

| Uposlenik     |
|---------------|
| Id            |
| ime           |
| prezime       |
| korisnickolme |
| sifra         |

| Vlasnik       |
|---------------|
| Id            |
| ime           |
| prezime       |
| korisnickolme |
| sifra         |

| Snowboard |
|-----------|
| Id        |
| duzina    |

| SnowboardCipele |
|-----------------|
| Id              |
| velicina        |

| Kaciga   |
|----------|
| Id       |
| velicina |

| Pancerice |
|-----------|
| Id        |
| velicina  |

| Oprema    |
|-----------|
| Id        |
| cijena    |
| marka     |
| materijal |

| Skije  |
|--------|
| Id     |
| duzina |
| srina  |

| Stapovi |
|---------|
| Id      |
| duzina  |

| TipoviZahtjeva |
|----------------|
| Id             |
| Zahtjevid      |
| TipZahtjevald  |

| StavkaZahtjeva |
|----------------|
| Id             |
| Zahtjevid      |
| Opremaild      |

| TipZahtjeva |
|-------------|
| ID          |
| naziv       |
| cijena      |

| Zahtjev                 |
|-------------------------|
| Id                      |
| datumPodnosenjaZahtjeva |
| datumIzdavanjaUsluge    |
| datumZavrsetkaUsluge    |
| klijentId               |
| cijena                  |
| popust                  |
| placeno                 |

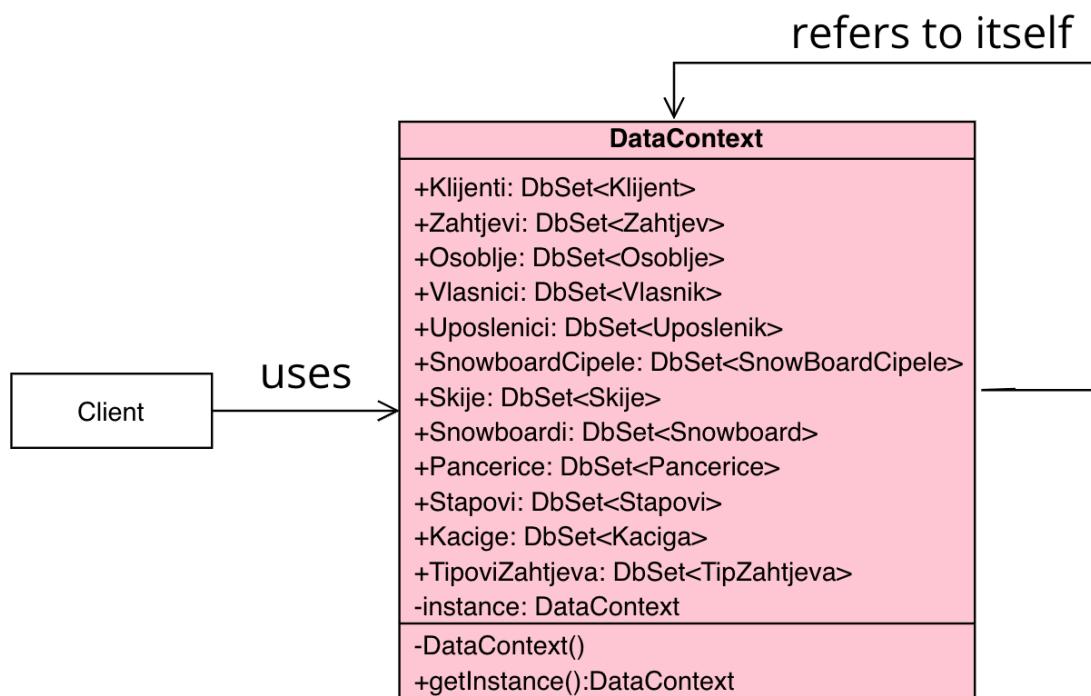
| Klijent      |
|--------------|
| Id           |
| ime          |
| prezime      |
| brojTelefona |
| email        |
| visina       |
| nivoJestine  |

# Kreacijski design patterni

Za implementaciju odabrani su Singleton i Prototype design patterni.

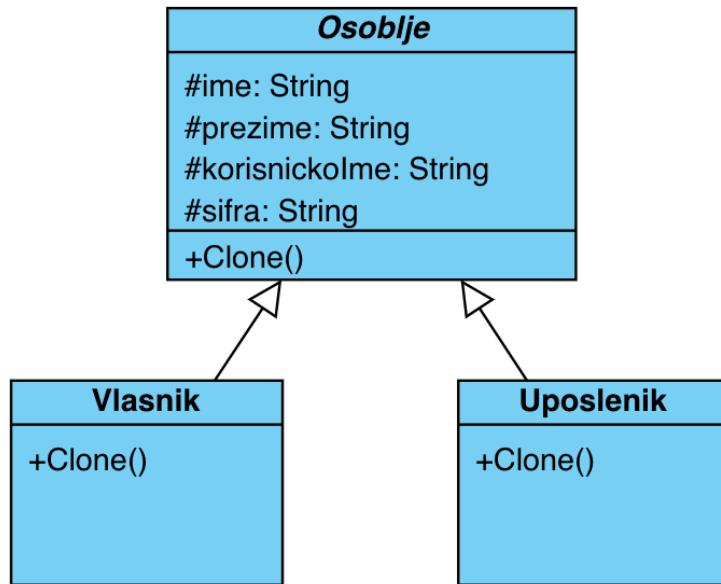
## 1. Singleton

Uloga Singleton patterna je da osigura da se klasa može instancirati samo jednom i da osigura globalni pristup kreiranoj instanci klase. U našem sistemu bismo mogli iskoristiti ovaj pattern pri pristupu bazi podataka, kako bi riješili problem mogućeg istovremenog pristupa od strane vlasnika i uposlenika. Klasa koja bi nam omogućila pristup bazi bi bila singleton.



## 2. Prototype

Uloga Prototype paterna je da kreira nove objekte klonirajući jednu od postojećih prototip instanci. Ako je trošak kreiranja novog objekta velik i kreiranje objekta je resursno zahtjevno tada se vrši kloniranje već postojećeg objekata. U našem sistemu ovaj pattern možemo iskoristiti nad klasom Osoblje. Mogli bismo iskoristiti prvu instancu klase, klonirati je a zatim promijeniti podatke koji se razlikuju. Klase implementiraju IPrototype interfejs i metodu Clone.



### 3. Builder

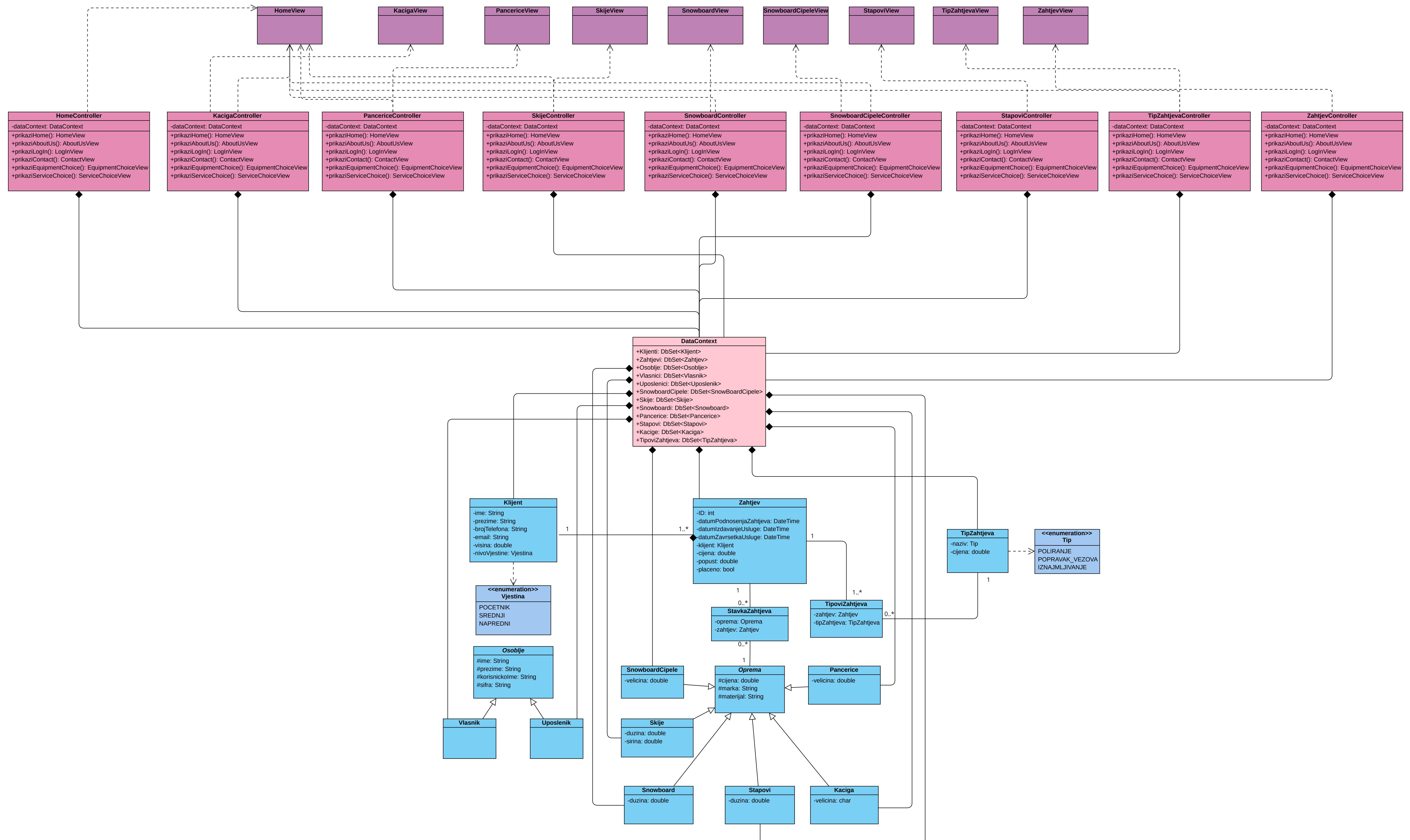
Uloga Builder paterna je odvajanje specifikacije kompleksnih objekata od njihove stvarne konstrukcije. Isti konstrukcijski proces može kreirati različite reprezentacije. U našem sistemu možemo ovaj pattern primijeniti za konstrukciju klase Zahtjev. Napravili bi „Direktor“ klasu koja bi definirala redoslijed pozivanja koraka konstrukcije. Također, trebali bi dodati builder interfejs koji bi sadržavao metode za dodavanje potrebnih informacija i builder klase koje mi implementirale ovaj interfejs.

