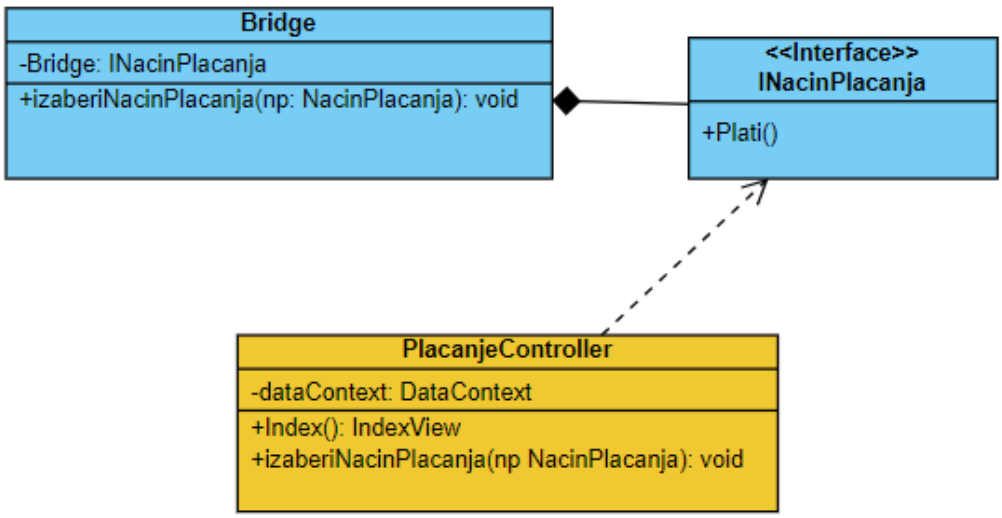
Strukturalni paterni

Adapter pattern

Adapter pattern bi se mogao iskoristiti u sortiranju proizvoda ako nekad budemo dodavali drugačije načine soritranja. Trenutno je osmišljeno da postoji Tip sortiranja - highest price first, lowest price first itd i metoda Sort prima taj tip sortiranja. Ako bi nekad uveli drugačije sortiranje koje ne prima ni jedan Tip sortiranja nego nešto drugo onda bi konverziju umjesto mijenjanja postojećih metoda mogli uraditi pomoću ovog pattern-a.

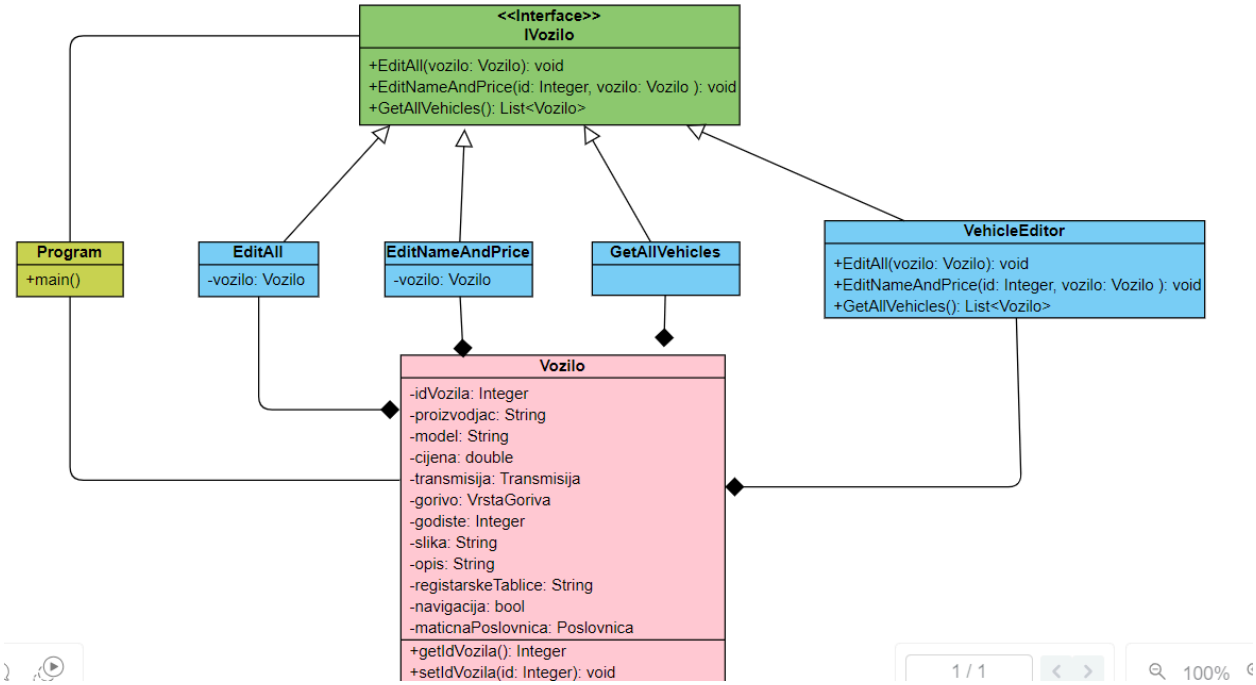
Bridge pattern

Bridge pattern moguće je iskoristiti na sljedeći način: Različiti načini plaćanja: Naša web aplikacija podržava različite načine plaćanja, poput kreditne kartice, u poslovnici 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



Decorator pattern

Administratorima mogućnost promjene opisa vozila (Opis, cijena, slike itd.). Promjene nad vozilima možemo podijeli u tri skupine: EditAll koja omogućava promjenu svih detalja vozila I EditNameAndPrice koja će davati uvid samo u naziv I cijenu vozila jer su to atributi koji su nam najvažniji I najviše korišteni za razne funkcionalnosti I algoritme, te metodu GetAllVehicles. Uvodimo Decorator klasu koja je povezana agregacijom s IVozilo interface-om I instancira jedan ili više IVozilo objekata.



Composite pattern

Composite pattern moguće je iskoristiti na sljedeći način:
Filtriranje vozila: Naša web aplikacija podržava filtriranje vozila prema različitim kriterijima, poput cijene, kategorije, dostupnosti, modela. Composite pattern možemo koristiti za implementaciju filtera za vozila. 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 “KategorijaFilter”, “ModelFilter”, “CijenaFilter”. Kada se primijeni “SlozeniFilter”, on će izvršiti filtriranje koristeći kombinaciju podfiltera kako bi pružio rezultate koji zadovoljavaju sve specifirane kriterije.

Facade pattern

Facade pattern bismo mogli iskoristiti prilikom implementacije validiranja kreditne kartice.Ako iskoristimo neku eksternu biblioteku za ovu svrhu ovaj pattern bi bio koristan da sakrije kompleksnost tog podsistema iza interfejsa.

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.

Flywight pattern

Flywight pattern bismo mogli implementirati u sistemu da olaksamo korisniku da dodje do zeljenog tipa vozila, na nacin da bismo koristili njegovu historiju rentanja (RentHistory) i izabrali dva ili tri modela vozila koje je on najvise puta iznajmio I vozila te vrste mu prikazali na pocetnoj stranici kao defaultne.