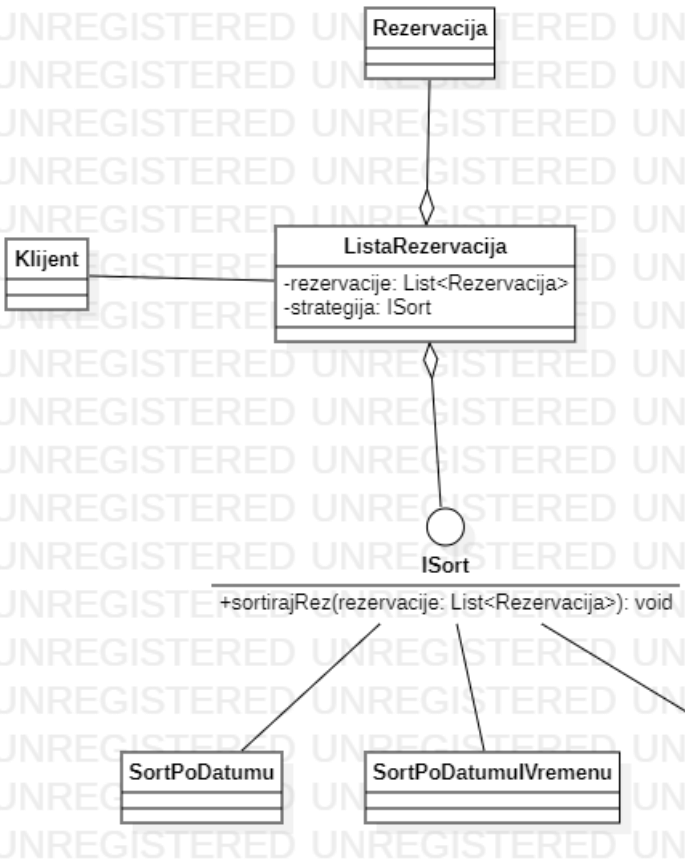
PATERNI PONAŠANJA

Stategy patern

Strategy patern omogucava izvadjanje algoritma iz maticne klase i ukljucuje ga u posebne klase.

U nasem sistemu bismo Strategy patern mogli iskoristiti tako da prethodno Musteriji omogucimo da sortira svoje rezervacije. Rezervacije bi se mogle sortirati na vise vise nacina (po lokalu u kojem se rezervisani stol pronalazi I po datumu I vremenu) Conext klasa bi za atribut imala listu rezervacija i strategiju sortiranja. Preko nje bi Client (Musterija) davala informacije za IStrategy algoritme.



State patern

State patern je dinamička verzija Strategy paterna odnosno objekat mijenja način ponašanja na osnovu trenutnog stanja.

State patern bi se u našem sistemu mogao realizvovati na sljedeci način. Kada Uposlenik

