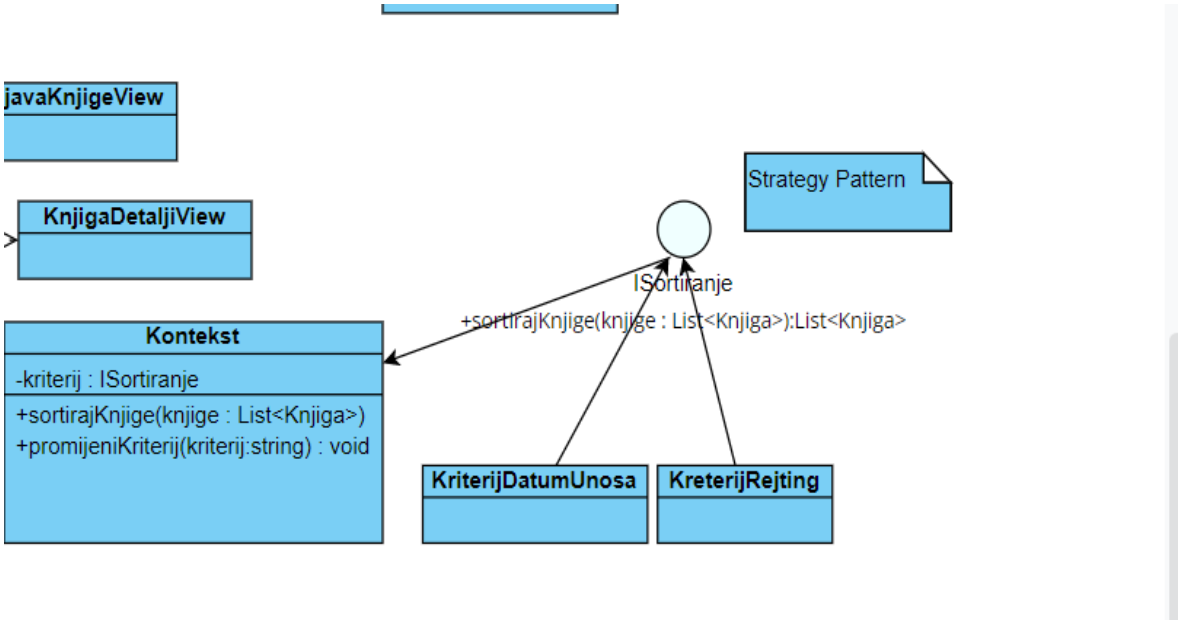
PATERNI PONAŠANJA

1. Strategy pattern

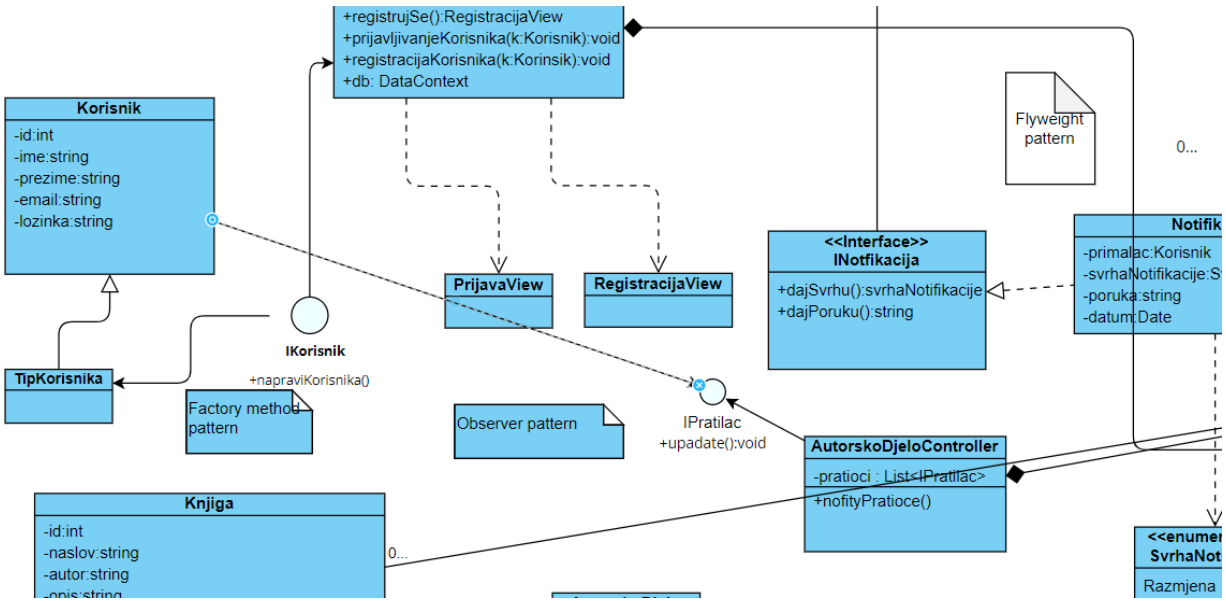
Strategy patern je u našem sistemu iskorišten u okviru funkcionalnosti sortiranja knjiga. Korisnik prilikom pretraživanja, može da izabere sortiranje knjiga po određenom kriteriju. Ovu funkcionalnost smo implementirale pomoću strategy paterna.
Dodale smo klasu Kontekst koja ima atribut kriterij I dvije metode soritirajKnjige() I promijeniKriterij() koji mijenja kriterij. Dodan je I interfej ISortiranje sa metodom sortirajKnjige(), te dvije klase koje realiziraju interfejs: KriterijDatumUnosa I KriterijRejting.



