Konkurentni pristup – Student 2

Konfliktna situacija 1:

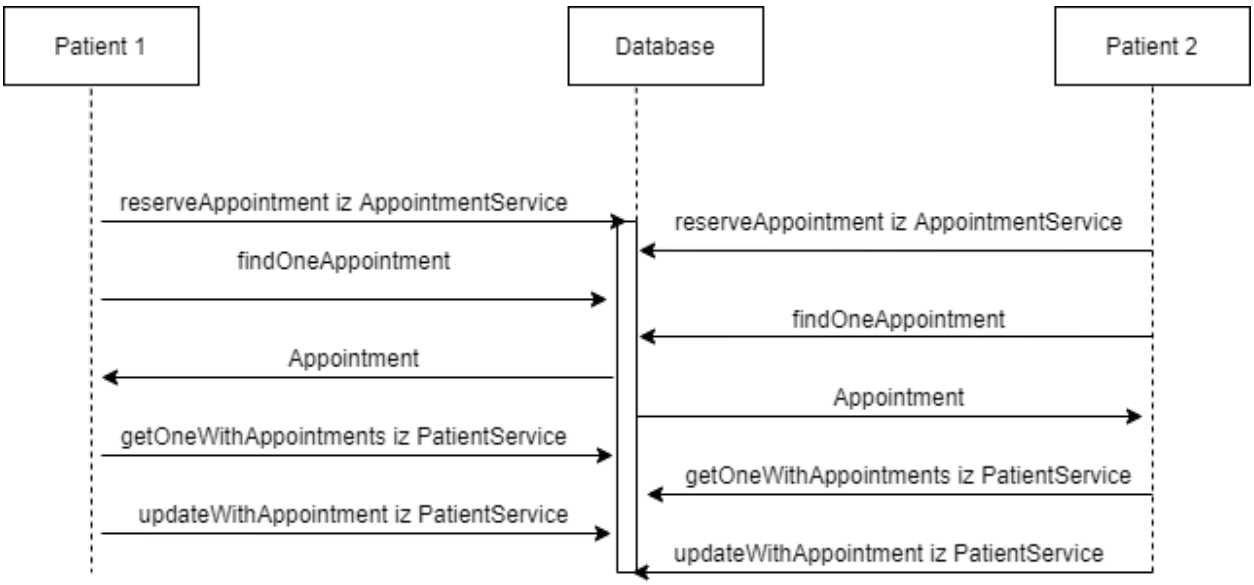
Opis konfliktne situacije

Pregledi koji su unapred definisani ne smeju biti rezervisani od strane više različitih korisnika, više istovremenih korisnika aplikacije ne može da zakaže savetovanje u istom terminu kod istog farmaceuta.

Tok zahteva klijenta i odgovor servera

Ukoliko dva korisnika sa dve različite mašine u isto vreme pokušaju da rezervišu unapred definisan pregled kod dermatologa ili savetovanje kod farmaceuta dolazi do konfliktne situacije.

Endpoint koji se gađa POST zahtevom je patients/reserve_appointment. Nakon uspešne autorizacije, proverava se da li korisnik postoji u sistemu, koliko ima negatinih poena, zatim se poziva metoda reserveAppointment(AppointmentDTO) iz AppointmentService. U metodi se dobavlja izabrani pregled, proverava se da li korisnik ima već zakazan pregled/savetovanje u izabranom terminu ili da li je došlo do preklapanja termina, zatim se pregled/savetovanje dodaje u kolekciju pregleda korisnika, upravo ovde dolazi do konflikta, jer oba korisnika dobijaju poruku o uspešnosti rezervisanja, ali je zapravo samo jedan uspeo da rezerviše. U nastavku je prikazan problem.



Rešenje

Ovaj problem rešen je pesimističkim zaključavanjem, gde je u AppointmentService iznad metode reserveAppointment (AppointmentDTO a) postavljena anitacija @Transactional, zatim u AppointmentRepositoryDB iznad metode findOneAppointment (Long id), dodato @Lock(LockModeType.PESSIMISTIC_WRITE) i @QueryHints({@QueryHint(name =”javax.persistence.lock.timeout”, value=”0”)}).

Konfliktna situacija 2:

