# DIZAJN PATERNI PONAŠANJA

## Strategy

Strategy dizajn patern izdvaja algoritam iz matične klase i uključuje ga u posebne klase, te omogućava klijentu izbor jednog od algoritama iz familije algoritama. U pogledu našeg sistema, funkcionalnost koja podrazumijeva pronalaženje odgovarajuće korisničke statistike može biti implementirana na više načina.  
To možemo implementirati:  
✓ deklaracijom Strategy interfejsa;  
✓ izdvajanjem algoritama u njihove klase;  
✓ dodavanjem reference na objekat strategije i setera za izmjenu tog polja.  
Korištenje:  
Strategy se koristi za odabir algoritma čiji je cilj izdvajanje statistike za korisnika u ovisnosti od njegove uloge – nastavnik može vidjeti detalje po predmetima, roditelj ukupan uspjeh, a učenik samo vlastite ocjene.

## State

Kod State dizajn paterna, objekat mijenja način ponašanja na osnovu trenutnog stanja. U sistemu eDnevnik, status prisutnosti učenika na času može se posmatrati kao stanje: prisutan ili odsutan. Prisutan učenik može biti ocijenjen, dok odsutan ne može.  
Implementacija može uključivati:  
✓ kreiranje interfejsa StanjePrisustva;  
✓ konkretne klase Prisutan i Odsutan;  
✓ kontekstualna klasa Ucenik koja mijenja stanje na osnovu prisustva.  
Ova primjena, iako jednostavna, doprinosi jasnijem upravljanju pravima nad funkcijama ocjenjivanja.

## Template Method

Template Method omogućava izdvajanje određenih koraka algoritma u odvojene podklase. U našem slučaju koristi se za kreiranje izvještaja o ocjenama i izostancima, pri čemu se određeni koraci poput izračunavanja prosjeka, broja izostanaka i prijedloga ocjena definišu u glavnoj klasi.  
To možemo implementirati:  
✓ kreiranjem apstraktne bazne klase IzvjestajGenerator;  
✓ implementiranjem metode koja piše izvještaj na osnovu izdvojenih koraka;  
✓ definisanjem skupa apstraktnih metoda koje predstavljaju korake algoritma.  
Korištenje:  
Razrednik može generisati izvještaje za sve učenike razreda, koje zatim dijeli roditeljima putem e-maila ili u štampanoj formi.

## Observer

Observer dizajn patern uspostavlja relaciju između objekata tako da kada jedan objekat promijeni stanje, drugi zainteresovani objekti se automatski obavještavaju. U eDnevniku se može koristiti kod ocjenjivanja – kada nastavnik unese ocjenu, razrednik dobije notifikaciju.  
Korištenje može biti opcionalno – razrednik bira da li želi notifikacije.  
Ovaj patern je posebno koristan u slučaju proširenja sistema za podršku real-time obavještenjima.

## Iterator

Iterator patern omogućava pristup elementima kolekcije bez poznavanja njene interne strukture. U okviru sistema eDnevnik, koristi se za prikaz kolekcija: učenika razreda, ocjena, izostanaka, časova itd.  
Implementacija može uključivati:  
✓ deklaraciju Iterator i Collection interfejsa;  
✓ konkretne klase iteratora za različite entitete (ocjene, učenici...);  
✓ metodu za dohvat iteratora.  
Korištenje:  
Omogućava nastavniku da pregleda samo ocjene jednog učenika bez direktnog pristupa cijeloj strukturi razreda.

## Command

Command patern se koristi za enkapsuliranje svih informacija potrebnih za izvršavanje određene komande. Primjena u eDnevniku je kod višekorakog unosa učenika od strane administratora – gdje se svaki korak predstavlja kao zasebna komanda.  
Komponente:  
✓ Command interfejs sa metodom execute();  
✓ AddStudentCommand kao konkretna implementacija;  
✓ Receiver koji sadrži metodu addStudent();  
✓ Invoker koji upravlja izvršenjem komandi.  
Korištenje:  
Administrator može izvršiti više komandi za dodavanje učenika, a sistem može podržavati 'undo' ukoliko je to potrebno.

## Memento

Memento patern omogućava spremanje i vraćanje prethodnog stanja objekta. U kontekstu eDnevnika, koristi se kod uređivanja izvještaja – razrednik može napraviti više verzija prije slanja i vratiti se na prethodnu ako je potrebno.  
Komponente:  
✓ Originator: klasa Izvještaj;  
✓ Memento: klasa IzvještajMemento;  
✓ Caretaker: upravlja historijom verzija.  
Korištenje:  
Omogućava fleksibilnije uređivanje izvještaja i prevenciju gubitka informacija.

## Mediator

Mediator centralizuje komunikaciju između objekata. U našem slučaju koristi se u okviru globalnog chata – korisnici ne komuniciraju direktno jedni s drugima već poruke prolaze kroz posrednika.  
Komponente:  
✓ Interfejs ChatMediator;  
✓ Konkretna klasa GlobalniChat;  
✓ Colleague objekti: Korisnik, Učenik, Nastavnik, Roditelj, Admin.  
Korištenje:  
Pojednostavljuje sistem poruka, posebno kada broj učesnika raste. Takođe omogućava logiku filtriranja poruka i skalabilnost sistema.