Design Pattern

1. Princip pojedinačne odgovornosti

Ovaj design patern sluzi da predstavljamo neki tip podataka drugacije u odnosu kako je sacuvano. U nasoj aplikaciji adapter pattern bi se mogao iskoristit tako da prikazujemo listu terena po nekom filteru, u zavisnosti od adaptera mogli bi prikazivati sve terene ili po nekom odredjenom filter

2. Facade patern

Facade patern se koristi kada sistem ima više identificiranih podsistema (subsystems) pri čemu su apstrakcije i implementacije podsistema usko povezane. U nasoj aplikaciji ovaj patern smo primjenili za editovanje terena od strane vlasnika terena I admina jer editovanje zapravo predstavlja usko povezane podsisteme npr. (editovanje naziva, editovanje lokacije, editovanje opisa, editovanje slika…) Svi ovi podsistemi se mogu zasebno raditi a usko su povezani jedni sa drugima

3. Decorator patern

Osnovna namjena Decorator paterna je da omogući dinamičko dodavanje novih elemenata i ponašanja (funkcionalnosti) postojećim objektima. U nasem projektu obaj patern bi mogli iskoriti pri kreiranju terena, dodavanje broja sala, lokacije, sportova koji se mogu odrzavati na terenu/sali.

4. Brigde patern

Osnovna namjena Bridge paterna je da omogući odvajanje apstrakcije i implementacije neke klase tako da ta klasa može posjedovati više različitih apstrakcija i više različitih implementacija za pojedine apstrakcije. U nasoj aplikaciji bridge pattern mozemo koristi pri racunanju cijene za tezervaciju u ovisnosti od Terena i termina

5. Proxy patern

Namjena Proxy paterna je da omogući pristup i kontrolu pristupa stvarnim objektima. Proxy je obično mali javni surogat objekat koji predstavlja kompleksni objekat čija aktivizacija se postiže na osnovu postavljenih pravila. U nasoj aplikaciji proxy patern smo implementirali za provjeravanje stanja transakcije, odnosno da pri rezervaciji korisnik ne moze rucno postaviti stanje transakcije da bude gotovo vec mora proci kroz proxy koji ce provjeriti tu verifikaciju

6. Composite patern

Osnovna namjena Composite paterna (kompozitni patern) je da omogući formiranje strukture stabla pomoću klasa, u kojoj se individualni objekti (listovi stabla) i kompozicije individualnih objekata (korijeni stabla) jednako tretiraju. U nasoj aplikaciji ovaj patern smo mogli iskoriti za dodavanje odnosno mijenjanje temina za odredjeni teren, gdje nam list predstavlja jedna isntanca termina dok nam je korijen lista slobodnih termina za dati teren.

7. Flyweight patern

Postoje situacije u kojima je potrebno da se omogući razlikovanje dijela klase koji je uvijek isti za sve određene objekte te klase (tzv. glavno stanje (engl. intrinsic state)) od dijela klase koji nije uvijek isti za sve određene objekte te klase (tzv. sporedno stanje (engl. extrinsic state)). Osnovna namjena Flyweight paterna je upravo da se omogući da više različitih objekata dijele isto glavno stanje, a imaju različito sporedno stanje. U nasoj aplikaciji ovaj patern se moze iskoristiti za prikaz filtriranih terena, filter po kojem se sortira moze biti po sportu ili po lokaciji.