**PRIMJENA STRUKTURALNIH PATERNA**

**Implementirani paterni u dijagramu klasa:**

**Facade Pattern – Jednostavniji pristup podsistemima**

Sistem ima više povezanih klasa kao što su Mec, Recenzija, Korisnik, Oglas. Trenutno klijent mora ručno upravljati međusobnim interakcijama.

**Rješenje:** Uvođenjem klase MatchleticFacade, možemo objediniti pristup najčešće korištenim operacijama kao što su:

* Kreiranje meča između korisnika na osnovu oglasa
* Dodavanje recenzije
* Prikaz svih oglasa korisnika

**S time smo smanjili kompleksnost za klijenta i poboljšali čitljivost.**

**Composite patern – Proširenje klase recenzija**

Pri razvijanju sistema klasa Recenzija, koja predstavlja ocjenu korisnika o nekom meču mogla bi izvršavati sljedeće stavke:

* Grupisanje više recenzija (npr. sve recenzije jednog korisnika, sve za jedan oglas ili meč).
* Izvršenje neke operacije nad cijelom grupom (npr. prikaz, brisanje, filtriranje).
* Tretiranje pojedinačne i grupisane recenzije na isti način.

**Ostali paterni:**

**Adapter Pattern**

Prilikom integracije sa eksternim sistemom za notifikacije (npr. email, SMS).

**Primjer:** Postojeći sistem koristi internu klasu Notifikacija, ali novi klijent koristi vanjsku biblioteku sa drugačijim interfejsom.

**Proxy Pattern**

Za kontrolu pristupa osjetljivim podacima (npr. samo admin može vidjeti sve recenzije).

**Primjer:** RecenzijaProxy provjerava da li korisnik ima pravo pristupa prije delegiranja na Recenzija.

**Composite Pattern**

**Primjena:** Postoji razlika između običnog korisnika i administratora (Uloga), ali može doći potreba za dodavanjem dodatnog ponašanja korisnicima (npr. super admin, verifikovani korisnik, itd.).

**Bridge Pattern**

**Primjena:** Za odvajanje logike prikaza rezultata (rezultat) i implementacije prikaza (tekstualno, grafički).