### 4. Abstract factory

Abstract Factory pattern omogućava da se kreiraju familije povezanih objekata/produkata. Ukoliko postoji više tipova istih objekata te različite klase koriste različite podtipove, te klase postaju fabrike za kreiranje objekata zadanoj podtipa bez potrebe za specificiranjem pojedinačnih objekata. Na ovaj način se, korištenjem nasljeđivanja, ukida potreba za postojanjem if-else uslova jer određeni tip fabrike sadrži određene tipove objekata i zna se tačno koju podklasu će instancirati. U našem sistemu ovaj pattern moguće je iskoristiti za klasu Oprema gdje bi to bila Abstract Factory a izvedene klase bi bile Concrete Factories.

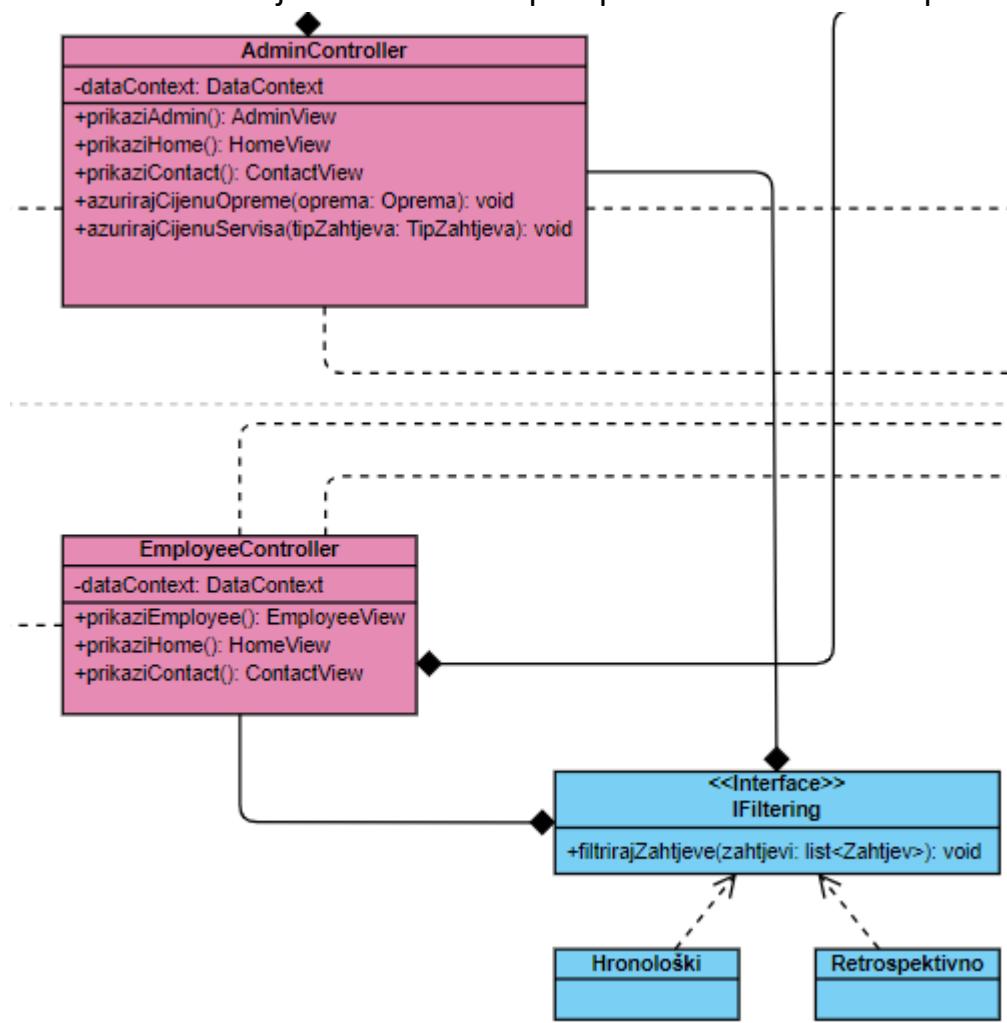
## **5. Factory method**

Uloga Factory Method paterna je da omogući kreiranje objekata na način da podklase odluče koju klasu instancirati. Različite podklase mogu na različite načine implementirati interfejs. Factory Method instancira odgovarajuću podklasu(izvedenu klasu) preko posebne metode na osnovu informacije od strane klijenta ili na osnovu tekućeg stanja. U našem sistemu je ovaj pattern moguće iskoristiti kod sistema personaliziranih preporuka gdje samo na osnovu informacije o visini klijenta instanciraju se odgovarajući objekti.



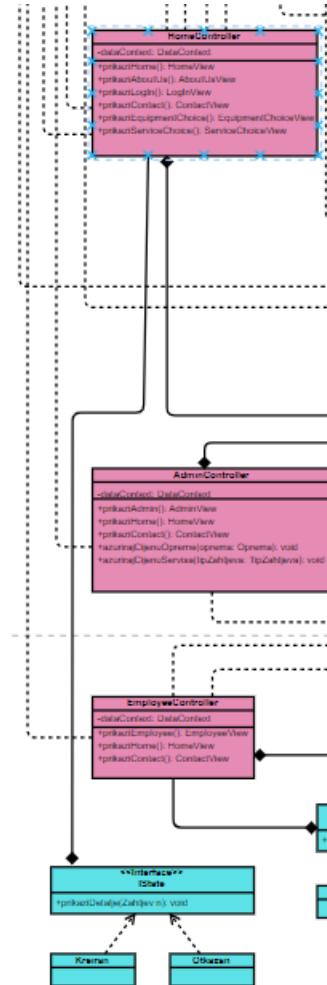
## Patterni ponašanja

Strategy pattern – izdvaja algoritam iz matične klase i uključuje ga u posebne klase. Pogodan je kada postoje različiti algoritmi za neki problem i omogućava korisniku odabir jednog od implementiranih algoritama. U našem konkretnom primjeru ćemo iskoristiti strategy pattern prilikom filtriranja zahtjeva hronološki i retrospektivno zavisno od vremena kada su napravljeni. Napraviti ćemo interfejs IFiltering koji će biti povezan sa kontrolerima koji rade sa filtriranjem. Također, ovaj pattern se mogao iskoristiti prilikom algoritma preporuke gdje sistem na osnovu unesene visine i vještine korisnika poreporuči određeni set opreme.



State pattern – primjenjuje se kada objekat ima više mogućih stanja. U našem primjeru, zahtjev može biti takav da je napravljen ili otkazan. Zahtjev je napravljen kada su popunjena sva obavezna polja i korisnik je odabrao finaliziranje svog zahtjeva, a zahtjev je otkazan kada korisnik putem svog ID-a zahtjeva svojevoljno

otkaže isti. Ovo ćemo sada i implementirati. Imat ćemo interfejs IState koji će biti povezan sa odgovarajućim kontrolerima. Da ponovimo, mogućnost otkzivanja zahtjeva ima isključivo korisnik.



