Kreacijski patterni

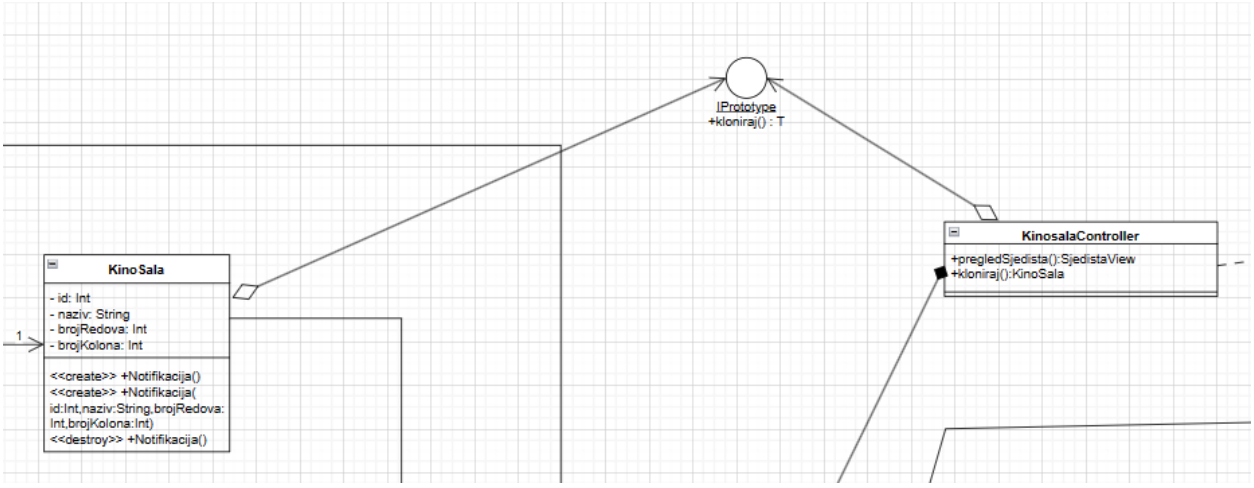
1. Singleton pattern

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

Ovo bi mogli izvesti kada bi posjedovali klasu Log u kojoj bi smjestali sve ocjene, komentare, informacije o dodanim filmovima, maknutim filmova, administratorske bilješke te registracije korisnika ujedno i bilježenje login korisnika. Ovo bi se smještalo u neku tekstualnu datoteku te istu datoteku bi mogli očistiti nakon nekog perioda.

2. Prototype pattern

Osnovna funkcija ovog paterna je da olakša kreiranje objekata koristeći postojeću instancu, koja se ponaša kao prototip. Novokreiranom objektu možemo promijeniti određene osobine po potrebi.



Ovaj pattern smo primjenili nad klasom KinoSala kako bi olaksali biljezenje sala te izvrsimo promjene naziva ili broja redova/kolona ukoliko je to potrebno.

3. Factory method pattern

Factory method patern omogućava kreiranje objekata na način da podklase odluče koju klasu instancirati. Ovo se radi preko interfejsa sa jednom metodom koju različite podklase mogu implementirati drugačije.

Kada bi smo u našoj aplikaciji imali neku vrstu mogućnosti oglasa, kojih bi mogli generisati. Imali bi smo klasu kao Kreator koja bi imala opciju PokreniOglase() onda bi imali klase vise

tipova oglasa primjerice PremijereFilma, Akcije onda imamo klasu IVrstaOglasa gdje ce biti implementirana metoda kreirajOglas().

