Konkurentni pristup resursima u bazi

Konfliktne situacije koje su rešene u sklopu funkcionalnosti vezanih za pacijenta su:

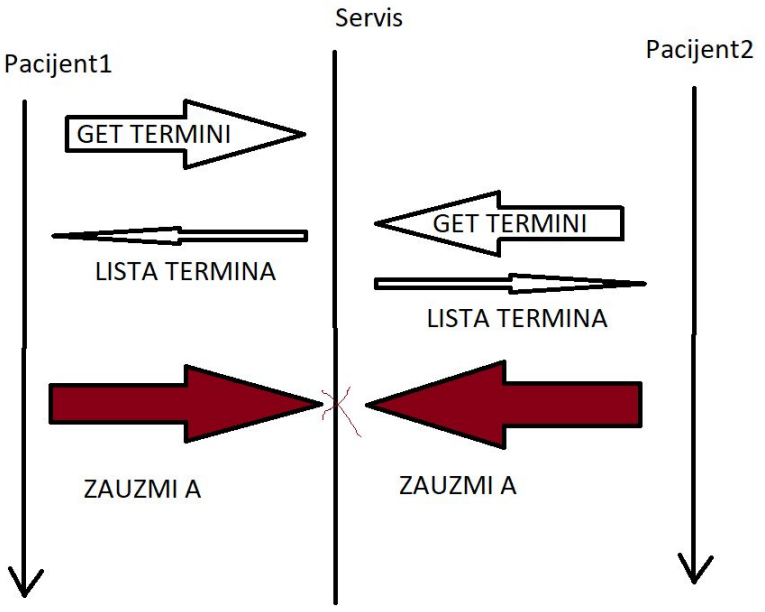
1. Rezervisanje unapred zadatih termina
2. Odabir posebnog termina (koji nije predefinisan)
3. Ocenjivanje klinike

1. Rezervisanje unapred zadatih termina

Kada pacijent pošalje zahtev za zauzimanje termina, servis prvo šalje upit u bazu kojim proverava da li slobodan pregled sa takvim ID-jem postoji. Ukoliko postoji, upisuje se ID pacijenta u slobodan pregled.

U slučaju da dva (i više) pacijenta jednovremeno zatraže isti termin, može doći do štetnog preplitanja, tako da oba zahteva izvrše upit koji će ih obavestiti da je ponuđeni pregled slobodan i u proizvoljnom redosledu izvršiti upisivanje svog IDa.

Slika ispod prikazuje pomenutu konfliktnu situaciju. Konfliktna situacija je razrešena pesimističkim pristupom. Klijent koji prvi zatraži rezervaciju termina, zaključaće red koji predstavlja pregled za izmene…...U trenutku kada se prijavi (upiše svoj ID), završiće se transakcija. Tada će drugi zahtev moći da sprovede svoju transakciju, ustanoviće da je izabrani pregled zauzet na osnovu prvog upita i korisnik će biti obavešten da je došlo do greške.
Implementacija: **AppointmentServiceImpl** (**bookAppointment**)



2. **Odabir posebnog termina (koji nije predefinisan)**

   Pacijent može na stranici sa listom klinika da pošalje upit, koji će mu vratiti listu klinika sa slobodnim lekarima za izabrani datum i tim pregleda. Svaki lekar poseduje listu slobodnih termina za izabrani datum. Lista se sastoji od vremenskih isečaka od 30 minuta (virtuelni pregledi) i predefinisanih pregleda koji se takođe nude.
   Klikom predefinisani pregled, konfliktna situacija se razrešava na gore pomenut način.

   Klikom na virtuelni pregled, šelje se zahtev servisu koji prvo šalje upit bazi da li postoji pregled (ili zahtev za pregled) koji se preklapa sa zahtevanim kod izabranog lekara. Ukoliko ne postoji, kreira se novi zahtev za pregled i upisuje u bazu.

   U ovom slučaju se dešava konfliktna situacija kada klijenti pošalju identične zahteve i zadovolje uslov da je izabrani lekar slobodan u zahtevanom intervalu. Doći će do upisivanja više zahteva za pregled koji se preklapaju. (Pošto ova akcija uključuje upisivanje nove vrednosti u bazu, preplitanje se razrešilo serijalizacijom zahteva.)

3. Ocenjivanje klinike

   Pacijent može da oceni kliniku u kojoj ima bar jedan (završen) pregled, takođe može i da menja datu ocenu.
   Postoje dva toka kada pacijent pošalje ocenu za kliniku. Prvi je da ocena već postoji, u tom slučaju se vrši promena (update) pronađene ocene. Drugi scenario je kada pacijent još nije ocenio kliniku, tada se kreira nova ocena i upisuje u bazu.

   Konflikt može da se desi kada pacijent “brzo” šalje ocene.
   U prvom slučaju to nije obrađeno, jer smatramo da to nije (naročito) štetan scenario. Pacijent će rapidno slati zahteve za koje će ustanoviti da je klinika već ocenjena i izmeniti staru ocenu novom. Na kraju će biti ipak biti neka od ocena koje je klijent slao.
   Ovde ćemo rešavati **drugi slučaj** koji može da naruši bazu. Ukoliko pacijent prvi put ocenjuje kliniku:
   a. proverava se da li je već ocenjiva (netačno)
   b. zatim se proverava da li pacijent ima završen pregled u klinici
   c. upisuje se ocena u bazu

   U ovom scenariju može doći do stetnog preplitanja tako da više zahteva u koraku b) ustanovi da ne postoji ocena i zatim da se ocena višestruko upiše ocenu pacijenta u bazu. Ovim se model narušava, jer jedan pacijent može da da samo jednu ocenu klinici.
   Ova konfliktna situacija je rešena tako što je izvršena serijalizacija. Prvi zahtev koji pristigne i propadne u scenario koji obrađuje slučaj kad se ocena upisuje u bazu (proverava da li pacijent ima završen pregled i nije dao ocenu) će izvršiti serijsku transakciju i time sprečiti neželjeno preplitanje.

Komponente koje implementiraju ovog konflikta: **ClinicGradeTransService** i **ClinicService**.