TemplateMethod pattern – omogućava algoritmima da izdvoje pojedine korake u podklase. U našem projektu bi se mogao ovaj pattern implementirati kreiranjem abstraktne klase prikaza zahtjeva iz koje bi se naslijedivale klase AdminMenu i EmployeeMenu. Da ponovimo, admin ima mogućnost izmjene cijena opreme, dok zaposlenik to nema. Oboje imaju mogućnost pregleda zahtjeva ali ne na isti način.

Chain of Responsibility pattern – predstavlja listu objekata, ukoliko objekat ne može da odgovori proslijeđuje zahtjev narednom u nizu. U našem projektu bi se moglo iskoristiti prilikom obrade korisničkih zahtjeva za pomoć. Okvirno bi se dizajnirao sistem koji bi postavio zaposlenika za podršku korisnicima, te ukoliko on nije u stanju da riješi korisnički problem, on se proslijeđuje dalje menadžeru, pa vlasniku, dakle, hijerarhijski.

Command pattern – razdvaja klijenta koji zahtjeva operaciju i omogućava slanje zahtjeva različitim primaocima. Mogao bi se iskoristiti u našem projektu tako što bi kroz ICommand interfejs imali klase koje bi modificirali određene informacije u nekom zahtjevu. Najlogičnije i najprirodnije bi bilo da se korisnik predomisli i izmijeni svoju narudžbu i npr. umjesto prethodno odabranih skija odluči se za iznajmljivanje snowboarda.

Iterator pattern – omogućava pristup elementima kolekcije sekvencialno bez poznavanja interne strukture kolekcije. Npr. U našem projektu bi se mogao ovaj pattern iskoristiti na način da admin ima pristupe kolekcijama korisnika koji su iznajmili skije i opremu, snowboard i opremu, isključivo popravak opreme itd. Omogućili bi jednostavniji pristup svojim korisnicima tako što bi kategorizovali iste radi lakšeg targetovanja publike ili pregleda korisničkih trendova i sl.

Mediator pattern – enkapsulira protokol za komunikaciju među objektima dozvoljavajući da objekti komuniciraju bez međusobnog poznavanje interne strukture objekta. Npr. U slučaju gdje bi SkiRentify biznis postao poznata franšiza na svim poznatim skijalištima iz BiH, bilo bi korisno implementirati Mediator pattern zarad lakše komunikacije između pojedinačnih biznisa, eventualne razmjene inventara itd. Zaposlenici bi eventualne informacije o stanju inventara zapisivali i u slučaju krize u vidu nestanka skija, tu informaciju šalju mediatoru koji obavještava sljedeći najbliži SkiRentify biznis.

Observer pattern – uspostavlja relaciju između objekata takvu da kad se stanje jednog objekta promjeni svi vezani objekti dobiju informaciju. Poželjno je koristiti ovaj pattern prilikom promjene stanja raspoloživosti nekog proizvoda. Npr. U našem projektu bi se iskoristilo da obavijesti korisnika ukoliko je u SkiRentify skladištu ponestalo skija za iznajmljivanje. U tom slučaju bi korisnik bio obaviješten o tome i ne bi mogao iznajmiti nepostojeće skije.

Visitor pattern – definira i izvršava nove operacije nad elementima postojeće strukture ne mijenjajući samu strukturu. Ovaj pattern bi koristili za prolazak kroz listu gdje bi morali izvršiti neku jedinstvenu operaciju nad svakim elementom liste. Dakle kada želimo dodati neku novu funkcionalnost klasi bez prevelikog mijenjanja koda, visitor pattern je izuzetno koristan. Npr. ako bismo htjeli generisati izvještaj o prometu unutar biznisa.

Interpreter pattern – podržava interpretaciju instrukcija napisanih za određenu upotrebu. Za podršku višejezičnog prikaza web aplikacije, neophodno bi bilo implementirati interpreter pattern.

Memento pattern – omogućava spašavanje internog stanja nekog objekta van sistema i njegovo ponovno vraćanje. Poprilično self-explanatory, prilikom nekog kraha sistema bi imali spašeno posljednje ispravno stanje koji bi mogli uzeti za trenutno stanje.

[O nama](#)[Kontakt](#)[Prijava za osoblje](#)

# SkiRentify - vaš skijaški san sa nama postaje stvarnost

[Iznajmljivanje](#)[Servisiranje](#)

Kontakt

Početna

# Prijava na sistem

Korisničko ime

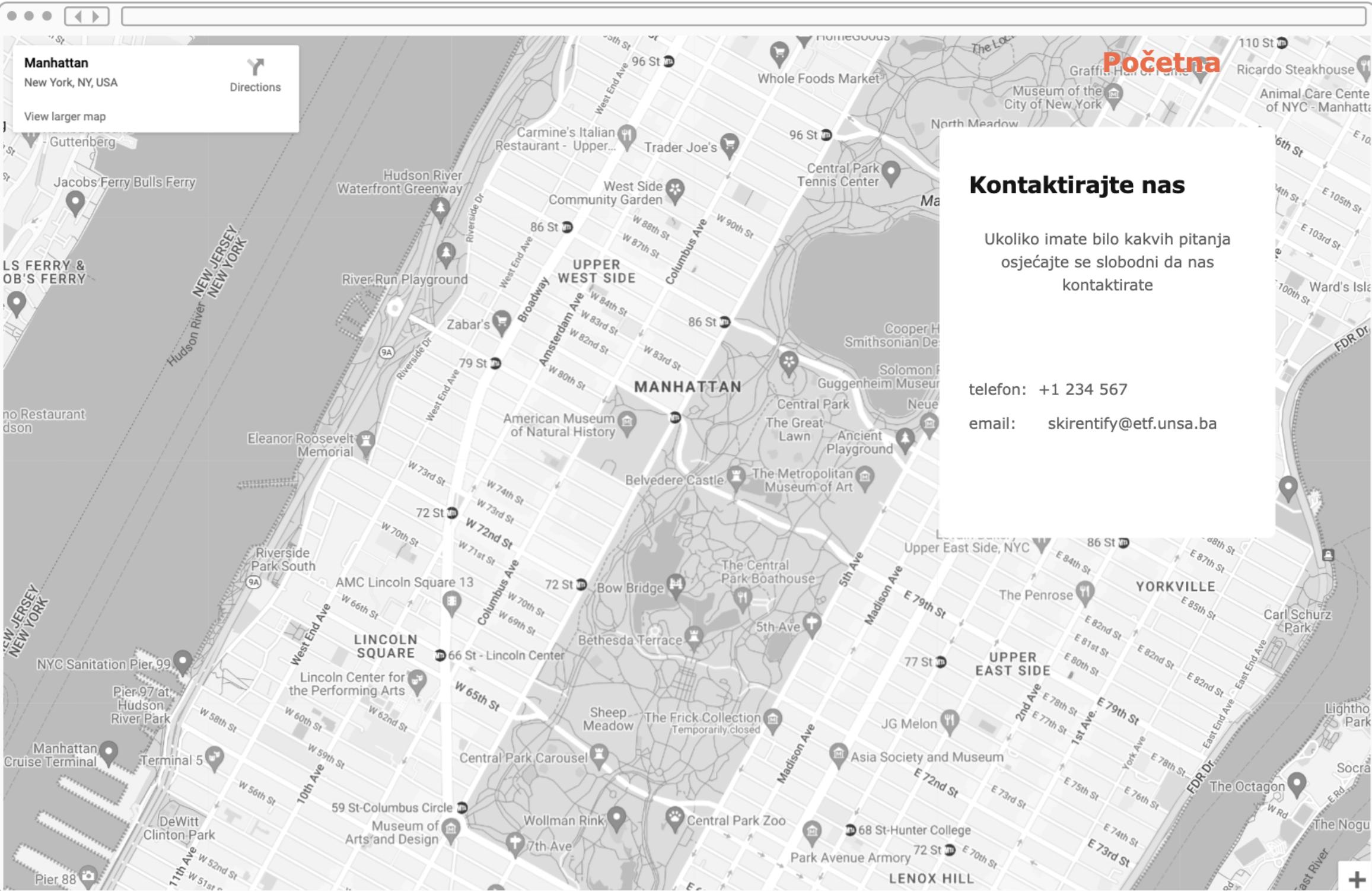
Lozinka

Prijavi se

## O nama

SkiRentify je web aplikacija koja omogućava iznajmljivanje i servisiranje skijaške opreme. Korisnici mogu pregledati dostupnu opremu, odabrati odgovarajuću te izvršiti uplatu. Tako se osigurava olakšan proces dobivanja navednih usluga bez dugih redova čekanja u poslovnici.

