1. SINGLETON PATTERN

Singleton patern osigurava da se neka klasa može instancirati samo jednom i omogućava globalni pristup toj instanci. Koristan je u slučajevima gdje je potrebna centralizovana kontrola i jedinstven objekt koji upravlja nekom zajedničkom logikom.
U dijagram klasa u koji trebamo da dodamo dva kreacijska paterna, upravo možemo iskoristiti ovaj patern za klasu koju ćemo nazvati RacunGenerator
Dakle, u našem sistemu se Singleton patern primjenjuje nad klasom RacunGenerator. Ova klasa generiše jedinstvene identifikatore i brojeve računa za sve narudžbe koje se kreiraju u sistemu. Budući da mora postojati samo jedan generator koji prati posljednji broj računa, Singleton je idealno rješenje.
To znači da kada u sistemu kreiramo više računa (*E-racun*), svaki od njih mora imati jedinstven broj ili ID (racun #001, #002…) i ako bi svaka instanca sama pokušavala generisati te brojeve, moglo bi doći do mogućih duplikata, nesinhronizacije, grešaka u vođenju evidencije i slično, stoga je Singleton pattern dobra opcija u ovom slučaju.
Također, Singleton pattern bi mogao biti koristan za:

- centralizovano upravljanje sesijom korisnika
- klasu koja vodi evidenciju svih prijava, akcija administratora i grešaka

2. PROTOTYPE PATTERN

Prototype patern omogućava kreiranje objekata na osnovu već postojećih – kloniranjem. Ovo je korisno kada imamo objekte koji se često ponavljaju s malim razlikama.
U sistemu apoteke, Prototype patern koristimo nad klasom NarudzbaProizvoda. Korisnici često prave slične narudžbe, pa im omogućavamo da kloniraju prethodnu narudžbu, a zatim izmijene samo količinu ili dodaju nove proizvode. Time se štedi vrijeme i smanjuje šansa za greške.
Također se može primijeniti na klase:

- korpa – duplikacija postojeće korpe
- proizvod – kada prodavač želi brzo dodati sličan novi proizvod

3. FACTORY METHOD PATTERN

Factory Method patern omogućava da podklase odluče koju konkretnu klasu da instanciraju. On centralizuje logiku kreiranja objekata i omogućava lako dodavanje novih varijanti.
U našoj aplikaciji, koristimo Factory patern za kreiranje različitih tipova korisnika, kao što su Administrator, Farmaceut, RegistrovaniKorisnik, Gost. Umjesto da direktno pozivamo konstruktore, možemo koristiti interfejs IKorisnikFactory sa metodom kreirajKorisnika(). Svaka fabrika implementira kreiranje odgovarajuće vrste korisnika.
Ovo rješenje omogućava:

- jednostavno dodavanje novih tipova korisnika
- bolju organizaciju i manju zavisnost klasa

4. ABSTRACT FACTORY PATTERN

U sistemu e-Apoteke, Abstract Factory patern se može primijeniti kako bi se olakšalo upravljanje različitim tipovima korisnika i njihovim specifičnim funkcionalnostima. Naprimjer, u sistemu koji mi imamo postoje tipovi korisnika poput administratora, farmaceuta, registrovanog korisnika i gosta, a svaki od njih ima različite uloge i pristup određenim dijelovima aplikacije. Umjesto da u kodu mi pravimo razne provjere kroz mnogo Ifova i slično, primjenom Abstract Factory paterna omogućavamo da se obavi centralizovano kreiranje svih potrebnih objekata za svaki tip korisnika. Naprimjer, za gosta bi se kreirali jednostavni interfejsi i ograničene funkcionalnosti, dok bi za administratora fabrika kreirala kompletan interfejs sa opcijama za upravljanje proizvodima, korisnicima i narudžbama. Na ovaj način, za svaki novi tip korisnika koji se eventualno doda u budućnosti, dovoljno je implementirati novu konkretnu fabriku bez potrebe za izmjenom postojećeg koda. Abstract Factory tako doprinosi lakšem održavanju i proširivanju sistema.

5. BUILDER PATTERN

Builder patern omogućava postepeno kreiranje složenih objekata. Koristan je kada objekat ima više opcionalnih ili zavisnih podataka.
U našoj aplikaciji, Builder se koristi za kreiranje objekta NarudzbaProizvoda, jer korisnik može unositi proizvode postepeno, dodavati ili uklanjati stavke, mijenjati adresu dostave, napomene itd. Sve to bi bilo komplikovano sa klasičnim konstruktorom sa puno parametara.
Builder rješenje omogućava:

- čitljiviji i fleksibilniji kod
- mogućnost validacije podataka prije konačnog kreiranja narudžbe