## SOLID PRINCIPLES

## Single-Responsibility Principle

Dijagram klasa organizovan je na principu da je svakoj klasi dodjeljena tačno jedna uloga. Npr. klasa Izvodjac, izvedena iz klase Korisnik čuva podatke isključivo vezane za izvodjaca u sistemu; dok klasa Iznajmljivač sadrži sve podatke neophodne za specifikaciju jednog iznajmljivača .Na taj način, svakoj od klasa je dodjeljena tačno jedna odgovornost. Stoga, princip vrijedi za cijeli dijagram.

## Open-Closed Principle

Ovaj princip je u našem sistemu ispunjen za klase Korisnik i Rezervacija. Dodavanjem novih formi rezervacija, kao i dodavanje nove vrste korisnika ne zahtjeva ni promjene ni brisanje kako bazne klase, tako ni drugih klasa. Također, dodavanje novih metoda ne postavlja potrebu za izmjenama, jer sve metode klasa i interfejsa obavljaju samo jednu funkcionalnost.

**Liskov Substitution Principle**

U slučaju obje bazne klase u našem sistemu („Korisnik“ i „Recenzija“) ne postoji potreba da naslijeđenu klasu šaljemo kao parametar umjesto bazne klase. Stoga je ispoštovan LISKOV princip.

## Interface Segregation Principle

Iako trenutno nije realiziran niti jedan interfejs, u slučaju da smo npr. imali interfejs za Običnog Korisnika i Iznajmljivača koji bi realizirao metode koje nisu esencijalne za ove klase, onda bi ovaj princip bio ispunjen jer bi taj interfejs razdvajao funkcionalnosti koje klase Obični Korisnik i Iznajmljivač ne smiju implementirati.

## Dependency Inversion Principle

Pojednostavljenje definicije princip naglašava da klasa ne smije ovisiti od konkrentih klasa, već od apstrakcija, kao i da prilikom nasljeđivanja modulu višeg nivoa ne smiju ovisiti od modula nižeg nivoa. U okviru našeg sistema, primjer klase za koju važi dati princip je klasa „Izvodjač“ , ovisna o klasi „Korisnik“, kao i klasa „RezervacijaDvorane“ koja je ovisna o klasi „Rezervacija“.