

KREACIJSKI PATERNI

U nastavku ovog dokumenta biti će analizirano pet različitih kreacijskih paterna:

1. Singleton patern - Uloga Singleton paterna je da osigura da se klasa može instancirati samo jednom i da osigura globalni pristup kreiranoj instanci klase.

U našem sistemu moguća primjena Singleton paterna je prilikom kontrole brojeva zaraženih, oporavljenih, umrlih, testiranih, itd... To jeste, klasa Statistika treba biti instancirana samo jednom i svakako da je potreban globalni pristup kreiranoj instanci klase Statistika.

2. Prototype patern - Uloga Prototype paterna je da kreira nove objekte klonirajući jednu od postojećih prototip instanci (postojeći objekat).

Prototype patern je često koristan i prilikom višestrukog korištenja dobavljenih podataka iz baze.

Na primjer, nakon ispunjenog zahtjeva za vakcinaciju, isti se pohranjuje. Ukoliko nakon određenog vremenskog perioda bude potrebno se ponovo vakcinisati, Korisnik ne mora popunjavati novi zahtjev već može klonirati prethodno ispunjeni zahtjev sa izmjenjenim datumom.

3. Factory method patern - Uloga Factory Method paterna je da omogući kreiranje objekata na način da podklase odluče koju klasu instancirati.

Na primjer, u našem u sistemu bismo mogli iskoristiti Factory method patern prilikom biranja simptoma. Tu bi se nalazile dvije podklase Teža Simptomatika i Lakša simptomatika, koja bi bila instancirana odlučuje FactoryMethod() koja bi na osnovu odabranih simptoma odlučivala da li je riječ o težim ili lakšim simptomima. Takav pogled na simptome olakšavao bi dalje koordiniranje doktora sa pacijentima.

4. Abstract Factory patern - omogućava da se kreiraju familije povezanih objekata/produkata.

Trenutno u našem sistemu ne postoji prevelika potreba za Abstract Factory paternom. Ono nad čim bi se mogao primjeniti patern je klasa Vakcina i vrste vakcina, međutim, ubacivanje ovog paterna zahtjevalo bi nekoliko većih izmjena u našoj konkretnoj klasi Vakcina.

5. Builder patern - Uloga Builder paterna je odvajanje specifikacije kompleksnih objekata od njihove stvarne konstrukcije.

Na primjer, mogli bismo dodati IKorisnikBuilder i u njemu različite metode koje bi pomagale pri analizi broja pacijenata kao što su npr. dajŽenskeMladeKorisnike(do 20 godina), dajMuškeSrednjeKorisnike(20-60 godina), itd... Metode bi zapravo vraćale liste Korisnika izdvojenih po zadatom kriteriju To bi zahtjevalo kreiranje klase kao što su ŽenskiMladiKorisnikBuilder, MuškiSrednjiKorisnikBuilder koje bi sadržavale liste svih Korisnika u sistemu.

