**SOLID principi – Principi dobrog dizajna**

*1.Princip pojedinačne odgovornosti*

*Princip pojedinačne odgovornosti ( eng. Single Responsibility Principle-SRP) glasi:*

*KLASA BI TREBALA IMATI SAMO JEDAN RAZLOG ZA PROMJENU.*

Klase za projekat Salona ljepote VBF tima zadovoljavaju ovaj primarni princip. Kao što se može uočiti na dijagramu klasa, pojedine klase uglavnom su zadužene za jednu stvar cjelokupnog projekta, tj. imaju pojedinačnu odgovornost, i samo jedan razlog za eventualne promjene. Na ovaj način smo spriječili krhkost kao osobinu lošeg dizajna.

*2.Otvoreno zatvoren princip*

*Otvoreno zatvoren princip ( eng. Open Closed Principle-OCP) glasi:*

*ENTITETI SOFTVERA (KLASE, MODULI, FUNKCIJE) TREBALI BI BITI OTVORENI ZA*

*NADOGRADNJU, ALI ZATVORENI ZA MODIFIKACIJE.*

Ovaj princip je također sadržan u pomenutim klasama jer entiteti koje posjeduje klasa imaju izričitu ulogu i njihova modifikacija bi uništila dato im značenje za cjelokupan projekat. Međutim, s druge strane, za eventualne nadogradnje su vrlo fleksibilne, te za određenu potrebu dale bi se nadograditi novim karakteristikama, pazeći da se ne naruši primarna uloga entiteta.

*3. Liskov princip zamjene*

*Liskov princip zamjene (eng. Liskov Substitution Principle-LSP) glasi:*

*PODTIPOVI MORAJU BITI ZAMJENJIVI NJIHOVIM OSNOVNIM TIPOVIMA.*

Liskov princip zamjene se krši svaki put kada pokušamo napraviti nelegalan poziv funkcije na izvedenoj klasi. U slučaju ovog dijagrama klasa, to nije slučaj, s obzirom da izvedene klase imaju sve legalne dozvole za poziv funkcija u baznoj klasi. Tj. zamjenjive su njihovim osnovnim tipovima.

*4.Princip inverzije ovisnosti*

*Princip inverzije ovisnosti (eng. Dependency Inversion Principle-DIP) glasi:*

*A. MODULI VISOKOG NIVOA NE BI TREBALI OVISITI OD MODULA NISKOG NIVOA.*

*OBA BI TREBALO DA OVISE OD APSTRAKCIJA.*

*B. MODULI NE BI TREBALI OVISITI OD DETALJA. DETALJI BI TREBALI BITI OVISNI OD*

*APSTRAKCIJA.*

Sa UML tačke gledišta, ovaj princip je veoma lako provjeriti. Pratimo svaku strelicu na UML dijagramu i provjerite da li je cilj strelice interfejs ili jedna apstraktna klasa. Ako nije, i ako je to konkretna klasa, onda je princip inverzije ovisnosti narušen, i sistem će biti osjetljiv na promjene. U našem slučaju konkretno, kao što se vidi iz priloženog dijagrama klasa, bazna klasa je apstraktna.

*5.Princip izoliranja interfejsa*

*Princip izoliranja interfejsa ( eng. Interface Segregation Principle-ISP) glasi:*

*KLIJENTI NE TREBA DA OVISE O METODAMA KOJE NEĆE UPOTREBLJAVATI*

Što se tiče ovog principa, zbog nevelike kompleksnosti, bazna klasa ne sadrži suvišne metode čije promjene bi dovele do problema u izvedenim klasama, jer ove sve koje izlaze iz bazne koriste sve ono što je u njoj sadržano. Na taj način, možemo reći da je i ovaj princip nenarušen u analiziranom dijagramu klasa.