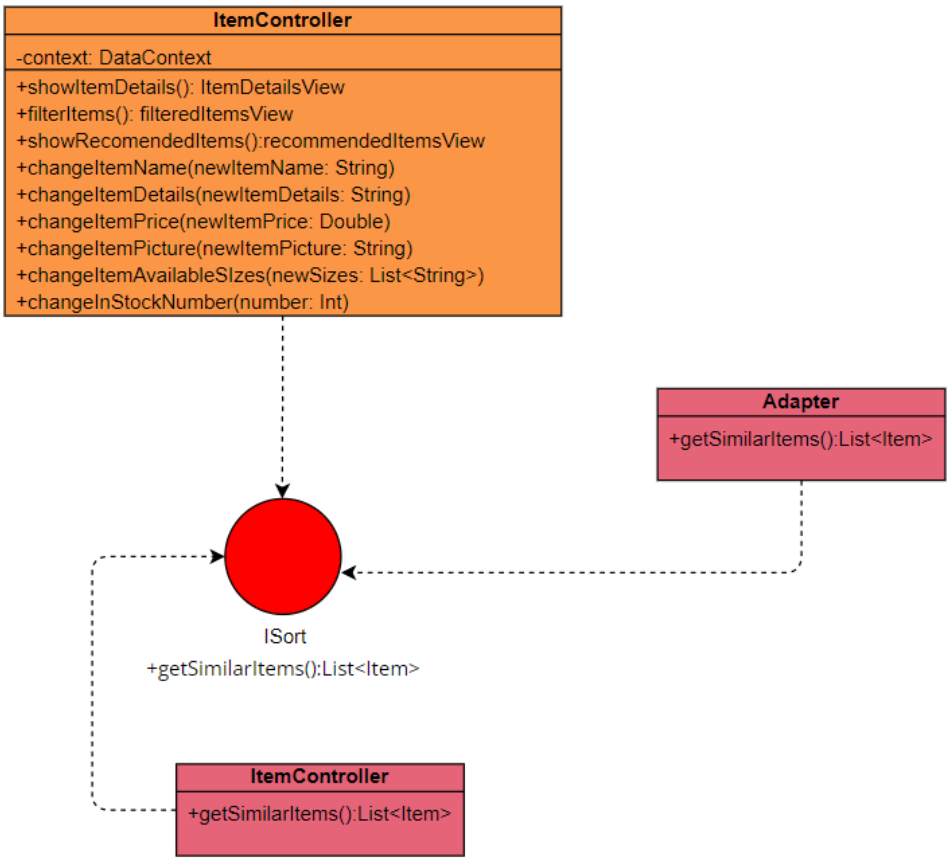
1. ADAPTER PATERN

Osnovna namjena Adapter paterna je da omogući širu upotrebu već postojećih klasa. Adapter patern kreira novu adapter klasu koja služi kao posrednik između originalne klase i željenog interfejsa. Tim postupkom se dobija željena funkcionalnost bez izmjena na originalnoj klasi i bez ugrožavanja integriteta cijele aplikacije.

U nasoj aplikaciji adapter patern se moze iskoristiti tako da implementiramo interface koji ce omoguciti prikazivanje slicnih artikala nakon otvaranja detalja određenog artikla. Klasa adapter implementira interfejs koji sadrzi metodu za dobavljanje slicnih artikala.



2. FACADE PATERN

Facade patern se koristi kada sistem ima više identificiranih podsistema (subsystems) pri čemu su apstrakcije i implementacije podsistema usko povezane. Osnovna namjena je da prikrije neku komplikovanu implementaciju i pruzi jednostavno pozivanje komplikovanih operacija.

U nasoj aplikaciji facade patern se moze iskoristiti za jednostavno pozivanje funkcija filtriranja i sortiranja, a da njihova implementacija bude u pozadini komplikovanija. Ove metode bismo smjestili u jednu klasu tj. klasu Fasad. U ovu klasu bismo mogli smjestiti metode kao što su sortiraj po cijeni (od manje ka većoj i obrnuto), filtriraj po veličini, kategoriji, brendu…

