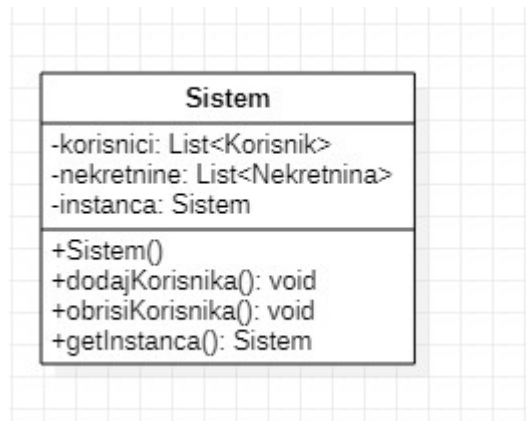


KREACIJSKI PATERNI

Singleton patern

Uloga Singleton paterna je da osigura da se klasa može instancirati samo jednom i da osigura globalni pristup kreiranoj instanci klase.

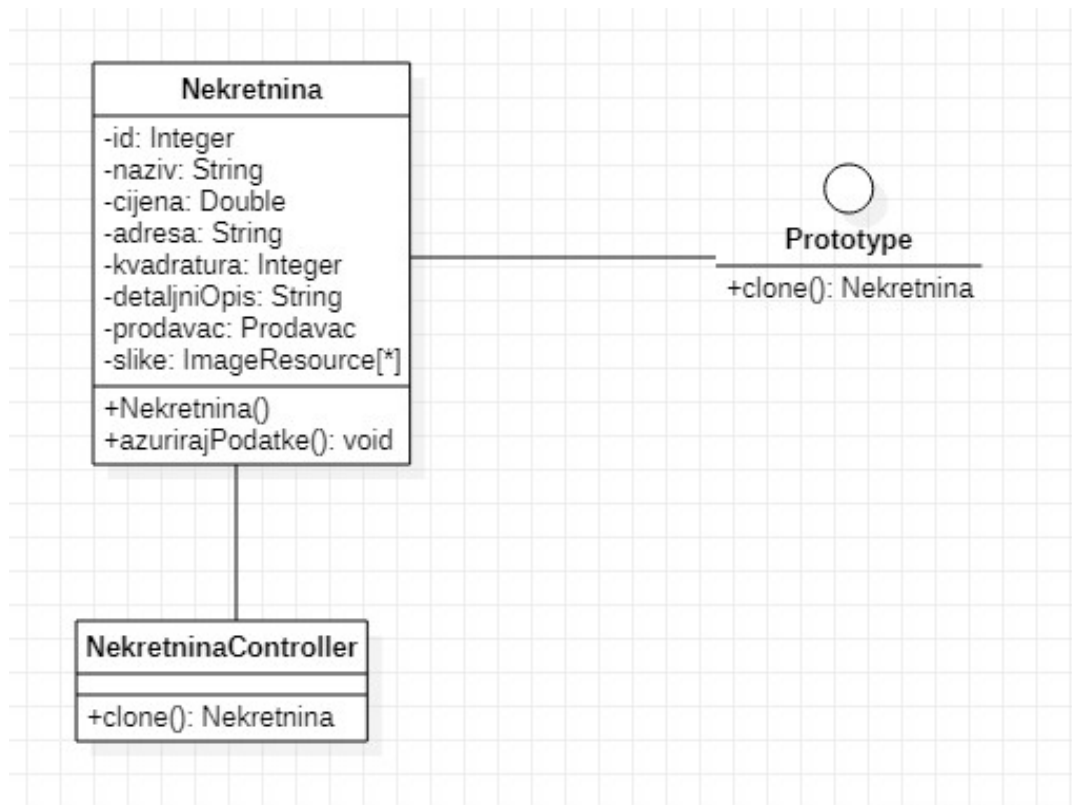
Aplikaciji je potrebna samo jedna instanca određene klase. Toj klasi se treba moći pristupiti sa globalnog nivoa. U našem projektu imamo klasu Sistem koja predstavlja cijeli sistem i potrebna je samo jedna instanca te klase.



Prototype patern

Uloga Prototype paterna je da kreira nove objekte klonirajući jednu od postojećih prototip instanci. Prototype patern dozvoljava da se kreiraju prilagođeni objekti bez poznavanja njihove klase ili detalja kako je objekat kreiran.

U sistemu možemo iskoristiti ovaj patern na klasi Nekretnina s obzirom da većina atributa koji se nalaze u toj klasi su jednaki, pa kreiranje kopije ima smisla.



Factory Method patern

Uloga Factory Method patern je da omogući kreiranje objekata na način da podklase odluče koju klasu instancirati. Različite podklase mogu na različite načine implementirati interfejs. Factory Method patern instancira odgovarajuću podklasu (izvedenu klasu) preko posebne metode na osnovu informacije od strane klijenta ili na osnovu tekućeg stanja.

U našem sistemu ovaj patern možemo iskoristiti kod registracije korisnika, tj. instanciramo objekte različitih tipova (Kupac, Prodavac). To možemo uraditi na način da kreiramo interfejs `IRegistracija`, nakon toga klase `RegistracijaKupac` i `RegistracijaProdavac` koje će implementirati interfejs. Trebamo kreirati i klasu `Creator` koja posjeduje `FactoryMethod()` koja će odlučiti koju klasu će instancirati.

Abstract Factory patern

Abstract Factory patern omogućava da se kreiraju familije povezanih objekata. Na osnovu apstraktne familije produkata kreiraju se konkretne fabrike produkata različitih tipova i različitih kombinacija. Patern odvaja definiciju (klase) produkata od klijenta. Zbog toga se familije produkata mogu jednostavno izmjenjivati ili ažurirati bez narušavanja strukture klijenta.

Ovaj patern u našem sistemu nećemo iskoristiti, ali pošto imamo filtere u sistemu, možemo iskoristiti za jednostavnije filtriranje nekretnina (po lokaciji, cijeni...). Kreirali bi interfejs IAFactory koji bi nam davao instancu Lokacije na osnovu filtera.

Builder patern

Uloga Builder paterna je odvajanje specifikacije kompleksnih objekata od njihove stvarne konstrukcije. Isti konstrukcijski proces može kreirati različite reprezentacije. Koristi se kada je neovisan algoritam za kreiranje pojedinačnih dijelova, kada je potrebna kontrola procesa konstrukcije, kada se više objekata na različit način sastavlja od istih dijelova.

Ovaj patern u našem sistemu nećemo iskoristiti, ali primjer primjene za naš sistem je ukoliko korisnik želi da kupi dodatna parking mjesta koja su vezana za nekretninu, npr. Zgrada. Tada trebamo napraviti novi interface IBuilder i klasu Builder koja će biti povezana sa klasom Nekretnina.