

## Solid principi :

### 1. Single Responsibility Principle:

- svaka klasa ima samo jednu ulogu, što znači da je zadovoljen ovaj princip
- jedina problematična klasa je klasa KozmetickiSalon (jer njen aimplementacija objedinjuje više različitih klasa sa različitim funkcionalnostima), ali i ona zadovoljava ovaj princip jer je njena jedina uloga da upravlja sistemom. Problem smo riješile uvodeći različite interfejse za svaku od funkcionalnosti

### 2. Open/Closed Principle:

- princip je zadovoljen jer je svaka od klasa otvorena za nadogradnju, a zatvorena za modifikaciju
- klase koje bi potencijalno mogle kršiti ovaj princip su klase KozmetickiSalon, Tretman, Rezervacija, Klijent i Uposlenik , ali bilo kakva izmjena u implementaciji klasa (koje su atributi gore navedenih klasa), neće dovesti do modifikacije istih

### 3. Liskov Substitution Principle:

- jedino nasljeđivanje u sistemu klasa je nasljeđivanje Klijenta,Administradora i Uposlenika iz klase Osoba (apstraktna klasa), koja i dalje sadrži iste attribute kao i izvedene klase i zamjenjiva je svim izvedenim klasama, pa je princip zadovoljen.

### 4. Interface Segregation Principle:

- svaki od interfejsa u našoj aplikaciji obavlja samo jednu vrstu akcije, čime je zadovoljen princip S, a samim time zadovoljen je i princip I
- primjer: interfejs Prijava ima samo metodu *prijava* i jedina uloga mu je prijava prilikom pristupa sistemu, dok interfejs Registracija ima metodu *registracija* i jedina uloga mu je registracija prilikom prvog pristupa sistemu

### 5. Dependency Inversion Principle:

- u sistemu imamo jednu baznu klasu, klasu Osoba, koja je apstraktna, a nasljeđuju je klase Klijent,Uposlenik, Administrator, pa je princip zadovoljen.