Refactoring u dizajn paterne:

* Strukturalni patern:
  + Ukoliko bi zahtjev agencije bio da samo odredjeni administratori dodaju, brisu i azuriraju ponude i putovanja, postavit cemo da jedino administrator sa odredjenim pravima moze dodavati mijenjati i brisati ponude. Tu bi bio iskoristen Proxy pattern.

Ovaj refactoring je je dobar da ne bi doslo do neusaglasenih podataka u bazi i slicno.

Nije potrebno da svaki administrator ima parvo mijenjanja baze.

* Kreacijski patern:
  + Kako citav administratorski dio aplikacije radi sa klasom agencija, u koju su pri pokretanju uneseni podaci iz baze ne bi bilo pozeljno dodavati nove instance te klase tako da je u ovom slucaju prepravljeno da se klasa agencija moze samo jednom instancirati te je ispostovan singleton patern
* Bihejvioralni patern:
  + Implementiran je observer patern koji obavjestava administratora da ce se putovanje odrzati, kada se dostigne minimalan broj putnika. Administratoru je dodan atribut obavjestenje te pri pokretanju aplikacije, nakon logovanja, obavjestenja mu se ispisuju.

Sto se tice generalnog refactoringa uradjene neke stvari su uradjene tokom izrade projekta:

* Dodavanje funkcija za akcije koje se cesto desavaju (npr. “PrikaziPoruku(string s)” koja prikazuje asinhrono sve poruke od sistema korisniku / adminsitratoru.
* Izbjegavano je izvrsavanje petlji koje nisu potrebne upotrebom predefinisanih funkcija
* Dodane su regije u kod radi bolje preglednosti citavog koda te su cjeline grupirane
* Ubaceni su dodatni try-catch blokovi radi sigurnosti rada programa odnosno radi zastite od krahiranja

Prikaz koda – primjer refactoringa koji je uradjen:





