Strukturalni paterni

1. Adapter pattern

Adapter pattern moguće je iskoristiti na sljedeće načine:

Integracija sa platnim gatewayima:
Naša web aplikacija podržava više platnih gatewaya (npr. PayPal, kartično, pri pouzeću) svaki gateway može imati vlastiti interfejs za obradu plaćanja. Možemo stvoriti adapter klase koje implementiraju zajednički interfejs za plaćanje i prilagoditi specifična interfejs platnih gatewaya tom zajedničkom interfejsu.
Obrada slika:
Naša web aplikacija može zahtijevati mogućnosti obrade slika, poput promjene veličine, obrezivanja. Različite biblioteke za obradu slika mogu imati vlastiti interfejs. Stvaranjem adaptera za obradu slika možemo zapakirati funkcionalnosti određene biblioteke za obradu slika i izložiti dosljedan interfejs koji naša web aplikacija može koristiti, bez obzira ba temeljnu biblioteku.
Pretvorba veličina:
Veličine odjeće mogu varirati ovisno o regijama ili markama, što može izazvati zbunjenost kod kupaca. Možemo stvoriti adapter koji pretvara veličine iz jednog standarda u drugi.

2. Facade pattern

Facade pattern moguće je iskoristiti na sljedeći način:

Obrada narudžbi:
Naša web aplikacija uključuje složen proces obrade narudžbi koji uključuje zadatke poput upravljanja inventarom, obrade plaćanja, koordinacije dostave i slanja e-mail-a. Umjesto da izlažemo detalje i složenosti tih podsistema, možemo stvoriti fasadu koja obuhvata cijelu logiku obrade narudžbi. Fasada će pružiti pojednostavljeni interfejs koji klijenti mogu koristiti za slanje narudžbi, a unutar sebe će upravljati svim potrebnim koracima obrade narudžbi i koordinacijom uključenih podsistema.

3. Decorator pattern

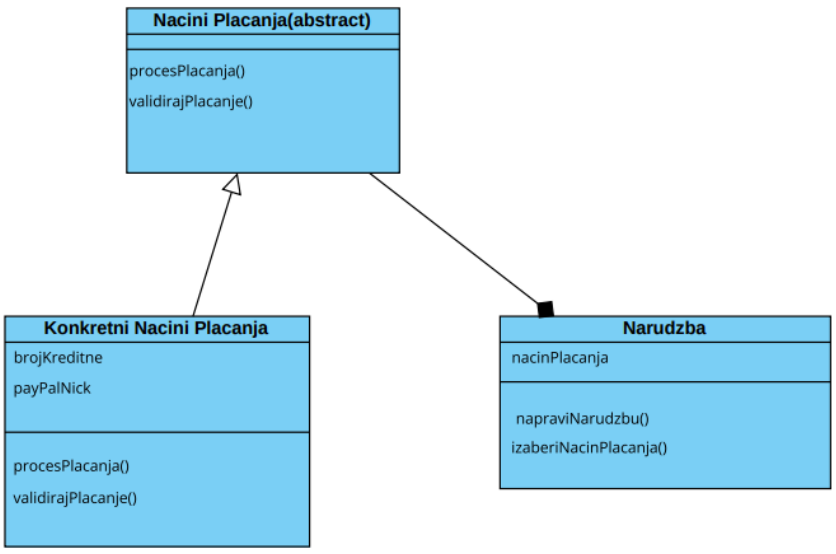
Decorator pattern moguće je iskoristiti na sljedeće načine:

Dodatne karakteristike proizvoda:
Decorator pattern možemo koristiti kako bismo dodali dodatne karakteristike ili opcije za proizvode u našoj web aplikaciji. Možemo stvoriti decoratore za proizvode koji dodaju personalizirane opcije poput odabira boje ili mogućnosti prilagođenog printa na odjeći. Dekoratori bi omotali osnovne objekte proizvoda i proširili njihove mogućnosti, pružajući korisnicima veći izbor i prilagodbu proizvoda.
Prikaz dodatnih informacija:
Decorator pattern možemo koristiti kako bismo dodali dodatne informacije o proizvodima prilikom prikaza. Možemo stvoriti decoratore koji pružaju informacije o sastavu materijala, njegovim karakteristikama ili preporukama za održavanje.
Obrada narudžbi:
Decorator pattern možemo koristiti za dodavanje posebnih zahtjeva ili opcija dostave. Možemo stvoriti decoratore koji pružaju mogućnost odabira brze dostave, praćenja pošiljke ili specifičnih zahtjeva za pakovanje.

4. Bridge pattern

Bridge pattern moguće je iskoristiti na sljedeć način:

Različiti načini plaćanja:
Naša web aplikacija podržava različite načine plaćanja, poput kreditne kartice, PayPal-a ili putem pouzeća. Bridge pattern možemo primijeniti kako bi se odvojila apstrakcija načina plaćanja od konkretnih implementacija. Možemo imati različite implementacije bridge-a koje nasljeđuju klasu "NaciniPlacanja" i implementiraju specifične načine plaćanja.



5. Composite pattern

Composite pattern moguće je iskoristiti na sljedeći način:

Filtriranje proizvoda:
Naša web aplikacija podržava filtriranje proizvoda prema različitim kriterijima, poput boje, veličine ili cijene. Composite pattern možemo koristiti za implementaciju filtera za proizvode. Možemo imati apstraktnu klasu "Filter" koja definiše osnovne metode filtriranja. Zatim, možemo imati implementaciju composite-a koja nasljeđuje tu klasu i predstavlja složeni filter. Ova implementacija može sadržavati podfiltere kao svoje komponente. Na primjer, klasa "SlozeniFilter" može naslijediti "Filter" i sadržavati podfiltere poput "BojaFilter", "VelicinaFilter", "CijenaFilter". Kada se primijeni "SlozeniFilter", on će izvršiti filtriranje koristeći kombinaciju podfiltera kako bi pružio rezultate koji zadovoljavaju sve specifirane kriterije.

6. Proxy pattern

Proxy pattern moguće je iskoristiti na sljedeći način:

Zaštita pristupa:
Proxy pattern možemo koristiti kako bismo implementirali mehanizam zaštite pristupa određenim resursima. Administratorske funkcionalnosti su dostupne samo administratoru, Proxy može kontrolisati pristup tim funkcionalnostima. Proxy će provjeriti autentifikaciju i ovlasti korisnika prije nego što omogući pristup administratorskim funkcijama. Ovo pomaže u zaštiti od neovlaštenog pristupa i održavanju sigurnosti u aplikaciji.

7. Flyweight pattern

Flyweight pattern moguće je iskoristiti na sljedeći način:

Upravljanje veličinama odjeće:
Naša web aplikacija ima različite veličine odjeće koje se koriste u više proizvoda. Flyweight pattern možemo koristiti kako bismo optimizirali upravljanje veličinama odjeće. Možemo stvoriti Flyweight objekte koji predstavljaju jedinstvene veličine i njihove atribute. Ovi objekti se mogu dijeliti među različitim proizvodima koji koriste istu veličinu. Na taj način, umjesto da se za svaki proizvod stvara nova instanca veličine, koriste se već postojeći Flyweight objekt što smanjuje potrošnju memorije i olakšava upravljanje veličinama.