BookMyStyle- kreacijski patterni

U sistemu BookMyStyle, koji omogućava zakazivanje termina u frizerskim salonima, identificirana je potreba za korištenjem kreacijskih paterna radi optimizacije procesa kreiranja objekata te povećanja fleksibilnosti i skalabilnosti sistema. Korištenjem kreacijskih paterna Factory Method i Builder, postigli bismo modularnost, smanjenje zavisnosti između klasa, te olakšanu integraciju budućih proširenja.

Factory Method Pattern

Zašto smo izabrali:

Primjena Factory Method paterna omogućava nam jednostavno i kontrolirano kreiranje različitih tipova obavijesti u sistemu bez direktnog instanciranja konkretnih klasa. Ovaj patern smanjuje povezanost između klijenta i konkretnih implementacija obavijesti, olakšavajući dodavanje novih tipova obavijesti u budućnosti.

Primjena u sistemu:

Korištena je abstraktna klasa „CreatorObavijest“ kao centralna tačka za kreiranje različitih tipova obavijesti. Implementirane su konkretne klase „CreatorEmailObavijest“ i „CreatorQRObavijest“ koje realiziraju kreiranje specifičnih tipova obavijesti.

Primjer implementacije:

- Abstract Class: CreatorObavijest
- Concrete Creators: CreatorEmailObavijest, CreatorQRObavijest
- Product Interface: IObavijest
- Concrete Products: EmailObavijest, QRObavijest

Builder Pattern

Zašto smo izabrali:

Builder pattern odabran je zbog potrebe složenog kreiranja termina koji uključuju različite opcije kao što su usluga, vrijeme, datum i frizer. Ovaj pattern pojednostavljuje kreaciju složenih objekata kroz korake, omogućavajući jednostavniju i jasniju inicijalizaciju objekta bez potrebe za velikim brojem konstruktora.

Primjena u sistemu:

Builder pattern implementiramo za klasu „TerminBuilder“ koja omogućava da se kreiranje termina izvodi kroz jasno definirane korake, kao što su definiranje usluge, frizera, vremena i datuma.

Primjer implementacije:

- Director: TerminDirector

- Builder: ITerminBuilder
- Concrete Builder: KonkretniTerminBuilder
- Product: Termin

Potencijalna primjena drugih kreacijskih paterna:

- **Singleton Pattern**: Može se koristiti za klasu koja upravlja konekcijom na bazu podataka ili centralnim upravljanjem konfiguracijama, gdje je potreban samo jedan instanca objekta kroz cijeli sistem.
- **Prototype Pattern**: Prikladan je za klasu „Termin“, gdje postoji potreba za kreiranjem velikog broja sličnih objekata termina, a gdje bi se smanjila količina redundantnog koda kloniranjem postojećih objekata.
- **Abstract Factory Pattern**: Može se implementirati za upravljanje različitim grupama povezanih objekata, kao što su različiti setovi UI komponenti za web aplikaciju ili mobilnu aplikaciju.

Zaključak:
Uvođenjem Factory i Builder kreacijskih paterna značajno poboljšavamo strukturu, modularnost i skalabilnost sistema BookMyStyle. Potencijalnim korištenjem Singleton, Prototype i Abstract Factory paterna dodatno se povećava fleksibilnost za buduće proširenje funkcionalnosti te se olakšava održavanje koda.