**SINGLETON PATTERN**

**Opis:** Singleton patern osigurava da određena klasa ima samo jednu instancu i pruža globalnu tačku pristupa toj instanci. Pogodan je za komponente koje centralizovano upravljaju resursima.

**Primjena u našem sistemu:**  
Klasa NotifikacijaController koristi se za slanje i upravljanje notifikacijama korisnicima. Kako bi se spriječilo višestruko kreiranje kontrolera (što bi moglo dovesti do duplih obavijesti), NotifikacijaController treba da implementira Singleton patern.

**Prednosti:**

* Centralizovano upravljanje slanjem notifikacija
* Jednostavan pristup iz više komponenti sistema
* Eliminisano neželjeno dupliranje

**PROTOTYPE PATTERN**

**Opis:** Omogućava kloniranje postojećih objekata i modifikovanje bez kreiranja od nule.

**Primjena u našem sistemu:**  
Kod generisanja izvještaja (Izvjestaj), moguće je da profesori često generišu slične izvještaje za studente sa samo minimalnim izmjenama. Umjesto ručnog unosa, može se klonirati prethodni izvještaj i promijeniti samo sadržaj.

**Prednosti:**

* Ušteda vremena pri ponavljanju sličnih operacija
* Manje grešaka jer se polazi od ispravnog šablona

**FACTORY PATTERN**

**Opis:** Factory Method patern omogućava podklasama da odluče koju klasu će instancirati.

**Primjena u našem sistemu:**  
Kada se kreira objekat Korisnik, na osnovu njegove uloge (Student, Profesor, Dekan, Gost), sistem treba da instancira odgovarajući podtip korisnika.

**Prednosti:**

* Fleksibilno dodavanje novih korisničkih uloga
* Klijent ne mora znati detalje instanciranja

**ABSTRACT FACTORY PATTERN**

**Opis:** Omogućava kreiranje čitave porodice povezanih objekata bez poznavanja konkretnih klasa.

**Primjena u našem sistemu:**  
Za podršku različitih tipova obavještenja (npr. putem e-maila ili u okviru aplikacije), možemo koristiti Abstract Factory koja generiše odgovarajuće instance Notifikacija, KanalSlanja, SablonPoruke.

**Prednosti:**

* Jednostavno dodavanje novih tipova notifikacija
* Jedinstveno kreiranje kompletnog sistema za slanje obavještenja

**BUILDER PATTERN**

**Opis:** Omogućava postepeno konstruisanje kompleksnih objekata, bez potrebe za korištenjem dugih konstruktora.

**Primjena u našem sistemu:**  
Kreiranje ispita (Ispit) može zahtijevati više parametara: predmet, datum, tip ispita itd. Builder omogućava fleksibilno i čitko kreiranje ispita.

**Prednosti:**

* Bolja čitljivost koda
* Lahka validacija i postepena izgradnja objekta