2. Observer patern

U našem sistemu observer patern je iskoristen u slucaju kada imamo situaciju da je vise korisnika zainteresovano za neko autorsko djelo koje vise nije na stanju. U tom slučaju, korisnici se mogu pretplatiti na obavijesti I primiti notifikacuju kad ponovo bude moguće kupiti to djelo.

Definišemo interfejs IPratilac koji će imati metodu update() kojeg će implementirati klasa Korisnik. U klasu AutorskoDjeloController dodajemo listu u kojoj će se nalaziti svi korisnici koji žele da dobiju obavještenje kad knjiga ponovo postane slobodna i dodajemo metodu notifyPratioce() koja im šalje tu informaciju.



3. State pattern

State patern omogućava objektu da mijenja svoja stanja, od kojih zavisi njegovo ponašanje. Sa promjenom stanja objekt se počinje ponašati kao da je promijenio klasu. Stanja se ne mijenjaju po želji klijenta, već automatski, kada se za to steknu uslovi. Ovaj pattern nismo implementirale, ali mogao bi biti implementiran za prikaz stanja knjiga, odnosno je li rezervisana za razmjenu ili ne. Ukoliko korisnik pošalje zahtjev za razmjenu drugom korisniku, knjiga je rezervisana. Stanje je moguće promijeniti u slučaju da korisnik odustane od razmjene.

4. Iterator Pattern

Iterator patern namijenjen je kako bi se omogućio prolazak kroz listu elemenata bez da je neophodno poznavati implementacijske detalje strukture u kojoj se čuvaju elementi liste. Izvedba liste može biti u obliku stabla, jednostruke liste, niza i sl., no klijentu se omogućava da na jednostavan način dolazi do željenih elemenata
U našem sistemu bi to mogli iskoristiti tako sto bi napravili klasu Korisnici, a ona bi simulirala klasu Collection I u njoj bi se cuvali svi korisnici aplikacije. Imali bi interface IKorisnici koji bi u sebi imao metodu getKorisnici koju implementira klasa Korisnici.

5. Template Method Pattern

Template method patern služi za omogućavanje izmjene ponašanja u jednom ili više dijelova. Najčešće se primjenjuje kada se za neki kompleksni algoritam uvijek trebaju izvršiti isti koraci, no pojedinačne korake moguće je izvršiti na različite načine.
Template Method patern bismo mogli iskoristiti u nasem sistemu za sortiranje knjige po različitim parametrima. Dodali bi apstraktnu klasu Sortiranje sa listom knjiga kao privatnim

atributom, ona će imati metodu templateMethod(), te ćemo imati i metode koje će biti implementirane u izvedenim klasama tipa kriterij(). Imat ćemo izvedene klase pod nazivima AutorSort i DatumUnosaSort koje će imati svoju metodu kriterij().

6. Mediator pattern

Mediator patern koristi se kako bi se omogućilo smanjenje broja veza između objekata. Veliki broj objekata bi bio povezan na međuobjektnom medijatorom, koji je zadužen za njihovu interakciju i komunikaciju.

Ono što bi mogli uraditi u našoj aplikaciji jeste da uvedemo neki chat između neregistrovanog korisnika, registrovanog korisnika I super korisnika. Upravo klasa Chat bi bila Medijator klasa. Upravljanje ovim chatom bi imao administrator. Sve informacije o tome ko prima poruku ili ko šalje poruku bi imala ova klasa.

7. Chain of responsibility pattern

Chain of Responsibility pattern služi radi razbijanja procesa obrade na način više objekata koji na različite načine procesiraju primljene podatke. Ovo omogućava smanjivanje kompleksnosti svakog pojedinačnog procesa, kao i jednostavniji pregled Sistema.
U našem sistemu bi mogli ovaj pattern primijeniti pri kupovini knjige. Prije nego sto započnemo process kupovine, potrebno je provjeriti da li korisnik koji želi obaviti kupovinu ima potrebna sredstva na računu.