

Konkurentni pristup – Student 4

Konfliktna situacija br. 1

Opis konfliktne situacije

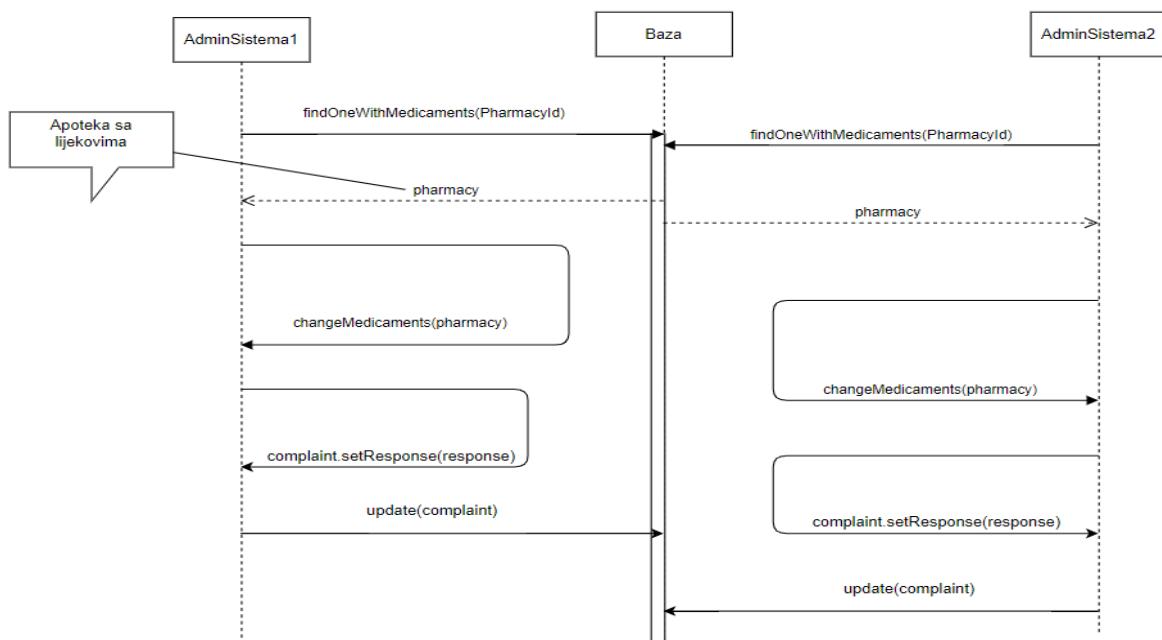
Više pacijenata ne može da kupi lijekove preko eRecepta u isto vrijeme u istoj apoteci , jer se stanje lijekova može dovesti u nekonzistentno stanje.

Crtež zahtjeva i odgovor klijenta

Dva pacijenta mogu da izvrše upload QR kodova, i ukoliko odluče da kupe lijekove iz iste apoteke, dolazi do konfliktne situacije.

1. Endpoint: POST /patients/addEPrescription.
2. Funkcija: createePrescription(QRCodeDTO) u ePrescriptionService-u.

Kreira se novi eRecept na osnovu podataka u QR kodu, provjerava se da li je datum preuzimanja istekao, i ukoliko nije količine u apoteci se u skladu sa količinama u eReceptu smanjuju. Ukoliko niko nije pristupao toj apoteci, stanje lijekova će se promjeniti i klijentu će biti prikazana poruka o uspješnoj kupovini, dok, ukoliko je već neko pokušao da pristupi toj apoteci i mijenja stanje lijekova, neće mu to biti dozvoljeno i dobiće poruku o neuspješnoj kupovini.



Rješenje

Ovaj problem je riješen pomoću pesimističnog zaključavanja, tako što je u PharmacyRepositoryDB iznad funkcije **Pharmacy getOneWithMedicaments(Long id)** dodato
@Lock(LockModeType.PESSIMISTIC_READ), kao i
@QueryHints({@QueryHint(name = „javax.persistence.lock.timeout“, value = „0“)})

Dok je iznad **ePrescriptionService** iznad funkcije **createePrescription(QRCodePharmacyDTO dto)** dodana anotacija **@Transactional**.

Na ovaj način je riješen problem, tako što ukoliko dva klijenta pokušaju preko eRecepta da kupe lijekove, prvom koji zauzme taj objekat će uspjeti da izmijeni stanje lijekova, dok drugom neće pristup biti omogućen.