[Nazad na početnu](#)



Početna

## Kontaktirajte nas

Ukoliko imate bilo kakvih pitanja  
osjećajte se slobodni da nas  
kontaktirate

telefon: +1 234 567

email: skirentify@etf.unsa.ba

Odaberite opremu/uslugu koju želite da iznajmите



Skije

Odaberite



Pancerice

Odaberite



Štapovi

Odaberite



Kaciga

Odaberite



Snowboard

Odaberite



Snowboard čizme

Odaberite

Dalje

Odaberite uslugu koju želite



### Popravak vezova

Odaberite



### Poliranje skija

Odaberite

Dalje

## Formiranje zahtjeva

Unesite sljedeće podatke:

Ime

Prezime

Broj telefona

E-mail

Visina ( u cm)

Nivo vještine

Datum izdavanja usluge

Datum završetka usluge

Odabrana oprema:



Skije

15km

Preporučena dužina skija

Brend



Pancerice

10km

Brend

Broj



Štapovi

5km

Preporučena dužina štapova

Brend



Kaciga

5km

Brend

Veličina

Idi na plaćanje

[Kontakt](#)[Početna](#)

## Plaćanje

Broj kartice

---

Ime na kartici

---

CVV

---

Plati

## Formiranje zahtjeva

Ime

Odabrane usluge:

Prezime



Popravak vezova

30km

Broj telefona

E-mail



Poliranje skija

15km

Datum donošenja opreme

Datum završetka usluge

[Idi na plaćanje](#)

## Pregled svih zahtjeva za današnji dan

| ID | Ime     | Prezime | Broj telefona | E-mail               | Oprema   | Usluge servisiranja |
|----|---------|---------|---------------|----------------------|----------|---------------------|
| 1  | Klijent | Klijent | +1 234 567    | klijent1@etf.unsa.ba | Odabрано | Odabрано            |
| 2  | Klijent | Klijent | +1 234 567    | klijent2@etf.unsa.ba | Odabрано | Odabрано            |
| 3  | Klijent | Klijent | +1 234 567    | klijent3@etf.unsa.ba | Odabрано | Odabрано            |
| 4  | Klijent | Klijent | +1 234 567    | klijent4@etf.unsa.ba | Odabрано | Odabрано            |
| 5  | Klijent | Klijent | +1 234 567    | klijent5@etf.unsa.ba | Odabрано | Odabрано            |

## Odaberite uslugu

[Pregled zahtjeva](#)[Ažuriranje cijena](#)

## Pregled svih zahtjeva

| ID | Ime     | Prezime | Broj telefona | E-mail               | Oprema   | Usluge servisiranja | Datum izdavanja/završetka |
|----|---------|---------|---------------|----------------------|----------|---------------------|---------------------------|
| 1  | Klijent | Klijent | +1 234 567    | klijent1@etf.unsa.ba | Odabрано | Odabрано            | 01-01-2025<br>05-05-2025  |
| 2  | Klijent | Klijent | +1 234 567    | klijent2@etf.unsa.ba | Odabрано | Odabрано            | 01-01-2025<br>05-05-2025  |
| 3  | Klijent | Klijent | +1 234 567    | klijent3@etf.unsa.ba | Odabрано | Odabрано            | 01-01-2025<br>05-05-2025  |
| 4  | Klijent | Klijent | +1 234 567    | klijent4@etf.unsa.ba | Odabрано | Odabрано            | 01-01-2025<br>05-05-2025  |
| 5  | Klijent | Klijent | +1 234 567    | klijent5@etf.unsa.ba | Odabрано | Odabрано            | 01-01-2025<br>05-05-2025  |

## Odaberite opremu/uslugu koju želite da izmijenite

**Skije**

20km

**Pancerice**

10km

**Štapovi**

10km

**Kaciga**

5km

**Snowboard**

20km

**Snowboard čizme**

10km

[Idi na usluge servisiranja](#)

Odaberite opremu/uslugu koju želite da izmijenite



**Poliranje skija**

15km



**Popravak vezova**

30km

## Dijagrami slučajeva upotrebe – Scenariji

### Scenario 1

|  |   |
|--|---|
| <b>Naziv slučaja upotrebe</b>            | Odabir usluga   |
| <b>Opis slučaja upotrebe</b>             | Akter sistema bira vrstu usluge   |
| <b>Vezani zahtjevi</b>                   | -   |
| <b>Preduslovi</b>                        | -   |
| <b>Posljedice – uspješan završetak</b>   | Otvaranje web interfejsa za odabranu uslugu   |
| <b>Posljedice – neuspješan završetak</b> | -   |
| <b>Primarni akteri</b>                   | Klijent, vlasnik, uposlenik   |
| <b>Ostali akteri</b>                     | -   |
| <b>Glavni tok</b>                        | Akter sistema pri ulasku bira vrstu usluge. On vrši prijavu u slučaju da je vlasnik/uposlenik te dobija njihove privilegije. Klijent bira vrstu usluge koja mu je potrebna. Na osnovu njihovih odabira prikazuje se odgovarajući interfejs. |
| <b>Alternative/proširenja</b>            | Prijava   |

### Tok događaja 1.1 – Uspješan završetak za klijenta

| Klijent               | Sistem                                   |
|-----------------------|--|
| 1. Odabir usluge      |  |
|                       | 2. Prikaz interfejsa odgovarajuće usluge |
| 3. Pregled interfejsa |  |

## Tok događaja 1.2 – Uspješan završetak za vlasnika/uposlenika

| Vlasnik/uposlenik           | Sistem                  |
|-----------------------------|-------------------------|
| 1. Unos podataka za prijavu |                         |
|                             | 2. Validacija podataka  |
|                             | 3. Odobravanje pristupa |
|                             | 4. Prikaz interfejsa    |
| 5. Pregled interfejsa       |                         |

## Scenario 2

|  |  |
|--|--|
| <b>Naziv slučaja upotrebe</b>            | Sistem personaliziranih preporuka  |
| <b>Opis slučaja upotrebe</b>             | Klijent dobija preporučenu opremu na osnovu unesenih informacija   |
| <b>Vezani zahtjevi</b>                   | -  |
| <b>Preduslovi</b>                        | 1. Odabrana usluga iznajmljivanja  |
| <b>Posljedice – uspješan završetak</b>   | Prikaz preporučene opreme  |
| <b>Posljedice – neuspješan završetak</b> | Ponovni unos podataka  |
| <b>Primarni akteri</b>                   | Klijent  |
| <b>Ostali akteri</b>                     | -  |
| <b>Glavni tok</b>                        | Nakon što klijent odabere uslugu iznajmljivanja, treba da unese podatke na osnovu kojih mu sistem preporučuje najprikladniju opremu. |
| <b>Alternative/proširenja</b>            | -  |

### Tok događaja 2.1 – Uspješan završetak

| Klijent                             | Sistem                           |
|-------------------------------------|----------------------------------|
| 1. Unos podataka                    |                                  |
|                                     | 2. Validacija i analiza podataka |
|                                     | 3. Prikaz preporučene opreme     |
| 4. Izbor između preporučenih opcija |                                  |

### Tok događaja 2.2 – Neuspješan završetak

| Klijent                  | Sistem                                      |
|--------------------------|---|
| 1. Unos podataka         |   |
|                          | 2. Validacija i analiza podataka            |
|                          | 3. Prikaz greške prilikom neispravnog unosa |
| 4. Ponovni unos podataka |   |

### Scenario 3

|  |   |
|--|---|
| <b>Naziv slučaja upotrebe</b>            | Ažuriranje cijena usluga                    |
| <b>Opis slučaja upotrebe</b>             | Vlasnik ima mogućnost izmjene cijena usluga |
| <b>Vezani zahtjevi</b>                   | -   |
| <b>Preduslovi</b>                        | 1. Prijava vlasnika                         |
| <b>Posljedice – uspješan završetak</b>   | Izmjenjena cijena usluge                    |
| <b>Posljedice – neuspješan završetak</b> | -   |
| <b>Primarni akteri</b>                   | Vlasnik                                     |
| <b>Ostali akteri</b>                     | -   |

|                               |  |
|-------------------------------|--|
| <b>Glavni tok</b>             | Nakon što se vlasnik prijavi na sistem i odabere opciju ažuriranja cijena, može izmijeniti cijenu za uslugu koju želi. |
| <b>Alternative/proširenja</b> | -  |

