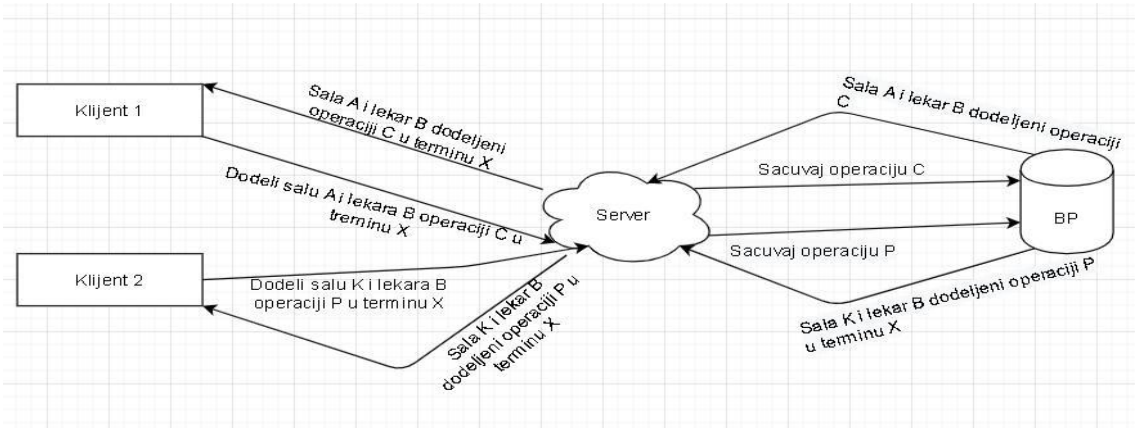
Ime i prezime: __Aleksa Goljović__

Broj indeksa: __SW14-2017__

Zadatak: __Rešavanje konfliktnih situacija za studenta 3__

1. Konkurentno dodeljivanje istog lekara različitim operacijama koje se održavaju u istom terminu

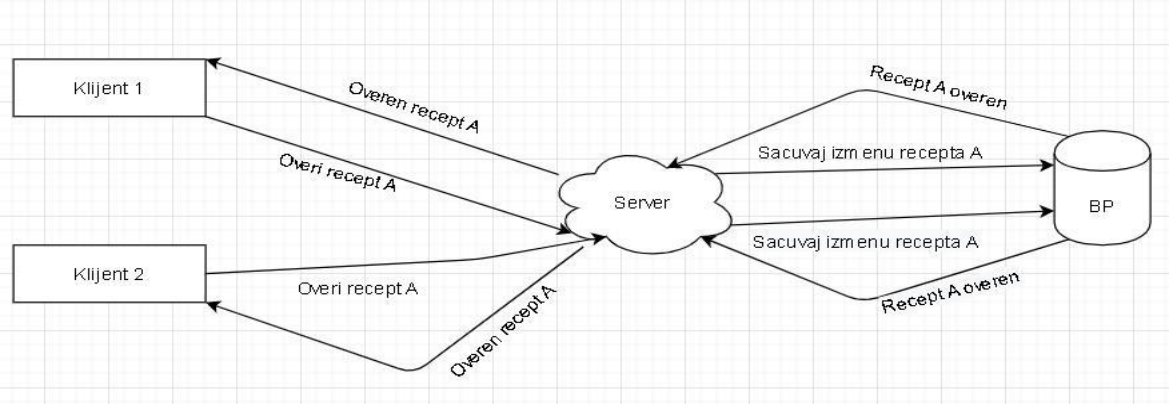
Problem: Admin klinike vrši dodeljivanje sala za operacije. Pri tom dodeljivanju, admin klinike može pored već odabranog lekara za operaciju, da odabere neke dodatne lekare koji su obavezni da prisustvuju toj operaciji. Na taj način, prilikom dodatnog dodeljivanja lekara može se desiti da dva različita admina dodele istog lekara različitim operacijama koje se održavaju u istom terminu. Na ovaj način, jedan isti lekar će u jednom terminu biti prisutan na dve ili više operacija što nije dozvoljeno da se desi u našem sistemu, te s toga, potrebno je da se reši ovaj problem.



