**Adapter patern – Strukturalni patern**

Osnovna namjena Adapter paterna je da omogući širu upotrebu već postojećih klasa. U situacijama kada je potreban drugačiji interfejs već postojeće klase, a ne želimo mijenjati postojeću klasu koristi se Adapter patern. Adapter patern kreira novu adapter klasu koja služi kao posrednik između originalne klase i željenog interfejsa. Tim postupkom se dobija željena funkcionalnost bez izmjena na originalnoj klasi.

Adapter patern ustvari mapira interfejs jedne klase u drugu tako da mogu raditi zajedno.

U našem projektu moze biti iskoristen adapter patern u klasi Racun (dodavanjem jedne nove klase(adapter) i interfejsa(iTarget)). Klasa Racun ima metodu obracunajPorez, koja dodaje određenu vrijednost na iznos u zavisnosti od PDV-a. Ukoliko se zahtjeva i neki drugi, novi standard (npr. Korištenjem aplikacije u drugoj državi i sl.) može se uvesti interface koji sadrzi metodu koja vrši novo izračunavanje .Takođe je potrebno dodati i novu klasu koja bi bila nasljeđena iz klase račun i implementirala bi novi interfejs.

**Proxy patern** **– Strukturalni patern**

Namjena Proxy paterna je da omogući **pristup i kontrolu pristupa stvarnim** objektima. Proxy je obično mali javni surogat objekat koji predstavlja kompleksni objekat čija aktivizacija se postiže na osnovu postavljenih pravila. Proxy patern rješava probleme kada se objekt ne može instancirati direktno (npr. zbog restrikcije pristupa).

Mogli bismo koristiti ovaj patern za omogućavanje pristupa i kontrolu pristupa klasi za pristup svim podacima o uposlenim i korisnicima. Treba ove operacije(metode) dozvoliti samo onome ko je registrovan kao administrator.

Potrebno je napraviti klasu administrator(privatnu) i klasu ProtectionProxy(javnu) obje mogu imati istu metodu za pristup podacima. Klasa ProtectionProxy ima atribut tipa Administrator i korisnicko ime i lozinka koji trebaju biti uneseni kako bi sistem prepoznao administratora. Ova klasa takodje ima metodu Authenticate koja kao parametre prima korisničko ime i lozinku i provjerava jesu li tacni, ukoliko jesu omogucava poziv metode za pristupPodacima, tako sto instancira stvarni objekat(Administrator).

**Prototype patern – Kreacijski patern**

Uloga Prototype paterna je da kreira nove objekte klonirajući jednu od postojećih prototip instanci (postojeći objekat). Prototype dizajn patern dozvoljava da se kreiraju prilagođeni objekti bez poznavanja njihove klase ili detalja kako je objekat kreiran.

Protypes patern se koristi kada je potrebno da se sakriju konkretne klase od klijenta, dodaju ili izbrišu nove klase za vrijeme izvršavanja, da se broj klasa u sistemu održi na minimumu, kada je potrebna promjena strukture podataka za vrijeme izvršavanja

Ovaj patern moze biti iskoristen za kreiranje registrovanih firmi ukoliko je već registrovan korisnik koji želi registrovati firmu (to je moguće jer su obje nasljeđene iz klase korisnik), npr. Moguće je postojeću instancu klase korisnika klonirati (koristeći shallow kloniranje) i dodjeliti je instanci klase novog korisnika (u ovom slučaju firma) – svi podaci ostaju isti, samo se promijeni podatak o statusu i doda ime firme. Potrebno je samo dodati metodu Clone u klasu Korisnik koja vrsi kloniranje pomocu metode MemberwiseClone().

Takođe se ovaj patern može koristiti za vršenje analize nad nekim skupom podataka iz baze (to se obavlja u klasi administrator), tj.tada se isti objekat moze klonirati vise puta i koristit za vise analiza(npr. Podaci o narudzbama)