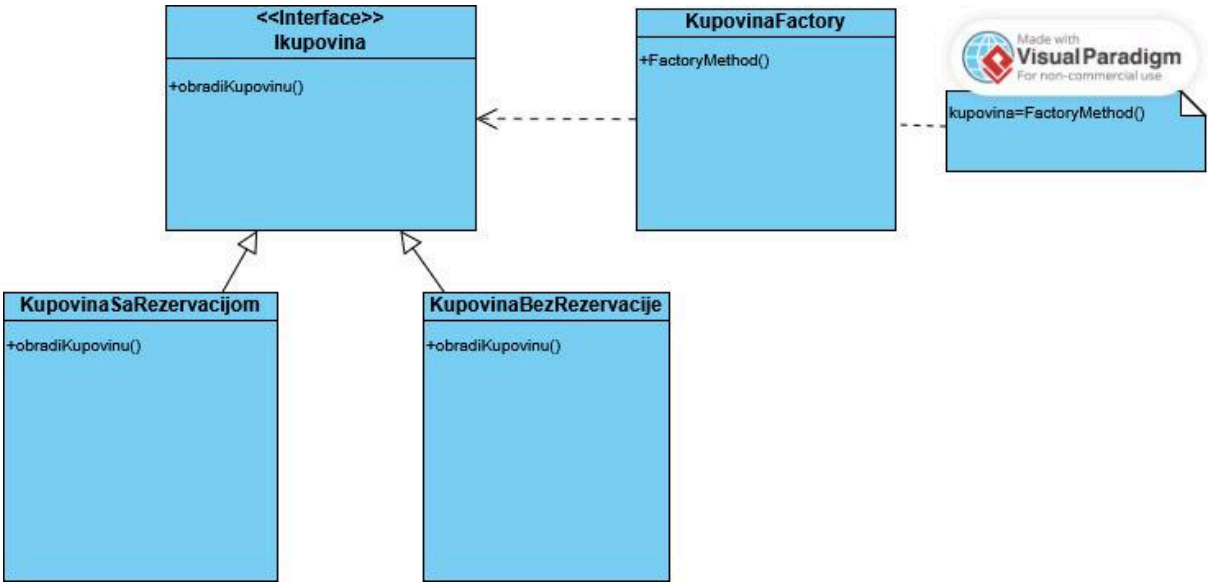
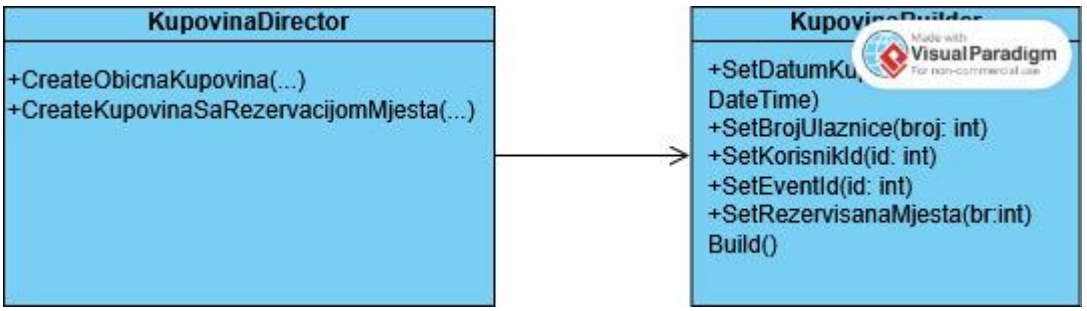
Factory Method pattern

U ovom rješenju primijenjen je Factory Method kreacioni pattern s ciljem da se odvoji logika kreiranja objekata kupovine od samog kontrolera i ostatka aplikacije. Time se postiže fleksibilnost, proširivost i slaba povezanost. Kontroler ne zna ništa o konkretnim klasama kupovine, on koristi samo KupovinaFactory i interfejs IKupovina. U KupovinaFactory imamo metodu FactoryMethod koja donosi odluku koju konkretnu klasu kupovine treba instancirati.



Builder pattern

Kupovina ima više polja i mogu postojati različiti načini kreiranja objekata (npr. sa ili bez rezervacije mjesta), te samim tim korištenje klasičnog konstruktora postaje nepregledno i podložno greškama. Zato se koristi Builder pattern koji omogućava fleksibilno i čitljivo konstruisanje objekata. Builder (KupovinaBuilder) omogućava postavljanje pojedinačnih polja objekta preko imenovanih metoda (SetDatumKupovine(...), SetBrojUlaznice(...) itd.). Završava se pozivom Build(), koji vraća potpuno kreiran objekat klase Kupovina. Omogućava različite reprezentacije istog objekta, ovisno o tome koje metode su pozvane (npr. Kupovina sa ili bez rezervisanih mjesta). Objekat Kupovina je uvijek isti sa strane njegove reprezentacije, ali način kako se do njega dolazi je odvojen i fleksibilan.



Singleton pattern

Pošto sadašnji sistem nije pogodan za implementaciju Singleton paterna, bilo bi potrebno uvesti novu klasu NotificationCenter koja bi bila zadužena za slanje podsjetnika korisnicima. Ova klasa bi se implementirala kao Singleton kako bi postojala samo jedna njena instanca u cijeloj aplikaciji. Na taj način bi se izbjeglo ponovno kreiranje objekta i uspostavljanje konekcije prema serveru pri svakom slanju podsjetnika. Umjesto toga, sve poruke bi se slale kroz jednu, centralizovanu instancu koja zadržava aktivnu vezu prema servisu za slanje e-mailova.

Prototype pattern

Prototype pattern bi se mogao primijeniti na klasu Event jer ponekad treba duplicirati već postojeći događaj (isti naziv, lokacija, opis) i zatim promijeniti samo neke atribute, poput datuma ili vremena.

Abstract Factory pattern

Abstract Factory pattern omogućava kreiranje cijele porodice međusobno povezanih objekata bez navođenja konkretnih klasa. U sadašnjem sistemu takva porodica ne postoji, pa treba uvesti novi scenarij da bi se pattern mogao primijeniti. Primjer koji se prirodno uklapa je generisanje izvještaja nakon kupovine. Svaki izvještaj se sastoji od tri dijela: zaglavlje, tijelo i podnožje. Ti dijelovi zajedno čine porodicu objekata, i za svaki tip izvještaja (HTML, PDF, PlainText) izgledaju različito. Abstract Factory je tu da omogući da se svi ti elementi zajedno kreiraju za odgovarajući tip izvještaja.