Rešenje: Ovaj problem je rešen tako što je primenjena tehnika optimističkog zaključavanja tabele *Lekar*. Da bi se ova tehnika mogla sprovesti potrebno je dodati novi atribut koji je anotiran sa *@Version.* Međutim, prilikom dodele lekara operaciji, taj lekar se neće menjati, te se vrednost atributa anotiranog sa *@Version* neće promeniti. Iz tog razloga je potrebno uvesti novi atribut *lastChange.* Student 1 je već ovo uradio kako bi rešio svoje konfliktne situacije. Iz tog razloga, sve što je potrebno da uradim da bih rešio ovaj problem jeste da prođem kroz sve lekare koji su dodeljeni operaciji, promenim vrednost atributa *lastChange* i sačuvam promene tog lekara. Na ovaj način, ako je neka druga transakcija već dodelila u tom trenutku nekog od lekara drugoj operaciji ili pregledu, ova transakcija se neće izvršiti.

2. Konkurentna overa recepata

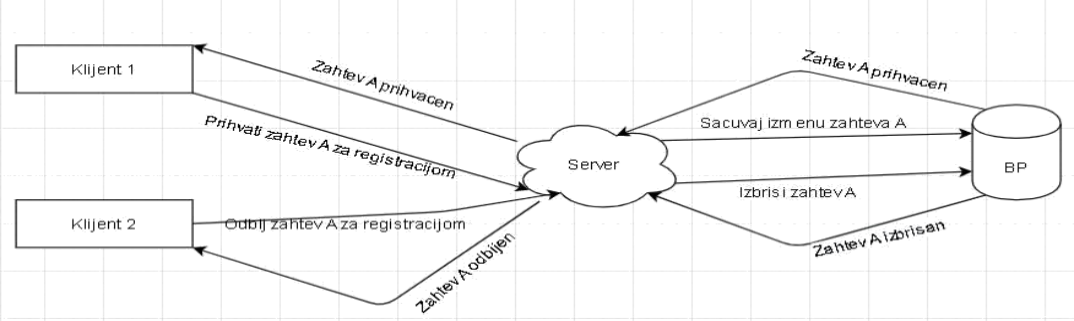
Problem: Nakon što je pregled obavljen i pacijentu je prepisan lek, potrebno je overiti recept za lek. Recepte može da overi medicinska sestra koja radi na klinici na kojoj je održan pregled. U našem sistemu, bilo koja sestra koja radi na klinici na kojoj je održan pregled može da overi recept. Na ovaj način, kada se jedna transakcija overe recepta završi, sledeća transakcija će da „pregazi“ ono što je prethodna transakcija izmenila.



Rešenje: Rešenje ove konfliktne situacije ostvaruje se optimističkim zaključavanjem, tako što se uvodi novi atribut koji se anotira sa *@Version.*Na ovaj način, kada prva transakcija dodeli medicinsku sestru receptu, vrednost atributa anotiranog sa *@Version*se uvećava za jedan. Na taj način sledeća transakcija kada pokuša da sačuva svoje promene, *Entity Manager* će prvo da uporedi vrednosti atributa anotiranog sa *@Version* u bazi sa vrednošću tog atributa u toj transakciji. S obzirom da se vrednosti neće podudarati, doći će do izuzetka i ta transakcija se neće završiti.

3. Konkurentna obrada zahteva za registraciju

Problem: Admini kliničkog centra mogu da prihvate ili odbiju zahtev za registracijom pacijenta. Problem nastaje kada dva admina istovremeno pokušaju da prihvate/odbiju zahtev za registracijom. Ako je prvi admin prihvatio zahtev, a drugi odbio zahtev, pacijentu će prvo stići mejl da potvrdi registraciju, a zatim mejl da je zahtev za registracijom odbijen. U bazi će se prvo kreirati zdravstveni karton za tog pacijenta, a zatim, nakon što se izvrši druga transakcija, taj pacijent će biti izbrisan iz baze. Zbog toga je potrebno na neki način rešiti ovaj problem.



Rešenje: Problem se rešava pesimističkim zaključavanjem. Upit koji dobavlja korisnika za kog se potvrđuje zahtev se anotira sa *@Lock(LockModeType.PESSIMISTIC_WRITE).* Na ovaj način u jednom trenuku samo jedna transakcija može da menja selektovanu torku. Pored ovoga, ako se ne navede *timeout,* postaviće se podrazumevana vrednost i na taj način će sledeća transakcija da sačeka izvesno vreme da se torka otključa i opet će dobiti mogućnost da ažurira isti zahtev. Da bismo sprečili da transakcija čeka da se torka otključa, vrednost za *timeout* postavljamo na 0. Na taj način prva transakcija zaključa torku, a sledeća transakcija kada pokuša da pristupi toj torci neće moći i dobiće izuzetak.