#### Tok događaja 3.1 – Uspješan završetak

| Vlasnik                               | Sistem                         |
|---------------------------------------|--------------------------------|
| 1. Odabir opcije ažuriranja cijena    |                                |
|                                       | 2. Prikaz svih usluga          |
| 3. Odabir usluge za promjenu cijene   |                                |
| 4. Unos željene cijene i njeno slanje |                                |
|                                       | 5. Ažuriranje cijene u sistemu |

#### Scenario 4

|  |   |
|--|---|
| <b>Naziv slučaja upotrebe</b>            | Pregled svih zahtjeva   |
| <b>Opis slučaja upotrebe</b>             | Vlasnik/uposlenik ima opciju pregleda svih zahtjeva   |
| <b>Vezani zahtjevi</b>                   | -   |
| <b>Preduslovi</b>                        | 1. Prijava vlasnika/uposlenika  |
| <b>Posljedice – uspješan završetak</b>   | Omogućen pregled stanja   |
| <b>Posljedice – neuspješan završetak</b> | -   |
| <b>Primarni akteri</b>                   | Vlasnik, uposlenik  |
| <b>Ostali akteri</b>                     | -   |
| <b>Glavni tok</b>                        | Nakon što se vlasnik/uposlenik prijavi na sistem i odabere opciju pregleda zahtjeva, dobija prikaz istih. |
| <b>Alternative/proširenja</b>            | -   |

### Tok događaja 4.1 – Uspješan završetak za vlasnika

| Vlasnik                            | Sistem   |
|------------------------------------|--|
| 1. Odabir opcije pregleda zahtjeva |  |
|                                    | 2. Prikaz svih mogućih zahtjeva upućenih sistemu |
| 3. Pregled zahtjeva                |  |

### Tok događaja 4.2 – Uspješan završetak za uposlenika

| Uposlenik                          | Sistem                        |
|------------------------------------|-------------------------------|
| 1. Odabir opcije pregleda zahtjeva |                               |
|                                    | 2. Prikaz zahtjeva za taj dan |
| 3. Pregled zahtjeva                |                               |

### Scenario 5

|  |   |
|--|---|
| <b>Naziv slučaja upotrebe</b>            | Prijava   |
| <b>Opis slučaja upotrebe</b>             | Uposlenik/vlasnik se prijavljuju uz pomoć username/passworda da bi pristupili web aplikaciji                                      |
| <b>Vezani zahtjevi</b>                   | -   |
| <b>Preduslovi</b>                        | <ol style="list-style-type: none"> <li>1. Posjedovanje validnog username/passworda</li> <li>2. Odabir usluge - prijava</li> </ol> |
| <b>Posljedice – uspješan završetak</b>   | Prikaz interfejsa za uposlenika/vlasnika  |
| <b>Posljedice – neuspješan završetak</b> | Poruka o neuspješnoj prijavi  |
| <b>Primarni akteri</b>                   | Uposlenik i vlasnik   |
| <b>Ostali akteri</b>                     | -   |
| <b>Glavni tok</b>                        | Uposlenik i vlasnik unose svoje podatke u predviđena polja za username i password.  |

|                               |   |
|-------------------------------|---|
| <b>Alternative/proširenja</b> | - |
|-------------------------------|---|

#### Tok događaja 5.1 – Uspješan završetak

| Vlasnik/uposlenik            | Sistem  |
|------------------------------|---|
| 1. Unos username i passworda |   |
|                              | 2. Validacija podataka                          |
|                              | 3. Odobravanje pristupa                         |
|                              | 4. Prikaz interfejsa za uposlenika ili vlasnika |

#### Tok događaja 5.2 – Neuspješan završetak

| Vlasnik/uposlenik            | Sistem                                 |
|------------------------------|--|
| 1. Unos username i passworda |  |
|                              | 2. Validacija podataka                 |
|                              | 3. Odbijanje pristupa                  |
|                              | 4. Prikaz poruke o neuspješnoj prijavi |
| 5. Ponovni unos podataka     |  |

## Scenario 6

|  |   |
|--|---|
| <b>Naziv slučaja upotrebe</b>            | Plaćanje  |
| <b>Opis slučaja upotrebe</b>             | Klijent plaća usluge preko web aplikacije   |
| <b>Vezani zahtjevi</b>                   | -   |
| <b>Preduslovi</b>                        | 1. Odabir usluge – iznajmljivanje/servisiranje  |
| <b>Posljedice – uspješan završetak</b>   | Uspješan prikaz jedinstvenog ID zahtjeva  |
| <b>Posljedice – neuspješan završetak</b> | Povratak na ponovni unos podataka za plaćanje   |
| <b>Primarni akteri</b>                   | Klijent   |
| <b>Ostali akteri</b>                     | -   |
| <b>Glavni tok</b>                        | Nakon što klijent odabere uslugu i formira svoj zahtjev, on unosi svoje lične podatke i podatke o kartici te vrši plaćanje. |
| <b>Alternative/proširenja</b>            | 1. Obračun popusta  |

### Tok događaja 6.1 – Uspješan završetak

| Klijent          | Sistem                                  |
|------------------|---|
|                  | 1. Formiranje zahtjeva                  |
|                  | 2. Obračun popusta na osnovu broja dana |
| 3. Unos podataka |   |
|                  | 4. Validacija podataka                  |
|                  | 4. Generiranje ID zahtjeva              |
|                  | 5. Prikaz ID zahtjeva                   |

### Tok događaja 6.1 – Neuspješan završetak

| Klijent          | Sistem                                  |
|------------------|---|
|                  | 1. Formiranje zahtjeva                  |
|                  | 2. Obračun popusta na osnovu broja dana |
| 3. Unos podataka |   |
|                  | 4. Validacija podataka                  |
|                  | 5. Prikaz greške pri unosu              |
| 6. Ponovni unos  |   |

### Scenario 7

|  |   |
|--|---|
| <b>Naziv slučaja upotrebe</b>            | Lokacija poslovnice   |
| <b>Opis slučaja upotrebe</b>             | Svi akteri sistema vide lokaciju poslovnice                                   |
| <b>Vezani zahtjevi</b>                   | -   |
| <b>Preduslovi</b>                        | -   |
| <b>Posljedice – uspješan završetak</b>   | Ispravno prikazana lokacija   |
| <b>Posljedice – neuspješan završetak</b> | -   |
| <b>Primarni akteri</b>                   | Klijent, vlasnik, uposlenik   |
| <b>Ostali akteri</b>                     | -   |
| <b>Glavni tok</b>                        | Svi akteri sistemi na vidnom mjestu imaju slikovit prikaz lokacije poslovnice |
| <b>Alternative/proširenja</b>            | -   |

### Tok događaja 7.1 – Uspješan završetak

| Klijent/vlasnik/uposlenik | Sistem  |
|---------------------------|---|
|                           | 1. Prikazuje kartu s označenom lokacijom poslovnice |

## Scenario 8

|  |   |
|--|---|
| <b>Naziv slučaja upotrebe</b>            | Obračun popusta   |
| <b>Opis slučaja upotrebe</b>             | Sistem automatski obračunava popust na osnovu upućenog zahtjeva   |
| <b>Vezani zahtjevi</b>                   | -   |
| <b>Preduslovi</b>                        | -   |
| <b>Posljedice – uspješan završetak</b>   | Uspješan obračun popusta  |
| <b>Posljedice – neuspješan završetak</b> | -   |
| <b>Primarni akteri</b>                   | Klijent   |
| <b>Ostali akteri</b>                     | -   |
| <b>Glavni tok</b>                        | Nakon što klijent odabere uslugu iznajmljivanja i unese broj dana, sistem obračunava popust ukoliko je broj dana veći od određenog broja. |
| <b>Alternative/proširenja</b>            |   |

### Tok događaja 8.1 – Uspješan završetak

| Klijent | Sistem                                  |
|---------|---|
|         | 1. Obračun popusta na osnovu broja dana |
|         | 2. Ažuriranje cijene zahtjeva           |
|         | 3. Prikaz cijene                        |

## SOLID PRINCIPI

### 1. Single-Responsibility Principle

Ovaj princip je princip pojedinačne odgovornosti klase. Svaka klasa je odgovorna samo za ono što se tiče direktno nje. Klasa Klijent je odgovorna za informacije o klijentu. Klasa Zahtjev posjeduje instancu klase Klijent, onog klijenta koji je podnio zahtjev. Tip zahtjeva je enum, može biti poliranje, popravak\_vezova i iznajmljivanje. Svaki zahtjev može imati više stavki klase Oprema. Oprema je apstraktna klasa, nju nasljeđuju klase Skije, Pancerice, Štapovi, Snowboard, SnowboardCipele i Kaciga. U ovim klasama različite vrste opreme imaju različite atribute, a klasa Oprema sadrži samo ono što je zajedničko svim klasama koje su iz nje nasljeđene i radi samo ono za šta ona može biti zadužena. Klase Uposlenik i Vlasnik su nasljeđene iz apstraktne klase Osoblje. Naprimjer da smo klasu Vlasnik nasljeđili iz klase Uposlenik, ona bi onda vjerovatno “znala previše”, tj. sadržavala bi i metode koje nikada ne bi pozivala, jer je zamišljeno da vlasnik i uposlenik obavljaju drukčije dužnosti. Tako vodimo računa principu pojedinačne odgovornosti.

