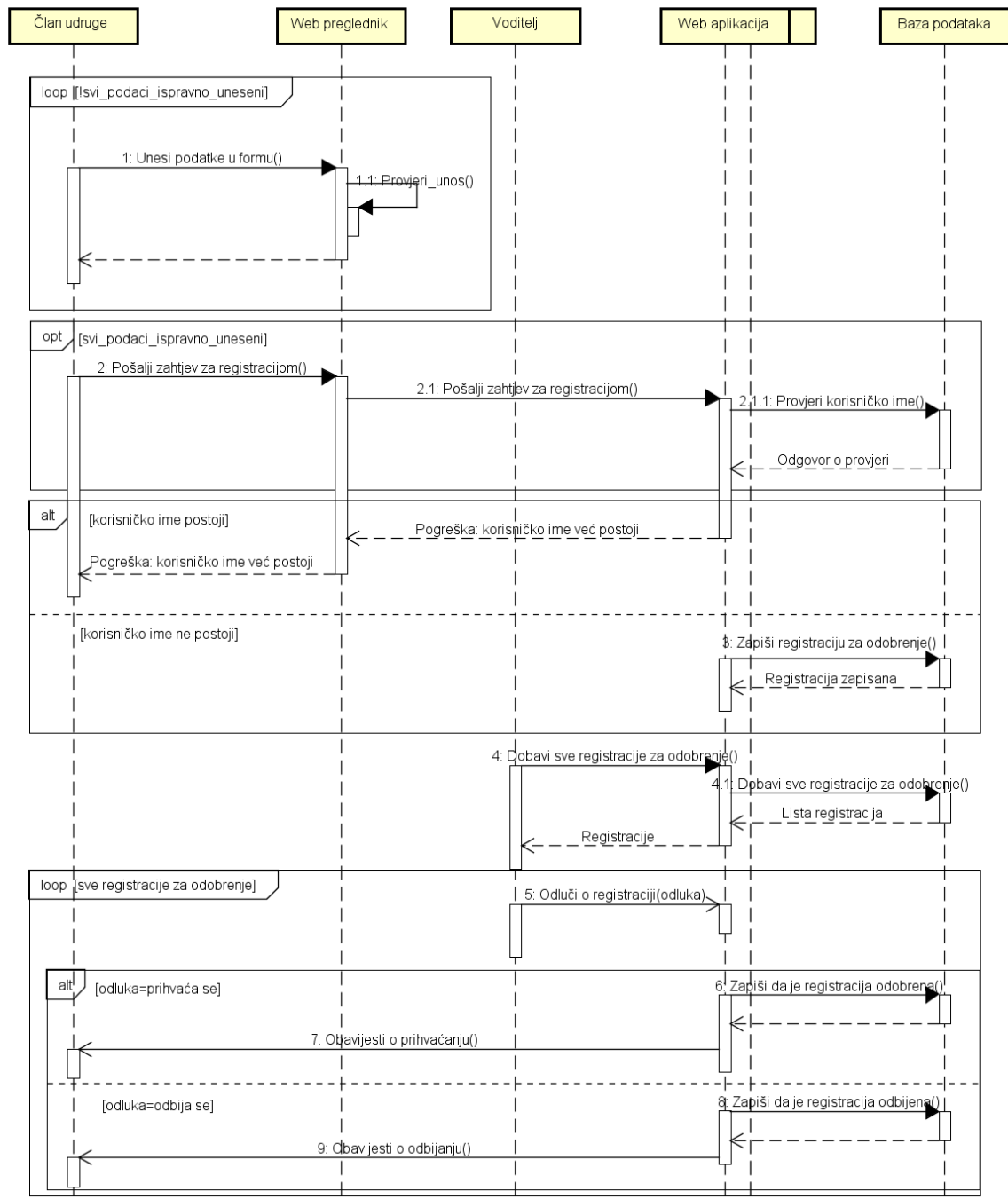
Voditelj je specifikacija administratora, a administrator je specifikacija člana udruge. Moguće je crtati i da je član udruge specifikacija javnog korisnika.

