SOLID PRINCIPI

1. *Single Responsibility Principle (SRP)*
   Definicija: Klasa treba da ima samo jednu odgovornost – treba da postoji samo jedan razlog za promjenu.
   - Korisnik: Odgovorna je isključivo za podatke o korisniku i njegove osnovne informacije. Operacije nad korisničkim profilom su u drugim komponentama. Time se olakšava održavanje i izbjegava miješanje logike.
   - Nekretnina: Predstavlja podatke o nekretnini (opis, cijena, broj soba itd.) – bez preuzimanja logike prikaza, pretrage, itd. Na taj način klasa ostaje jednostavna za razumijevanje.
   - Oglas: Sadrži informacije o objavi oglasa (datum, status, broj pregleda) – bez sadržaja same nekretnine.
   - Obavjestenje: Zadužena je za čuvanje podataka o obavještenju i njegovim kriterijima.
   - Lokacija: Odvojena entitetska klasa koja modeluje fizičku lokaciju nekretnine. Time se omogućava ponovno korištenje i pojednostavljuje ažuriranje lokacijskih podataka.
   - Poruka: Modeluje komunikaciju – ima odgovornost samo za poruke između korisnika. Izolovanjem komunikacije postiže se veća jasnoća i fleksibilnost u razvoju.
   - FilterNekretnina: Čuva filter parametre za filtriranje nekretnina. Time se filter logika odvaja od prikaza i smještanja podataka.

2. *Open Closed Principle (OCP)*
   Definicija: Klasa treba da bude otvorena za proširenje, ali zatvorena za izmjene.
   - Nekretnina: Mogu se dodati nove vrste nekretnina ili dodatni atributi koristeći enum Kategorija i ekstenzije.
   - Korisnik: Može se proširiti u Agent, Vlasnik, Kupac, bez promjene osnovne strukture. Ovo omogućava različita ponašanja specifična za svaki tip korisnika.
   - Obavjestenje, FilterNekretnina: Kriteriji se mogu proširiti kroz dodavanje novih atributa (npr. godina izgradnje), bez izmjene osnovne klase.

3. *Liskov Substitution Principle (LSP)*
   Definicija: Objekti izvedenih klasa treba da mogu da zamijene objekte bazne klase bez narušavanja ponašanja.
   - Korisnik: U slučaju da se koristi nasljeđivanje (npr. Agent : Korisnik), sistem treba da funkcioniše jednako bez obzira na tačan tip korisnika. Sve metode koje očekuju korisnika treba da prihvate i agente, kupce i vlasnike bez posebnog prilagođavanja.

- Nekretnina: Ako se podtip (recimo Stan) naslijedi iz Nekretnina, svi atributi se i dalje očekuju i poštuju strukturu klase.

4. *Princip izoliranja interface-a*
   Definicija: Klijenti ne treba da budu prisiljeni da implementiraju interfejse koje ne koriste.

   Ovaj princip podrazumijeva osiguravanje da korisnici ne budu opterećeni metodama koje im nisu potrebne. Za svaku vrstu korisnika implementiraju se različite metode, uzimajući u obzir da zaposlenici sistema imaju određene privilegije u vezi sa pristupom sistemu.

5. *Princip inverzije ovisnosti*
   Definicija: Viši nivoi koda ne treba da zavise od nižih konkretnih implementacija, već od apstrakcija.

   Ovaj princip interpretira da je sigurno ovisiti o konkretnim klasama koje se neće mnogo mijenjati. Klasa Korisnik je takva da pohranjuje elemente za različite aktore, bez mijenjanja nje same. Ovisnosti se rješavaju putem interfejsa i apstrakcija, što omogućava lakšu zamjenu implementacija bez promjena u osnovnoj logici sistema.