### 1. Open-Closed Principle

Po ovom principu, entiteti softvera (klase, moduli, metode) trebaju biti otvoreni za nadogradnju, ali zatvoreni za modifikacije. Sve klase koje su nasljeđene iz klase Oprema, mogu se izmijeniti bez potrebe za izmjenom osnovne klase. Kao i klase Uposlenik i Vlasnik koje su nasljeđene iz klase Osoblje. Možemo im dodavati nove metode, atribute bez promjene klase iz koje su nasljeđene. Možemo ih mijenjati, bez promjene svrhe njihovog postojanja.

### 2. Liskov Substitution Principle

Po ovom principu, podtipovi moraju moći biti zamijenjeni svojim osnovnim tipovima tako da i dalje program ispravno funkcionira. Klase Skije, Pancerice, Štapovi, Snowboard, SnowboardCipele i Kaciga mogu zamijeniti objekte klase Oprema te Klase Uposlenik i Vlasnik mogu zamijeniti objekte klase Osoblje.

### 3. Interface Segregation Principle

Ovaj princip nalaže da klijenti ne treba da ovise o metoda koje neće upotrebljavati. Baš zato u našem slučaju postoje odvojene klase Uposlenik i Vlasnik, koje imaju zajedničke neke metode i atribute, no imaju i raličite metode. Obični uposlenik ne može koristiti metodu promjeniCijenu(), i on o njoj ne treba da ovisi. U slučaju da želimo imati dvije vrste klijenta, implementirali bismo interfejs Klijent, te iz njega naslijedili klase KlijentOnline i KlijentUživo, ako bismo u budućnosti željeli uvesti dvije vrste plaćanja i opet držati se pravila ovog principa.

### 5. Dependency Inversion Principle

Ovaj princip nalaže da moduli visokog nivoa ne bi trebali ovisiti od modula niskog nivoa. Klasa StavkaNarudžbe ovisi direktno od apstraktne klase Oprema. Time klasa Zahtjev može sadržavati više različitih stavki klase Oprema, svejedno koje konkretno klase. Postignuto je da su moduli ovisni od apstrakcija.

## Specifikacija projekta

### 1. Osnovne informacije o sistemu

**Naziv teme:** SkiRentify – Sistem za iznajmljivanje, servisiranje i popravak skijaške opreme

**Logo:**



**Naziv tima:** Tim 25

**Nastavna grupa:** 7

**Link na repozitorij tima:** <https://github.com/OOAD-2023-2024/Tim25-SkiRentify>

**Članovi tima:**

1. (Daris Mujkić 19413)
2. (Amina Kazazović 19364)
3. (Sara Dautbegović 19463)
4. (Mina Duranović 19290)

**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).*

SkiRentify je web aplikacija koja omogućava iznajmljivanje i servisiranje skijaške opreme. Klijenti mogu pregledati dostupnu opremu, odabrati odgovarajuću te izvršiti uplatu. Tako se osigurava olakšan proces dobivanja navednih usluga bez dugih redova čekanja u poslovnici.

## 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: Odabir usluga

**Vrsta funkcionalnosti:** Usluga sistema

**Opis funkcionalnosti:**

*Opisati način ostvarivanja funkcionalnosti sa maksimalno pet rečenica.*

Akter sistema bira uslugu koju želi da realizuje putem sistema. Klijentu je ponuđeno iznajmljivanje, servisiranje skijaške opreme i otkazivanje zahtjeva. Uposlenik može odabrati ili opciju izdavanja opreme korisniku ili pregleda svih upućenih zahtjeva za jedan dan. Vlasnik može odabrati ili opciju pregleda svih upućenih zahtjeva ili ažuriranje cijena usluga.

### 2) Naziv funkcionalnosti: Sistem personaliziranih preporuka

**Vrsta funkcionalnosti:** Operacija sa specifičnim algoritmom obrade

**Opis funkcionalnosti:**

*Opisati način ostvarivanja funkcionalnosti sa maksimalno pet rečenica.*

Klijent unosi visinu i nivo vještine te nakon toga na temelju unesenih informacija, algoritam analizira i procjenjuje najprikladniju vrstu i veličinu skija te dodatnu opremu. Klijentu se zatim prikazuje preporučena opcija opreme prilikom odabira iznajmljivanja.

### 3) Naziv funkcionalnosti: Ažuriranje cijena usluga

**Vrsta funkcionalnosti:** Perzistencija podataka (CRUD operacija)

**Opis funkcionalnosti:**

*Opisati način ostvarivanja funkcionalnosti sa maksimalno pet rečenica.*

Vlasnik ima mogućnost izmjene cijena usluga ukoliko se pojavi potreba za istom.

- 4) **Naziv funkcionalnosti:** Pregled svih zahtjeva

**Vrsta funkcionalnosti:** Operacija sa specifičnim algoritmom obrade

**Opis funkcionalnosti:**

*Opisati način ostvarivanja funkcionalnosti sa maksimalno pet rečenica.*

Uposlenik može pregledati sve zahtjeve koji su zakazani za taj dan dok vlasnik ima pregled svih zahtjeva upućenih sistemu sortiranih u hronološkom poretku.

- 5) **Naziv funkcionalnosti:** Prijava

**Vrsta funkcionalnosti:** Usluga sistema

**Opis funkcionalnosti:**

*Opisati način ostvarivanja funkcionalnosti sa maksimalno pet rečenica.*

Vlasnik i uposlenik pristupaju sistemu sa svojim username i passwordom.

- 6) **Naziv funkcionalnosti:** Plaćanje

**Vrsta funkcionalnosti:** Usluga sistema

**Opis funkcionalnosti:**

*Opisati način ostvarivanja funkcionalnosti sa maksimalno pet rečenica.*

Nakon odabira zahtjeva, klijent popunjava podatke o kartici i vrši plaćanje nakon čega dobija jedinstveni ID kod narudžbe.

7) **Naziv funkcionalnosti:** Lokacija poslovnice

**Vrsta funkcionalnosti:** Korištenje vanjskog uređaja

**Opis funkcionalnosti:**

*Opisati način ostvarivanja funkcionalnosti sa maksimalno pet rečenica.*

Lokacija poslovnice će biti vidljiva na web stranici.

8) **Naziv funkcionalnosti:** Obračun popusta

**Vrsta funkcionalnosti:** Operacija sa specifičnim algoritmom obrade - operacije koje koriste specifične algoritme obrade podataka

**Opis funkcionalnosti:**

*Opisati način ostvarivanja funkcionalnosti sa maksimalno pet rečenica.*

U pozadini, sistem obračunava popust na osnovu broja dana na koliko dugo iznajmljuje opremu.

### 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: Klijent

**Vrsta aktera:** Korisnik usluge

**Funkcionalnosti u kojima akter učestvuje:**

| Funkcionalnost sistema               | Način učešća         |
|--------------------------------------|----------------------|
| 1. Odabir usluga                     | Mogućnost pregleda   |
| 2. Sistem personaliziranih preporuka | Mogućnost pregleda   |
| 6. Plaćanje                          | Mogućnost uređivanja |
| 8. Lokacija                          | Mogućnost pregleda   |

#### b) Naziv aktera: Zaposlenik

**Vrsta aktera:** Zaposlenik sistema

**Funkcionalnosti u kojima akter učestvuje:**

| Funkcionalnost sistema   | Način učešća       |
|--------------------------|--------------------|
| 4. Pregled svih zahtjeva | Mogućnost pregleda |
| 5. Prijava               | Mogućnost pregleda |

#### c) Naziv aktera: Vlasnik



**Vrsta aktera:** Vlasnik

**Funkcionalnosti u kojima akter učestvuje:**

*Način učešća:*

- Mogućnost pregleda
- Mogućnost uređivanja

| Funkcionalnost sistema      | Način učešća         |
|-----------------------------|----------------------|
| 3. Ažuriranje cijena usluga | Mogućnost uređivanja |
| 4. Pregled svih zahtjeva    | Mogućnost pregleda   |
| 5. Prijava                  | Mogućnost pregleda   |

#### 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:** Sigurnost sistema

**Opis:**

*Opisati ograničenje sistema i način na koje se ono ispoljava.*

Samo zaposlenik i vlasnik imaju mogućnost pregleda zahtjeva i mijenjanja podataka u sistemu sa svojim pristupnim podacima

2) **Naziv nefunkcionalnog zahtjeva:** Dostupnost

**Opis:**

*Opisati ograničenje sistema i način na koje se ono ispoljava.*

Sistem je korisnicima na raspolaganju 24h dnevno.