### **4. Prikazati granice sustava i imenovati sustav.**

## **15. (4 boda) Sekvencijski dijagram**

Sekvencijskim UML-dijagramom prikažite proces registracije člana udruge (koji nije voditelj). Pri tome uzmite u obzir i sljedeće: prilikom popunjavanja forme za registraciju, web preglednik provjerava jesu li sva polja ispravno popunjena te dozvoljava slanje zahtjeva za registraciju tek nakon što je sve ispravno popunjeno. Kad se zahtjev pošalje, aplikacija u bazi podataka provjerava postoji li već neki korisnik s istim korisničkim imenom i tek ako ne postoji, šalje na potvrdu voditelju. Voditelj dobavlja listu registracija za odobrenje te ih redom razmatra: registraciju može potvrditi ili odbiti. O pozitivnim i negativnim odlukama, web aplikacija obavještava člana udruge.

Rj.

**Bodovanje:****Aktori (1 bod)**

- ne prepoznavanje aplikacije kao aktora: -1 bod
- ne prepoznavanje nekog drugog aktora: -0,5 boda
- ne prepoznavanje dva ili više aktora (ako nisu aplikacija): -0,75 boda

**Rješavanje zadatka:**

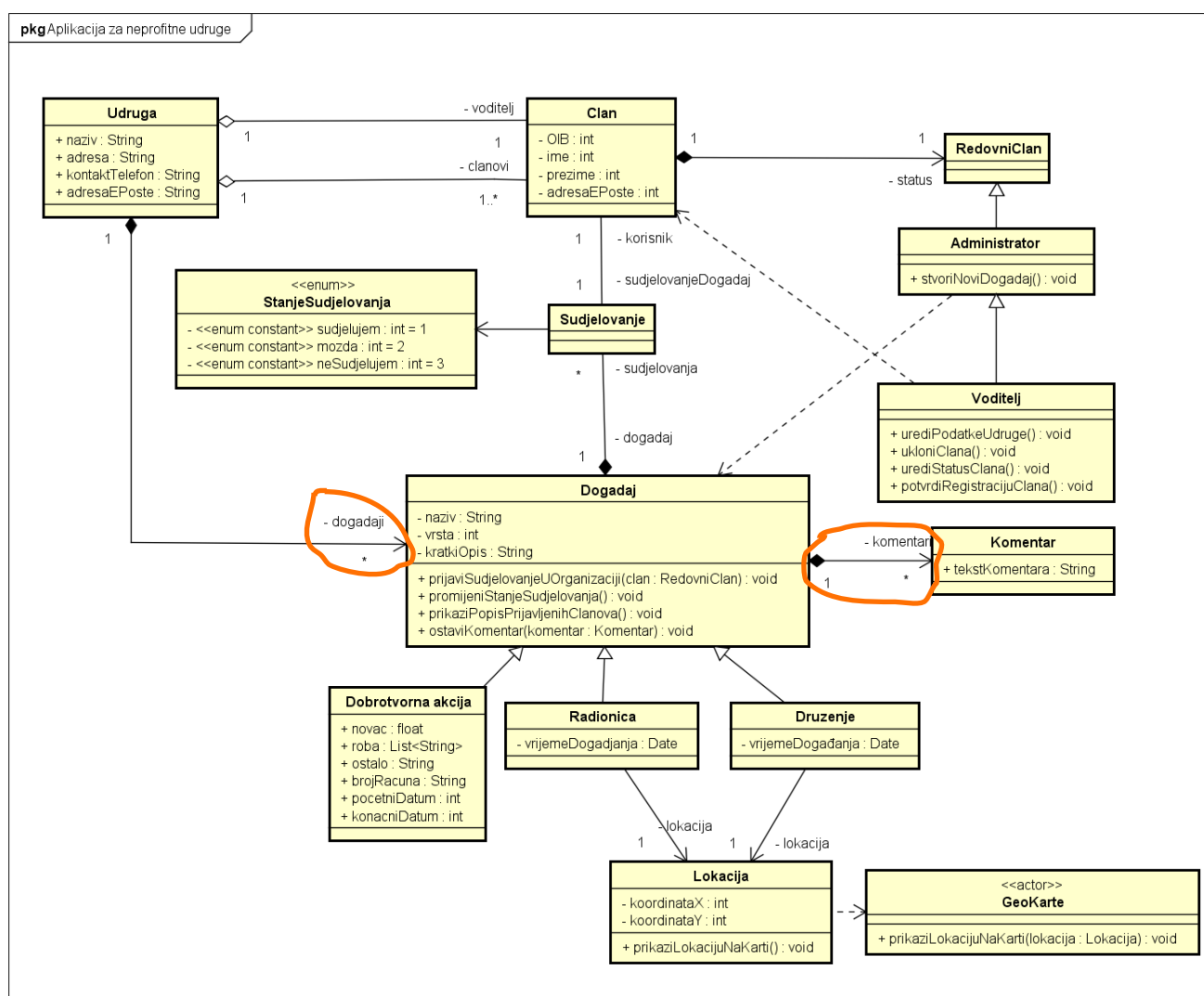
- Pridržavanje osnova grafičkog prikaza UMLa: 0,5 boda
- Provjera ispravnosti podataka kod unosa: 0,5
- Provjera postojanja korisnika u bazi podataka: 0,5 boda
- Postupak odobravanje korisnika s liste od strane voditelja: 0,5 boda

