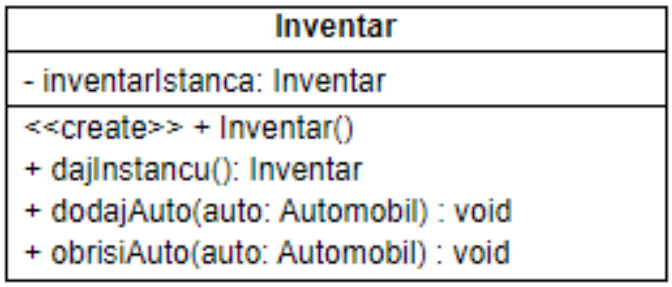
KREACIJSKI PATERNI

1. **Singleton pattern**

Uloga Singleton patterna je da osigura da se klasa može instancirati samo jednom i da osigura globalni pristup kreiranoj instanci klase. Postoji više objekata koje je potrebno samo jednom instancirati i nad kojim je potrebna jedinstvena kontrola pristupa. Instanciranje više nego jednom može prouzrokovati probleme kao što su nekorektno ponašanje programa, neadekvatno korištenje resursa ili nekonzistentan rezultat.

U našem sistemu mi smo napravili Inventar klasu. Ona omogućava administratoru da doda i ukloni automobile po volji. Ova klasa se instancira samo jednom i sve promjene koje se naprave uz pomoć ove klase, odrazit će se na cijeli sistem.



2. **Prototype pattern**

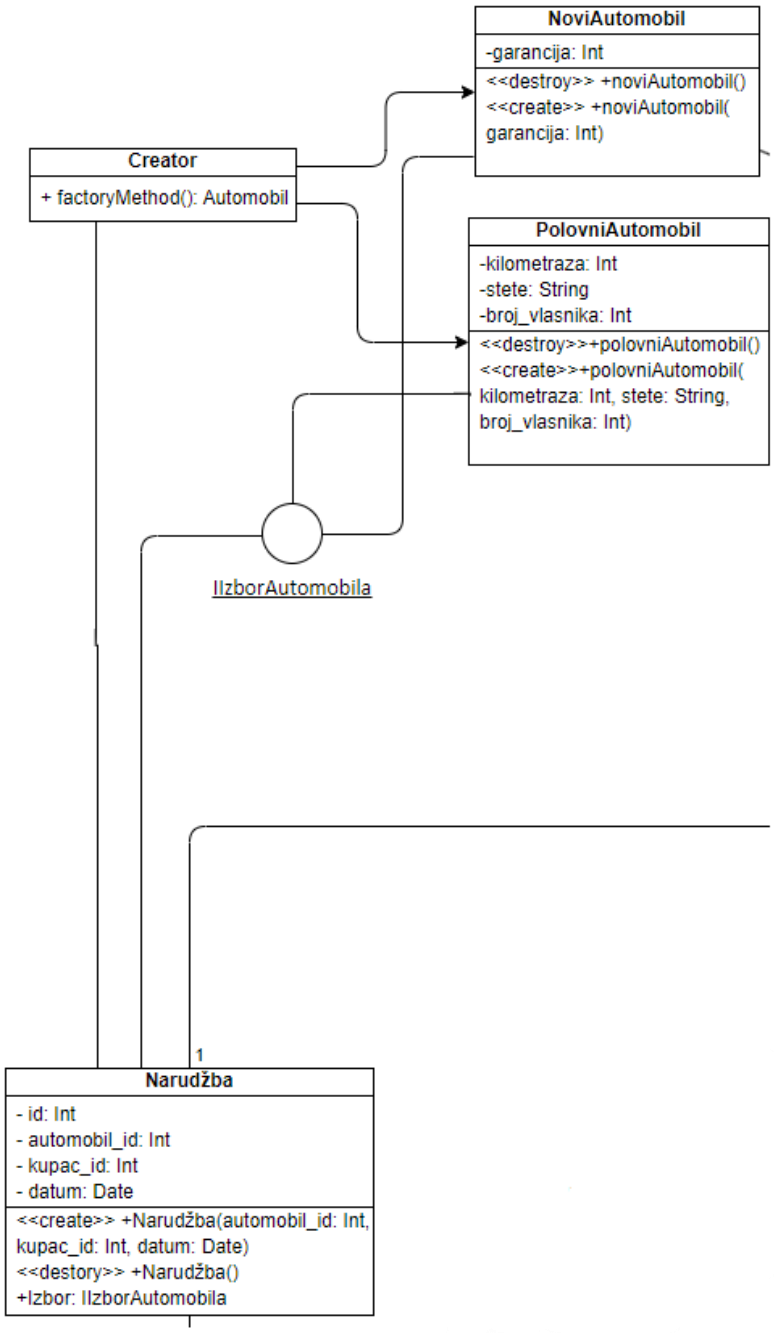
Uloga Prototype patterna je da kreira nove objekte klonirajući jednu od postojećih prototip instanci (postojeći objekat). Ako je trošak kreiranja novog objekta velik i kreiranje objekta je resursno zahtjevno tada se vrši kloniranje već postojećeg objekata. Dakle ovaj šablon omogućava jednostavnije kreiranje novih instanci klasa kod kojih je veliki broj atributa identičan za većinu instanci.

Naš sistem se već sastoji od klasa kod kojih je većina atributa za većinu instanci različita, čak u nekim slučajevima i svi su različiti, jer su u pitanju klase koje modeliraju specifične objekte, korisnike, narudžbe i zahtjeve, pa nema potrebe za kloniranjem.

3. Factory Method pattern

Uloga Factory Method patterna je da omogući kreiranje objekata na način da podklase odluče koju klasu instancirati. Različite podklase mogu na različite načine implementirati interfejs. Factory Method instancira odgovarajuću podklasu(izvedenu klasu) preko posebne metode na osnovu informacije od strane klijenta ili na osnovu tekućeg stanja.

U našem sistemu ovaj patern možemo implementirati kod naše klase Narudžba. Pomoću ovog paterna možemo kontrolisati mjenjanje stanja baze s obzirom da li naš kupac kupuje polovno ili novo auto.



4. **Abstract Factory pattern**

Abstract Factory pattern omogućava da se kreiraju familije povezanih objekata/produkata. Na osnovu apstraktne familije produkata kreiraju se konkretne fabrike (factories) produkata različitih tipova i različitih kombinacija.

U našem sistemu ovaj patern nije moguće promjeniti jer naš sistem ne posjeduje više različitih objekata od kojih se može kreirati familija.

5. **Builder pattern**

Uloga Builder patterna je odvajanje specifikacije kompleksnih objekata od njihove stvarne konstrukcije. Isti konstrukcijski proces može kreirati različite reprezentacije.

U našem sistemu već imamo implementiran ovaj patern kod naše klase Konfiguracija. Kako bi upotpunili ovaj patern mogli bismo nadodati interfejs IBuilder i Director klasu, ali to nam se ne čini nešto veoma potrebno našem sistemu.