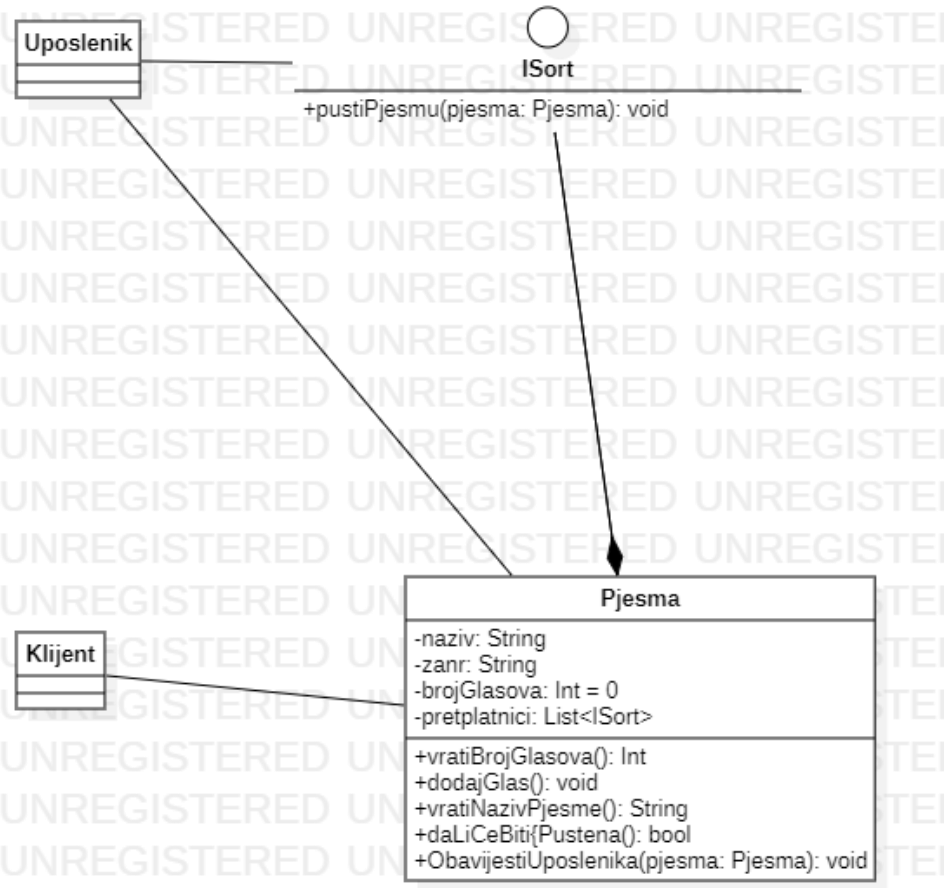
provjerava status neke određene narudžbe (potvrdjena, kreirana, otkazana, u priremi itd). Ako je narudžba kreirana Uposleniku bi se otvarao prozor na kojem moze da potvrdi narudžbu. Ako je narudzba u pripremi on može provjeriti do kojeg postotka (za reliziranje ovog koraka bilo bi potrebno u sistem dodati koristenje progress bara i još par stavki). Ako je Narudzba plaćena metoda bi to samo ispisala.

TemplateMethod Pattern
TamplateMethod patern nam omogućava izdvajanje odredjenih koraka algoritma u odvojene podklase.

Za primjenu ovog paterna u sistemu bismo dodali mogućnost da se prilikom kreiranja eventa isti može označiti kao VIP Event ili ne te bi isti postao podklasa klase Event. VIP Event bi se stavljao prvi na listu eventova nekog lokala te bi za isti bila potrebna dirketna pozivnica. Sve ostale funkcionalosti bi bile identicne kao za samu klasu Event.

Observer Patern
Uloga ovog paterna je uspostavljanje relacije između objekata tako kada jedan objekat promijeni stanje drugi zainteresirani objekti se obavještavaju.

U našem sistemu primjena ovog paterna bi se mogla realizovati tako što bi se Uposleniku svaki put kada završi glasanje za pjesmu, odnosno kad svi stolovi daju svoj glas, se poziva funkcija DaLiCeBitiPustena koja računa da li pjesma ima 50% ili više glasova za DA. Ako je pjesma izglasana da bude puštena Uposlenik dobija obavijest da pusti pjesmu.



Iterator Patern

Uloga Iterator paterna je da omogući sekvencijalni pristup elementima kolekcije bez prepoznavanja kako je kolekcija struktuirana.

Iterator patern bi se u našem sistemu mogao iskoristiti u slučaju kada Uposlenik pušta pjesme. Pošto se može desiti da su pjesme već naručene ili izglasane u trenutku kada ih nije moguće pustiti (već je puštena neka druga pjesma). Uposlenik stavlja pjesme u neki “red čekanja” (lista pjesama u lokalu) . Međutim Uposlenik može birati da li će puštati pjesme po najvećem broju glasova, p žanrovima, redoslijedom kojim su naručene ili pak nasumično.

Chain of responsibility patern

U našem sistemu bismo mogli iskoristiti Chain of responsibility patern kako bismo omogućili da Uposlenik ili Menadzer prilikom dodavanja novog Artikla u sistem samo upišu njegovo ime te da nova klasa AritklMaker putem interneta nadje njegovu cijenu prodaje te da se u Meni upiše

Artikl sa svim potrebnim informacijama.

Mediator patern

Mediator patern u našem sistemu bi se mogao iskoristiti ako bismo dodali novi korisnički zahtjev kao na primjer mogućnost chata među uposlenicima istog lokala. Uposlenik bi mogao slati poruke samo svom kolegi koji radi u istom lokalu. IMedijator bi vršio provjeru primaoca poruke.