Kreacijski paterni

Uvod

Sistem Mediplan sadrži entitete poput korisnika, termina, usluga i nalaza. Radi bolje skalabilnosti,

Održavanja i organizacije koda, uvedeni su kreacijski paterni – obrasci za kontrolisano i fleksibilno kreiranje objekata.

U ovom dokumentu su predstavljeni paterni: Singleton, Prototype, Factory Method, Abstract Factory i Builder.

NAPOMENA: U našem projektu implementirale smo samo Factory Method i Builder paterne, dok su ostali

Factory Method

Definiše interfejs za kreiranje objekta, a podklase određuju koju konkretnu klasu instancirati.

Naša implementacija:

KorisnikFactory – kreira različite tipove korisnika (Pacijent, Doktor) na osnovu unesenog tipa, bez potrebe za if-else logikom.

Kao dio implementacije, kreiran je interfejs 'KorisnikCreator' sa metodom 'KreirajKorisnika()', te konkretne klase 'PacijentKreator' i 'DoktorKreator' koje implementiraju ovu metodu za stvaranje specifičnih korisnika.

Prednosti: Fleksibilno dodavanje novih tipova objekata bez izmjene postojećeg koda, lakše testiranje i čitanje koda.

Builder

Patern koji omogućava postepeno kreiranje složenih objekata kroz niz metoda.

Naša implementacija:

U našem projektu implementirale smo klasu TerminBuilder, označenu kao <<builder>> u UML dijagramu. Ova klasa omogućava postepenu izgradnju objekta tipa Termin, uz sljedeće metode:

zaPacijenta(pacijentId) – postavlja pacijenta

kodDoktora(doktorId) – postavlja doktora

saDatumom(datum) – određuje datum termina

saLokacijom(lokacija) – dodaje lokaciju

saUslugama(listaUsluga) – dodaje medicinske usluge

build() – završava gradnju i vraća gotov Termin objekat

Ovaj patern omogućava korisnicima da unesu samo one podatke koje trenutno imaju, a ostale da dodaju naknadno. To je posebno korisno u formama i API pozivima gdje se informacije prikupljaju u koracima.

Prednosti: Veća fleksibilnost pri kreiranju objekata, bolja čitljivost koda, lakše testiranje i održavanje,prikladno za rad sa djelimičnim ili složenim podacima

Ostali paterni (opis radi konteksta)

Singleton

Ovaj patern osigurava da određena klasa ima samo jednu instancu i omogućava globalni pristup toj instanci. Koristan je za centralne resurse kao što su baze podataka, email servisi ili rasporedi termina.

Prototype

Omogućava brzo kloniranje postojećih objekata. Koristan je kada imamo šablonske objekte koji se često ponavljaju, poput medicinskih paketa ili termina. Umjesto da svaki put pravimo sve iznova, kloniramo već pripremljen prototip i prilagodimo ga po potrebi.

Abstract Factory

Omogućava kreiranje povezanih grupa objekata bez navođenja konkretnih klasa. Na primjer, jedna fabrika može praviti sve potrebne objekte za određeni tip klinike – termine, nalaze i specifične usluge. To omogućava jednostavnu zamjenu cjelokupne funkcionalnosti kada se mijenja odjel klinike ili tip usluga.