# SOLID principi

1.SINGLE RESPONSIBILITY PRINCIPLE (SRP)

Cilj ovog principa jeste da svaka klasa bude odgovorna samo za sebe. S obzirom da svaka od klasa u dijagramu radi samo jedan posao, ovaj princip je zadovoljen.

2.OPEN/CLOSED PRINCIPLE (OCP)

Po ovome principu trebalo bi da klase budu otvorene za proširenje, ali zatvorene za modifikaciju. Naprimjer, ako posmatramo klase Narudžba i Artikal, promjenom klase artikal, recimo brisanje nekog atributa ili dodavanje nove metode, nećemo uticati na klasu Narudžba, tj ona se neće promjeniti. Ova ista logika se može primjeniti i preko ostatka dijagrama.

3.LISKOV SUBSTITUTION PRINCIPLE (LSP)

Ovaj princip nam govori o tome da bi objekti trebalo da budu zamjenjivi svojim potklasama, tj. da ne utiču na ispravnost programa. U našem dijagramu, imamo klasu Korisnik, iz koje su naslijeđene klase RegistrovaniKorisnk, Admin i Uposlenik. S obzirom da je ispunjeno to da svi podtipovi moraju biti zamjenjivi njihovim osnovnim tipovima, možemo reći da je ovaj princip zadovoljen.

4.INTERFACE SEGREGATION PRINCIPLE (ISP)

Ovaj princip govori kako bi trebalo da se bude usmjereno ka korisnicima, a ne ka implementaciji, tako da bi klijenti trebalo da zavise samo od onoga što im je potrebno. S obzirom da u dijagramu vidimo kako svaka klasa obavlja aktivnosti vezane samo za nju, te da nije potrebno uvijek iskoristiti sve metode vezane za istu, možemo reći da je i ovaj princip zadovoljen.

5.DEPENDENCY INVERSION PRINCIPLE (DIP)

Zahtjev ovog principa jeste da visoki nivo modula ne bi trebalo da zavisi od niskog nivoa modula, već bi oba nivoa trebalo da zavise od apstrakcija, tj klase trebaju zavisiti o apstraktnim klasama, te da ne trebaju zavisiti od detalja. Pošto su kod nas klase RegistrovaniKorisnk, Admin i Uposlenik izvedene iz klase Korisnik koja je apstraktna, a

također mjenjanjem bilo čega unutar neke izvedene klase ne dešava promjena u ostalim, već samo mjenjanjem apstraktne, i ovaj princip je zadovoljen.