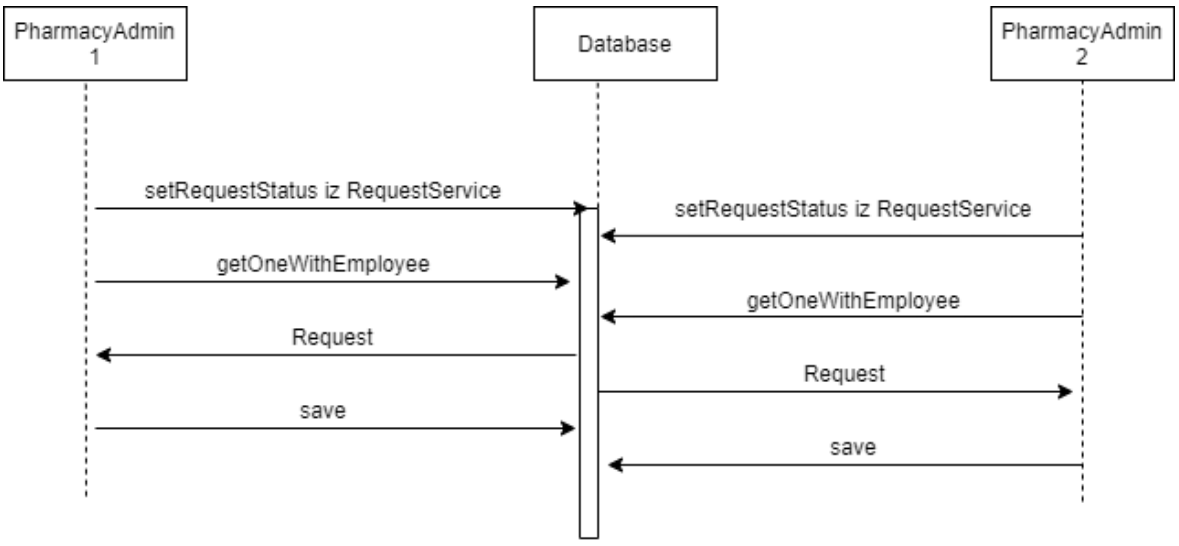
Opis konfliktne situacije

Prihvatanje ili odbijanje zahteva za odmor ne može biti prihvaćeno od strane više administratora apoteke.

Tok zahteva klijenta i odgovor servera

Ukoliko dva korisnika sa dve različite mašine u isto vreme pokušaju da odgovore na zahtev za odsustvom/ godišnjim odmorom dolazi do konfliktne situacije.

Endpoint koji se gađa PUT zahtevom je request/update. Zatim se poziva metoda setRequestStatus(Long id, RequestDTO request) iz RequestService. U metodi se dobavlja izabrani zahtev za odsustvom, zatim se postavlja status da li je prihvaćen ili ne, ovde dolazi do konflikta, jer oba korisnika dobijaju poruku o uspešnosti prihvatanja/odbijanja, ali je zapravo samo jedan uspeo da prihvati/odbije. U nastavku je prikazan problem.



Rešenje

Ovaj problem rešen je optimističkim zaključavanjem, gde je u RequestService iznad metode setRequestStatus(Long id, RequestDTO request) postavljena anotacija @Transactional, zatim je dodat Version atribut, sa get i set metodama u klasi Request.

Konfliktna situacija 3:

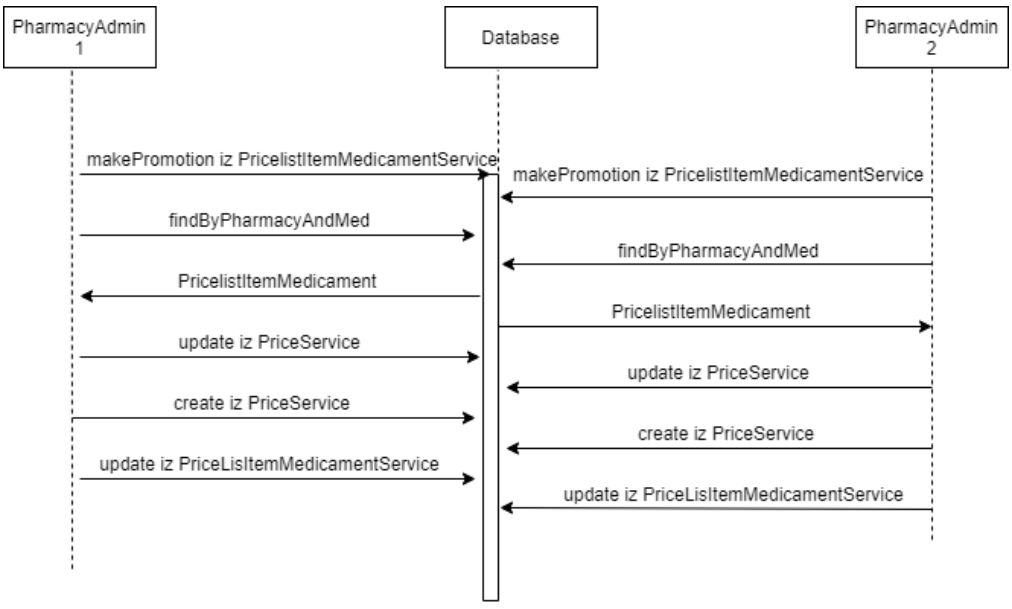
Opis konfliktne situacije

Definisanje promocije nad istim lekom ne može biti omogućeno od strane više administratora apoteke.

Tok zahteva klijenta i odgovor servera

Ukoliko dva korisnika sa dve različite mašine u isto vreme pokušaju da definišu promociju leka, dolazi do konfliktne situacije.

Endpoint koji se gađa PUT zahtevom je pricelistItems/promotion. Zatim se poziva metoda makePromotion (Long id, Long pid, PricelistItemMedicamentDTO pricelistItem) iz PricelistItemMedicamentService. U metodi se dobavlja izabrani lek sa cenom iz konkretne apoteke, zatim se trenutna cena logički briše, postavlja se nova cena, koja predstavlja promociju, koja počinje da važi od tog trenutka, ovde dolazi do konflikta, jer oba korisnika dobijaju poruku o uspešnosti definisanja promocije, ali je zapravo samo jedan uspeo da definiše. U nastavku je prikazan problem.



Rešenje

Ovaj problem rešen je pesimističkim zaključavanjem, gde je u PricelistItemMedicamentService iznad metode makePromotion (Long id, Long pid, PricelistItemMedicamentDTO pricelistItem) postavljena anitacija @Transactional, zatim u PricelistItemMedicamentRepositoryDB iznad metode findByPharmacyAndMed(Long id, Long pid), dodato @Lock(LockModeType.PESSIMISTIC_WRITE) i @QueryHints({@QueryHint(name =”javax.persistence.lock.timeout”, value=”0”)}).