3) **Naziv nefunkcionalnog zahtjeva:** Performanse

**Opis:**

*Opisati ograničenje sistema i način na koje se ono ispoljava.*

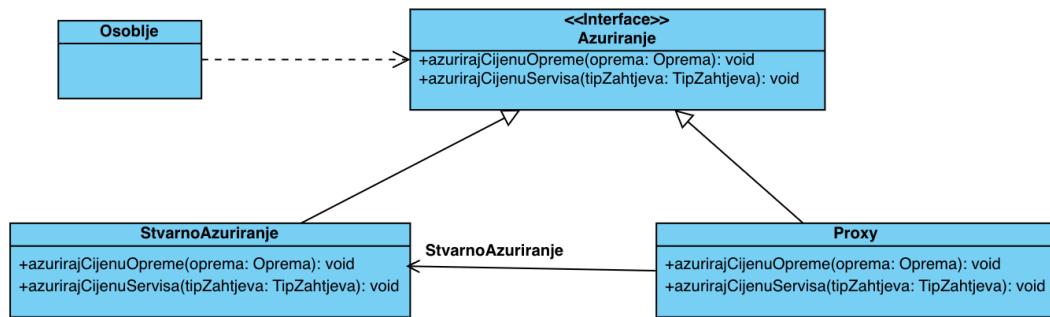
Sistem se prilagođava različitim veličinama uređaja, pruža veliku brzinu učitavanja i visok kvalitet korisničkog iskustva.

# Strukturalni design patterni

Za implementaciju odabrani su Proxy i Bridge design patterni.

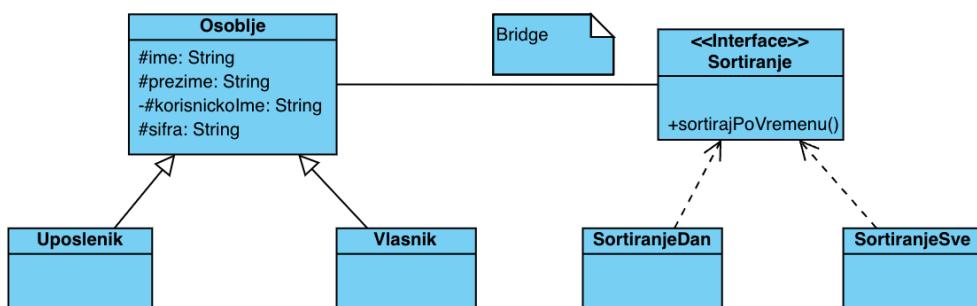
## 1. Proxy

Objekt klase *Oprema*, tačnije atribut cijena ovog objekta, može mijenjati samo vlasnik restorana, dok ostali korisnici ne mogu. Da bi se napravila restrikcija pristupa koristili bi Proxy pattern. Proxy objekt će različitim korisnicima omogućiti različit nivo pristupa štiteći sigurnost aplikacije.



## 2. Brigde

Osnovna namjena Bridge paterna 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. U našem slučaju Implementator odnosno *Sortiranje* zaduženo je za konkretnе implementacije. Klase koje implementiraju taj interfejs sadržavaće specifični kod implementacije, u zavisnosti da li je korisnik vlasnik ili uposlenik, imaće različit pregled zahtjeva.



### **3. Composite**

Osnovna primjena Composite paterna jeste pravljenje hijerarhije klasa i omogućavanje pozivanja iste metode nad različitim objektima sa različitim implementacijama. U našem sistemu, ovaj pattern mogao bi se primijeniti ukoliko klasa *Oprema* bude Composite klasa koja ima svoje elemente (Listove) a klase koje su izvedene od *Oprema*, budu komponente odnosno Leaf klase. One će implementirati neki IComposite interfejs u kome će biti definisano defaultno ponašanje koje će biti zajedničko za pojedinačnu komponentu i njihovu definiciju.

### **4. Adapter**

Adapter patern je design pattern čija je osnovna namjena da interfejs jedne klase pretvori u neki željeni interfejs kako bi se ona mogla koristiti u situaciji u kojoj bi inače problem predstavljali nekompatibilni interfejsi. Primjenom Adapter paterna dobija se željena funkcionalnost bez izmjena na originalnoj klasi i bez ugrožavanja integriteta cijele aplikacije. U našem sistemu možemo iskoristiti ovaj pattern tako što bi omogućili da *Osoblje* može mijenjati zahtjeve *Klijenata*. Onda bi napravili Adapter koji će mapirati interfejs *Klijenta* i prilagoditi ga za *Osoblje*.

### **5. Facade**

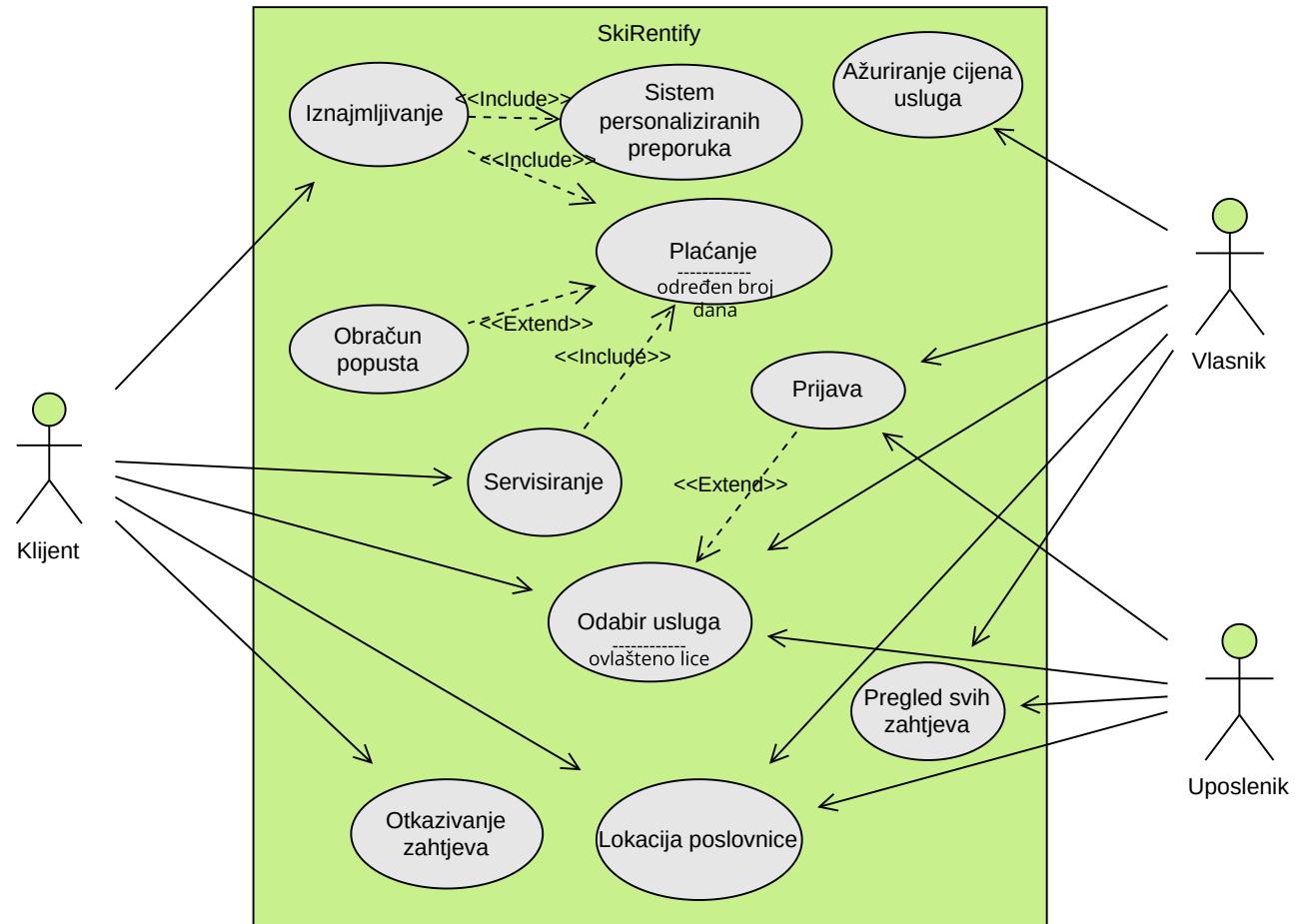
Facade patern se koristi kada sistem ima više identificiranih podsistema pri čemu su apstrakcije i implementacije podistema usko povezane. Osnovna namjena Facade paterna je da osigura više pogleda visokog nivoa na podsisteme. Ovaj patern sakriva implementaciju podistema od korisnika i pruža pojednostavljeni interfejs putem kojeg korisnik pristupa sistemu. U našem sistemu ovaj pattern možemo iskoristiti ukoliko bi napravili fasadnu klasu za *Uposlenika* ili *Vlasnika* tako da sadrži listu svih klijenata, zahtjeva, odvojeno za različite usluge i tako im pružiti pogled na sve podsisteme.