3. DECORATOR PATERN

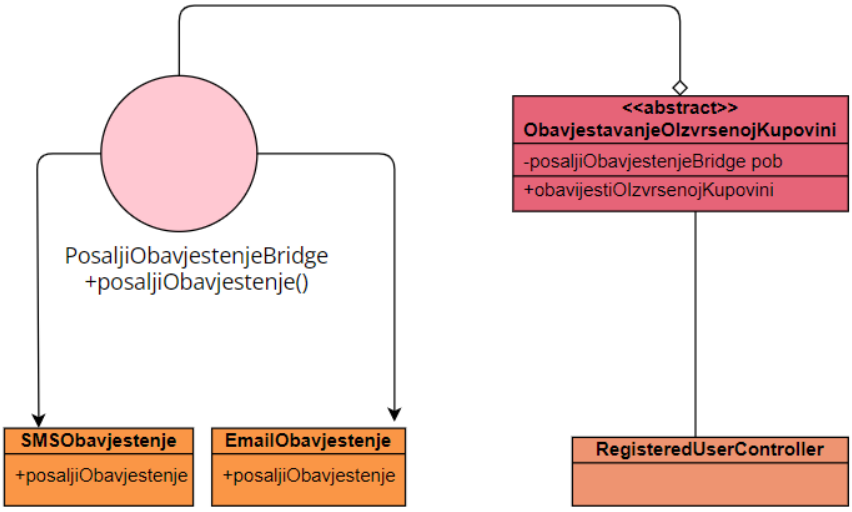
Osnovna namjena Decorator paterna je da omogući dinamičko dodavanje novih elemenata i ponašanja (funkcionalnosti) postojećim objektima.

U nasoj aplikaciji decorator patern se moze iskoristiti tako da omogucimo administratoru da prilikom objavljivanja proizvoda ima mogucnost uredjivanja slika, detalja, i slicno, tako da se prilikom uredjivanja nad objektom dinamicki vrse uredjivanja.

4. BRIDGE PATERN

Osnovna namjena Bridge paterna je da omogući odvajanje apstrakcije i implementacije neke klase tako da ta klasa može posjedovati više različitih apstrakcija i više različitih implementacija za pojedine apstrakcije.

U nasoj aplikaciji bridge patern se moze iskoristiti tako da se verifikacija prilikom registrovanja moze vrsiti na dva nacina, putem maila i sms-a.



5. PROXY PATERN

Namjena Proxy paterna je da omogući pristup i kontrolu pristupa stvarnim objektima. Proxy je obično mali javni surogat objekat koji predstavlja kompleksni objekat čija aktivizacija se postiže na osnovu postavljenih pravila.

U nasoj aplikaciji proxy patern se moze iskoristiti tako da se onemoguci wishlist za registrovanog korisnika koji nije VIP. U nasoj aplikaciji mogli bismo implementirati klasu “WishlistProxy” koja nasljedjuje interfjes sa metodom za pristup wishlistu, koja bi vrsila provjeru kojeg tipa je korisnik i ukoliko korisnik nije VIPUser, pristup wishlistu bi bio onemogucen.

6. COMPOSITE PATERN

Osnovna namjena Composite paterna (kompozitni patern) je da omogući formiranje strukture stabla pomoću klasa, u kojoj se individualni objekti (listovi stabla) i kompozicije individualnih objekata (korijeni stabla) jednako tretiraju (ilustracija 6-postoje različiti načini pregleda fotografija, jedna fotografija može se naći u više albuma, korisnik može prikazati jednu fotografiju ili album, izbrisati i fotografiju i album).

U nasoj aplikaciji composite pattern se moze iskoristit na nacin da se narudzbe obicnog registrovanog korisnika i vip korisnika dostavljacu prikazuju na isti nacin. Oba korisnika trebaju primiti status narudzbe. Medjutim, glavna razlika izmedju ove dvije vrste korisnika je ta da obicni korisnik moze da vidi samo kada je narudzba zaprimljena i izvrsena, dok vip korisnik vidi i trenutak kada je narudzba na putu.

7. FLYWEIGHT PATERN

Osnovna namjena Flyweight paterna je upravo da se omogući da više različitih objekata dijele isto glavno stanje, a imaju različito sporedno stanje

U nasoj aplikaciji flyweight patern se moze iskoristiti na nacin da omogucimo korisnicima da imaju svoju profilnu sliku. Ukoliko neko ne zeli da postavi svoju sliku onda je potrebno koristiti neku defaultnu sliku. Obzirom da vise korisnika moze da ima istu defaultnu sliku (dakle, mogu zadrzati sliku koja je dodijeljena od strane sistema) potrebno je implementirati flyweight patern kako bi korisnici koristili jedan zajednicki resurs.