4. Abstract factory pattern

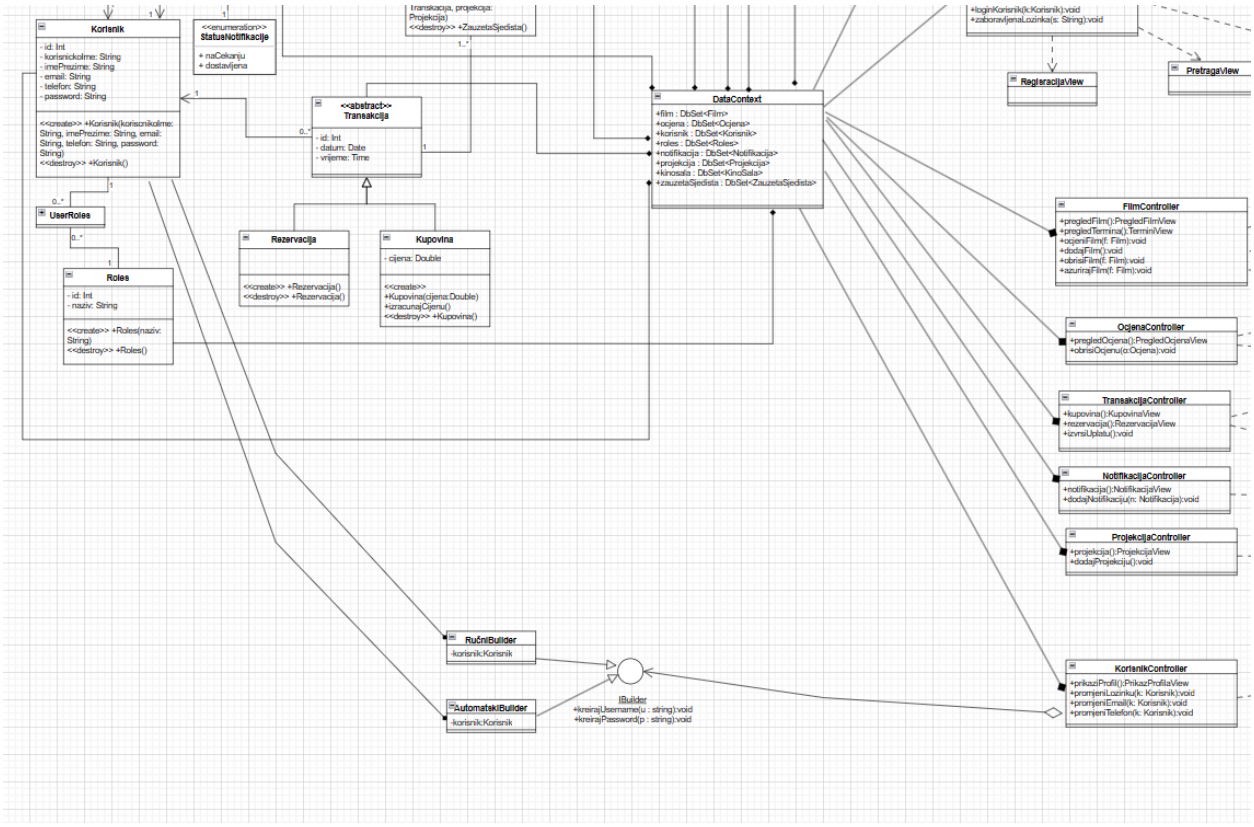
Ovaj patern nam omogućava kreiranje više familija objekata sa mogućnošću dodavanja novih u budućnosti. Abstract factory odvaja definiciju (klase) produkata od klijenta zbog čega se familije produkata mogu jednostavno izmijenjivati ili ažurirati bez narušavanja strukture klijenta.

Kada bi smo željeli implementirati ovaj pattern logčnije bi bilo da posjedujemo još jednu vrstu korisnika a to bi bio premium korisnik. Ono što bi smo omogućili jeste da Korisnik i PremiumKorisnik imaju mogu kupovati karte samo za svoje kategorije. Ovo bi dovelo do toga da PremiumKorisnik bi dobio mogućnost da kupuje posebne karte sa nekim povlasticama.

5. Builder pattern

Builder paterna koristimo kada nam je potrebno primijeniti različite postupke za konstrukciju istog objekta.

Ovaj patern ćemo implementirati tako da omogući korisniku koji se prvi put registruje na sistem da bira između dva načina registrovanja. On bi mogao odabrati opciju za automatsko generiranje username i password polja ili, ukoliko to želi, da ih sam unese.


Korisnik
- id: Int
- korisnickoIme: String
- imePrezime: String
- email: String
- telefon: String
- password: String
<<create>> +Korisnik(korisnickoIme: String, imePrezime: String, email: String, telefon: String, password: String)
<<destroy>> +Korisnik()
<<enumeration>>
StatusNotifikacije
+ naCekanju
+ dostavljena
<<abstract>>
Transakcija
- id: Int
- datum: Date
- vrijeme: Time
DataContext
+film : DbSet<Film>
+ocjena : DbSet<Ocjena>
+korisnik : DbSet<Korisnik>
+roles : DbSet<Roles>
+notifikacija : DbSet<Notifikacija>
+projekcija : DbSet<Projekcija>
+kinosala : DbSet<KinoSala>
+zauzetaSjedista : DbSet<ZauzetaSjedista>
UserRoles
Roles
- id: Int
- naziv: String
<<create>> +Roles(naziv: String)
<<destroy>> +Roles()
Rezervacija
<<create>> +Rezervacija()
<<destroy>> +Rezervacija()
Kupovina
: cijena: Double
<<create>>
+Kupovina(cijena:Double)
+izracunajCijenu()
<<destroy>> +Kupovina()
RegistracijaView
PretragaView
FilmController
+pregledFilm():PregledFilmView
+pregledTermina():TerminiView
+ocjeniFilm(f: Film):void
+dodajFilm():void
+obrisiFilm(f: Film):void
+azurirajFilm(f: Film):void
OcjenaController
+pregledOcjena():PregledOcjenaView
+obrisiOcjenu(o:Ocjena):void
TransakcijaController
+kupovina():KupovinaView
+rezervacija():RezervacijaView
+izvrsiUplatu():void
NotifikacijaController
+notifikacija():NotifikacijaView
+dodajNotifikaciju(n: Notifikacija):void
ProjekcijaController
+projekcija():ProjekcijaView
+dodajProjekciju():void
KorisnikController
+prikaziProfil():PrikazProfilaView
+promjeniLozinku(k: Korisnik):void
+promjeniEmail(k: Korisnik):void
+promjeniTelefon(k: Korisnik):void
RučniBuilder
-korisnik:Korisnik
AutomatskiBulider
-korisnik:Korisnik
IBuilder
+kreirajUsername(u : string):void
+kreirajPassword(p : string):void