### **6. Decorator**

Osnovna namjena Decorator paterna je da omogući dinamičko dodavanje novih elemenata i ponašanja (funkcionalnosti) postojećim objektima. U našem sistemu ovaj pattern možemo iskoristiti ukoliko bi htjeli da dodamo neke vrste skija, pancerica i sl. Tada bi Decorator klase predstavljale proširenje izbora *Opreme*.

## 7. Flyweight

Flyweight patern koristi se kako bi se onemogućilo bespotrebno stvaranje velikog broja instanci objekata koji u suštini predstavljaju isti objekat. U našem sistemu ovaj pattern možemo iskoristiti ukoliko bi htjeli da sve objekte tipa *Zahtjev* pohranimo u neku listu koja bi mogla memorisati (cache) instance i omogućiti njihovo ponovno korištenje. Određeni korisnik će onda preko FlyweightFactory zahtijevati objekte.



## Analiza i dizajn sistema

U nastavku je potrebno definisati sve potencijalne klase koje će se koristiti u sistemu. Za određivanje klasa koje će biti neophodne za rad sistema potrebno je koristiti specifikaciju sistema i prethodno kreirane dijagrame.

Template za jednu klasu potrebno je iskopirati onoliko puta koliko je neophodno da bi se definisale sve klase u sistemu.

### Definicija klase u sistemu

**Naziv klase:** Klijent

#### Funkcionalni zahtjevi u kojima klasa učestvuje:

Odabir usluga  
Sistem personaliziranih preporuka  
Plaćanje  
Lokacija poslovnice

#### Atributi koje klasa posjeduje:

| Naziv atributa | Tip varijable | Dodatne napomene   |
|----------------|---------------|--|
| ime            | String        | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je enumeration            |
| prezime        | String        | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je enumeration            |
| brojTelefona   | String        | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je enumeration            |
| email          | String        | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je enumeration            |
| visina         | double        | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je enumeration            |
| nivoVjestine   | enum          | <input type="checkbox"/> Atribut je statički<br><input checked="" type="checkbox"/> Atribut je enumeration |

**Naziv klase:** Zahtjev

**Funkcionalni zahtjevi u kojima klasa učestvuje:**

Plaćanje  
Obračun popusta  
Pregled svih zahtjeva

**Atributi koje klasa posjeduje:**

| Naziv atributa          | Tip varijable | Dodatne napomene  |
|-------------------------|---------------|---|
| ID                      | int           | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je <i>enumeration</i>            |
| datumPodnosenjaZahtjeva | DateTime      | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je <i>enumeration</i>            |
| datumIzdavanjaUsluge    | DateTime      | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je <i>enumeration</i>            |
| klijent                 | Klijent       | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je <i>enumeration</i>            |
| tipZahtjeva             | enum          | <input type="checkbox"/> Atribut je statički<br><input checked="" type="checkbox"/> Atribut je <i>enumeration</i> |
| cijena                  | double        | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je <i>enumeration</i>            |
| popust                  | double        | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je <i>enumeration</i>            |

**Naziv klase:** Oprema

**Funkcionalni zahtjevi u kojima klasa učestvuje:**

Plaćanje  
Obračun popusta  
Pregled svih zahtjeva  
Ažuriranje cijena usluga

**Atributi koje klasa posjeduje:**

| Naziv atributa | Tip varijable | Dodatne napomene   |
|----------------|---------------|--|
| cijena         | double        | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je <i>enumeration</i> |
| marka          | String        | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je <i>enumeration</i> |
| materijal      | String        | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je <i>enumeration</i> |

**Naziv klase:** Skije extends Oprema

**Funkcionalni zahtjevi u kojima klasa učestvuje:**

Plaćanje  
Obračun popusta  
Pregled svih zahtjeva  
Ažuriranje cijena usluga

**Atributi koje klasa posjeduje:**

| Naziv atributa | Tip varijable | Dodatne napomene   |
|----------------|---------------|--|
| duzina         | double        | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je <i>enumeration</i> |
| sirina         | double        | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je <i>enumeration</i> |

**Naziv klase:** Pancerice extends Oprema

**Funkcionalni zahtjevi u kojima klasa učestvuje:**

Plaćanje  
Obračun popusta  
Pregled svih zahtjeva  
Ažuriranje cijena usluga

**Atributi koje klasa posjeduje:**

| Naziv atributa | Tip varijable | Dodatne napomene   |
|----------------|---------------|--|
| velicina       | double        | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je <i>enumeration</i> |

**Naziv klase:** Stapovi extends Oprema

**Funkcionalni zahtjevi u kojima klasa učestvuje:**

Plaćanje  
Obračun popusta  
Pregled svih zahtjeva  
Ažuriranje cijena usluga

**Atributi koje klasa posjeduje:**

| Naziv atributa | Tip varijable | Dodatne napomene   |
|----------------|---------------|--|
| duzina         | double        | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je <i>enumeration</i> |

**Naziv klase:** Kaciga extends Oprema

**Funkcionalni zahtjevi u kojima klasa učestvuje:**

Plaćanje  
Obračun popusta  
Pregled svih zahtjeva  
Ažuriranje cijena usluga

**Atributi koje klasa posjeduje:**

| Naziv atributa | Tip varijable | Dodatne napomene   |
|----------------|---------------|--|
| velicina       | char          | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je <i>enumeration</i> |

**Naziv klase:** Snowboard extends Oprema

**Funkcionalni zahtjevi u kojima klasa učestvuje:**

Plaćanje  
Obračun popusta  
Pregled svih zahtjeva  
Ažuriranje cijena usluga

**Atributi koje klasa posjeduje:**

| Naziv atributa | Tip varijable | Dodatne napomene   |
|----------------|---------------|--|
| duzina         | double        | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je <i>enumeration</i> |

**Naziv klase:** SnowboardCipele extends Oprema

**Funkcionalni zahtjevi u kojima klasa učestvuje:**

Plaćanje  
Obračun popusta  
Pregled svih zahtjeva  
Ažuriranje cijena usluga

**Atributi koje klasa posjeduje:**

| Naziv atributa | Tip varijable | Dodatne napomene   |
|----------------|---------------|--|
| velicina       | double        | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je <i>enumeration</i> |

**Naziv klase:** Osoblje

**Funkcionalni zahtjevi u kojima klasa učestvuje:**

Pregled svih zahtjeva  
Ažuriranje cijena usluga  
Odabir usluga  
Prijava

**Atributi koje klasa posjeduje:**

| Naziv atributa | Tip varijable | Dodatne napomene   |
|----------------|---------------|--|
| ime            | String        | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je <i>enumeration</i> |
| prezime        | String        | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je <i>enumeration</i> |
| korisnickoIme  | String        | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je <i>enumeration</i> |
| sifra          | String        | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je <i>enumeration</i> |

**Naziv klase:** Vlasnik extends Osoblje

**Funkcionalni zahtjevi u kojima klasa učestvuje:**

Pregled svih zahtjeva  
Ažuriranje cijena usluga  
Odabir usluga  
Prijava

**Atributi koje klasa posjeduje:**

| Naziv atributa | Tip varijable | Dodatne napomene |
|----------------|---------------|------------------|
|----------------|---------------|------------------|

**Naziv klase:** Uposlenik extends Osoblje

**Funkcionalni zahtjevi u kojima klasa učestvuje:**

Pregled svih zahtjeva  
Ažuriranje cijena usluga  
Odabir usluga  
Prijava

**Atributi koje klasa posjeduje:**

| Naziv atributa | Tip varijable | Dodatne napomene |
|----------------|---------------|------------------|
|----------------|---------------|------------------|

**Naziv klase:** StavkaZahtjeva

**Funkcionalni zahtjevi u kojima klasa učestvuje:**

Pregled svih zahtjeva  
Ažuriranje cijena usluga  
Obračun popusta  
Plaćanje

**Atributi koje klasa posjeduje:**

| Naziv atributa | Tip varijable | Dodatne napomene   |
|----------------|---------------|--|
| oprema         | Oprema        | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je <i>enumeration</i> |
| zahtjev        | Zahtjev       | <input type="checkbox"/> Atribut je statički<br><input type="checkbox"/> Atribut je <i>enumeration</i> |

