1. Princip pojedinačne odgovornosti (Single Responsibility Principle)

KLASA BI TREBALA IMATI SAMO JEDAN RAZLOG ZA PROMJENU.

Zaključujemo da je ovaj princip zadovoljen jer klase našeg sistema uglavnom sadrže gettere i settere za atribute i samo metode koje su direktno vezane za nju. Npr. klasa „Kupovina“ sadrži metode „izvrši transakciju“, „cijena sa popustom“ koje se tiču samo klase „Kupovina“.

1. Otvoreno zatvoreni princip (Open Closed Principle)

ENTITETI SOFTVERA (KLASE, MODULI, FUNKCIJE) TREBALI BI BITI OTVORENI ZA NADOGRADNJU, ALI ZATVORENI ZA MODIFIKACIJE.

Klase našeg sistema su otovorene za nadogradnju, tj. dopuštaju dodavanje novih atributa i metoda, ali da to ne podrazumijeva modifikaciju prethodnog stanja klasa. Npr. klasa „Korisnik“ dopušta dodavanje novih atributa ukoliko je potrebno da se čuva još neka informacija o korisniku, ali bez promjene prethodnih atributa.

1. Liskov princip zamjene (Liskov Substitution Principle)

PODTIPOVI MORAJU BITI ZAMJENJIVI NJIHOVIM OSNOVNIM TIPOVIMA.

U našem sistemu klase „Radnik“ i „Musterija“ nasljeđuju klasu „Korisnik“ i u potpunosti se mogu zamijeniti k klasom „Korisnik“ budući da su sve metode klase „Korisnik“ potrebne i klasi „Radnik“.

1. Princip inverzije ovisnosti (Dependency Inversion Principle)
2. MODULI VISOKOG NIVOA NE BI TREBALI OVISITI OD MODULA NISKOG NIVOA. OBA BI TREBALO DA OVISE OD APSTRAKCIJA.
3. MODULI NE BI TREBALI OVISITI OD DETALJA. DETALJI BI TREBALI OVISITI OD APSTRAKCIJA.

Također, možemo ovaj princip primijeniti kod klasa „Radnik“ i „Musterija“ jer obje te klase ovise od apstraktne klase ili klase koje ovise od interfejsa, kao npr. klasa „Probna voznja“.

1. Princip izoliranja interfejsa (Interface Segregation Principle)

KLIJENTI NE TREBA DA OVISE O METODAMA KOJE NEĆE UPOTREBLJAVATI.

Ovaj princip je zadovoljen što vidimo iz toga da klase nemaju puno metoda već samo one koje su potrebne korisnicima za rad, da se ne bi opterećivao korisnik metodama koje mu nisu potrebne za rad.