- **Petlja (loop) za odobravanje svih članova na listi voditelja: 0,25 boda**
- **Upisivanje podatka o registraciji nakon odluke voditelja: 0,25 boda**
- **Ispravan i logičan slijed radnji sukladno zadatku: 0,25 boda**
- **Označavanje uvjeta u grananju: 0,25 boda**

## 16. (4 boda) Dijagram razreda

UML dijagramom razreda modelirajte aplikaciju za neprofitne udruge.

Rj.



## Bodovanje

- Dobro identificirani razredi – 1.25 bodova
- Veze između razreda – 1.25 bodova
- atributi i metode – 1 bod
- višestrukost – 0.5 bodova

Postupak rješavanja dijagrama razreda:

### Identificiranje razreda i odnosa među razredima:

Uočiti bitne imenice iz teksta zadatka koje opisuju aplikaciju:

- udruga, član, događaj -> događaji su vezani uz udrugu kompozicijom budući da ako se briše udruga nema smisla zadržati događaje; članovi mogu biti vezani uz udrugu agregacijom ako se smatra da je u budućnosti moguće proširenje aplikacije u kojem jedna osoba može biti član više udruga
- status članstva: redovni član, administrator i voditelj -> moguća promjena vrste članstva -> modelirati kao atribut razreda član

\* Priznaje se i rješenje u kojem razred Clan nema atribut status u kojem je hijerarhija statusa nego razredi Admin i Voditelj direktno nasljeđuju razred Clan

- vrste događaja: dobrotvorna akcija, radionica, druženje -> hijerarhija razreda
- sudjelovanje člana u događaju sa statusom -> sudjelovanje modelirati kao zaseban razred s atributom status u kojem je enumeracija StanjeSudjelovanja
- komentari vezani uz svaki događaj
- lokacija događaja - mora biti zaseban razred a ne String budući da se sastoji od min 2 atributa (koordinate)
- ovisnost o vanjskom servisu GeoKarte za prikaz karata (priznaje se i rješenje bez ovog razreda)

### Određivanje višestrukosti pridruživanja

- na svakoj vezi pridruživanja potrebno je to označiti broj mogućih objekata.

### Definiranje operacija i atributa za pojedine razrede

- razred Događaj ostvaruje sve odgovornosti vezane uz događaj (iščitati iz dijagrama obrazaca uporabe).
- razredi u hijerarhiji status članstva ostvaruju odgovornosti vezane uz akcije običnog člana, administratora i voditelja (iščitati iz dijagrama obrazaca uporabe).
- nužni atributi su osnovni podaci o članu i udruzi (zadani u zadatku), naziv događaja i vrijeme održavanja
- načelno, sve metode su javne, a svi atributi privatni.
- privatnim atributima pristupa se preko get i set metoda, što nije potrebno prikazivati, kao niti konstruktore.