Strukturni Dizajn Paterni

1 Adapter

Adapter je obrazac dizajna koji omogućava saradnju izmeu različitih interfejsa. U našem slučaju, možemo koristiti Adapter za integraciju vanjskih servisa za autentifikaciju korisnika.

- Kreiranje interfejsa `AuthenticationServiceAdapter` koji će definirati metode za autentifikaciju korisnika u našem sistemu.
- Implementacija konkretnog Adaptera `ExternalAuthenticationServiceAdapter` koji će prilagoditi interfejs vanjskog servisa za autentifikaciju na interfejs definiran unutar sistema.
- Unutar konkretnog Adaptera možemo implementirati metode autentifikacije korisnika prema API-ju vanjskog servisa.

Korištenje: Unutar sistema možemo koristiti Adapter za autentifikaciju korisnika umjesto direktnog poziva vanjskog servisa. Na taj način, naš sistem će biti nezavisan o konkretnim implementacijama vanjskih servisa, olakšavajući buduće promjene i održavanje.

2 Façade

Façade obrazac može se primijeniti kako bi se stvorio jednostavan interfejs za prikaz ocjena i evidencije prisustva za nastavnike, učenike i roditelje. Ovaj interfejs će sakriti složenost pozadinskih podsistema i omogućiti korisnicima jednostavan pristup tim informacijama.

3 Decorator

Decorator je obrazac dizajna koji omogućava dinamičko dodavanje ili promjenu funkcionalnosti objektima. Možemo koristiti Decorator za prilagodbu različitih funkcionalnosti prema korisničkim ulogama.

- Definiranje interfejsa `UserFunctionality` koji predstavlja osnovnu funkcionalnost korisnika.

- Implementacija `BasicUserFunctionality` koji pruža osnovne funkcionalnosti korisnika.
- Kreiranje interfejsa `UserFunctionalityDecorator` koji će definirati metode za dodatne funkcionalnosti.

Korištenje: Na primjer, prilikom prijave korisnika, možemo dinamički dodijeliti odgovarajući dekorator prema korisničkoj ulozi kako bi se omogućile specifične funkcionalnosti. Ovim pristupom omogućujemo fleksibilnost sistema, omogućavajući lako dodavanje ili promjenu funkcionalnosti prema potrebama korisnika.

4 Bridge

Bridge obrazac omogućava odvajanje apstrakcije (korisnički interfejs) od implementacije (poslovna logika) tako što uvodi posebne klase koje povezuju ove dvije komponente. Implementiranjem Bridge obrasca možemo stvoriti fleksibilan sistem u kojem se korisnički interfejsi (npr. prikaz ocjena) neće direktno oslanjati na poslovnu logiku (npr. izračunavanje ocjena). Umjesto toga, koristit će se mostovi koji će povezivati ove dvije komponente.

5 Composite

Naš sistem e-dnevnika sadrži složenu strukturu podataka koja uključuje različite entitete poput korisnika, predmeta, ocjena, prisustva i ostalog. Postoje hijerarhijski odnosi izmeu entiteta, na primjer, učenici su grupirani u razrede, a predmeti su dodijeljeni nastavnicima. Composite obrazac omogućava grupiranje objekata u hijerarhijske strukture kako bi se omogućilo korištenje pojednostavljenog interfejsa za rad sa složenim skupinama objekata.

Korištenje: Implementiranjem Composite obrasca možemo stvoriti strukturu entiteta u našem sistemu tako da svaki entitet bude kompozitni objekt koji može sadržavati druge entitete ili pojedinačne objekte. Na primjer, razred može sadržavati učenike kao podobjekte, a nastavnik može imati više predmeta kao podobjekte.

6 Proxy

Implementacija Proxy obrasca omogućava zaštitu osjetljivih podataka poput korisničkih informacija ili ocjena kako bi se osigurala njihova privatnost i integritet. Uvoenjem Proxy obrasca možemo provjeravati ovlaštenje korisnika prije nego što im se omogući pristup odreenim funkcionalnostima ili podacima. Korištenje Proxy obrasca se takoer može iskoristiti za keširanje podataka kako bi se smanjilo vrijeme odziva sistema i poboljšala ukupna izvedba.

7 Flyweight

Naš sistem e-dnevnika sadrži velike količine podataka kao što su ocjene, prisustvo, i informacije o korisnicima, što može dovesti do preopterećenja sistema. Korištenje Flyweight obrasca omogućava efikasnije upravljanje memorijom tako što se zajednički dijeljeni objekti koriste za smanjenje dupliranja podataka. Korisnički podaci poput imena, prezimena i kontaktnih informacija mogu se često dijeliti izmeu različitih dijelova sistema umjesto da se svaki put dupliciraju. Informacije o predmetima kao što su naziv, opis i broj časova takoer se mogu dijeliti izmeu različitih dijelova sistema umjesto da se svaki put ponavljaju.