Za implementaciju smo odabrali kreacijske paterne singleton i builder.

Singleton

Osigurava da neka klasa ima samo jednu instancu u sistemu, i pruža globalnu tačku pristupa toj instanci

Potreba u našem sistemu?

Svi dijelovi sistema koriste istu konekciju prema bazi i trebaju izvršavati upite nad istom bazom, pa baza treba koristiti Singleton pattern da ne bi koristili dodatne resurse i komplikovali sistem.

Prototype

Kreira nove objekte kloniranjem postojećih, umjesto korištenjem konstruktora. Time možemo kreirati objekte bez da poznajemo detalje oko kreiranja i smanjiti broj klasa u sistemu na minimum.

Potreba u našem sistemu?

U našem sistemu bi se mogao koristiti prototype pattern kod višestruke analize dobavljenih podataka iz baze, i onda možemo klonirati objekat za analizu i koristiti ga ponovo. Također možemo koristiti prototype da brzo kloniramo slične objekte klase Aktivnost jer između objekata te klase su često samo potrebne minimalne izmjene parametara.

Factory Method

Definiše interfejs za kreiranje objekta, ali prepušta podklasama da odluče koji konkretni objekat će instancirati.

Potreba u našem sistemu?

Factory method pattern bi mogao dosta poboljšati kod, kada bi umjesto provjerave podataka na mjestima gdje se zahtjeva konkretna korisnik podklasa (roditelj,vaspitač,administrator) imali KorisnikFactory koja instancira odgovarajuću podklasu i izbjegava razne uslove kroz kod i centralizuje provjeru.

Abstract Factory

Omogućava nam da kreiramo familije povezanih objekata bez da navodimo konkretne klase.

Potreba u našem sistemu?

U našem sistemu ne prodajemo nikakve različite proizvode i nemamo grupe povezanih objekata koje treba organizovati. Kada bi proširili naš sistem, mogli bi grupisati evaluacijske forme za različite godišta koje konstruišemo na različite načine u zavisnosti od godišta. Preko abstract factory patterna bi napravili set tih formi i učinili kod lakšim za održavanje.

Builder

Razdvaja konstrukciju kompleksnog objekta od njegove reprezentacije tj. omogućava pravljenje objekta korak po korak.

Potreba u našem sistemu?

Izvještaji o djetetu se kreiraju iz dosta komponenti: prisustvo, aktivnosti, komentari i napomene. Builder pattern pomaže da se taj izvještaj gradi fleksibilno i modularno. Također bi mogli razdvojiti dodavanja korisnika i dijeteta u RoditeljBuilder