Konfliktna situacija br. 2

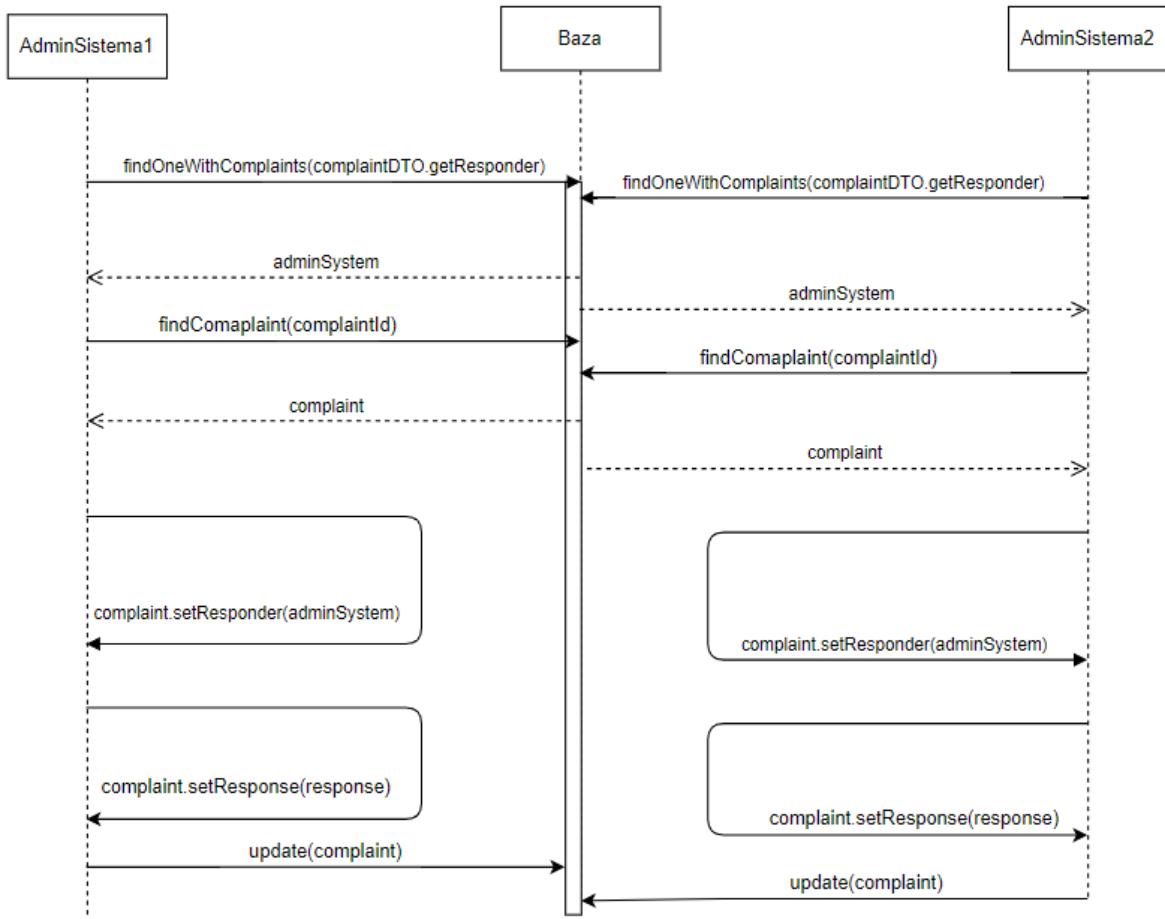
Opis konfliktne situacije

Više administratora sistema mogu da daju odgovor na žalbu, a treba omogućiti da samo jedan može, jer je tako predviđeno zahtjevima, a i modelom.

Crtež zahtjeva i odgovor klijenta

Dva administratora su prijavljena i imaju listu žalbi na koje mogu da odgovore, ukoliko oba u isto vreme pokušaju, drugi će da pregazi prvog.

1. Endpoint: POST api/complaints/update
2. Funkcija: writeResponse(ComplaintDTO) u ComplaintService



Rješenje

Ovaj problem je riješen pomoću optimističnog zaključavanja tako što je u **Complaint** klasu dodana kolona tj. atribut **Long version**, koji se anotira pomoću **@Version**.

Dok je u **ComplaintServiceImpl** funkcija **writeResponse(ComplaintDTO)**, anotirana pomoću anotacije **@Transactional**

Kada administratori pokušaju da odgovore na žalbu, ukoliko verzija se ne podudara, neće im biti dozvoljeno i rollback će da se desi.

Konfliktna situacija br. 3

Opis konfliktne situacije

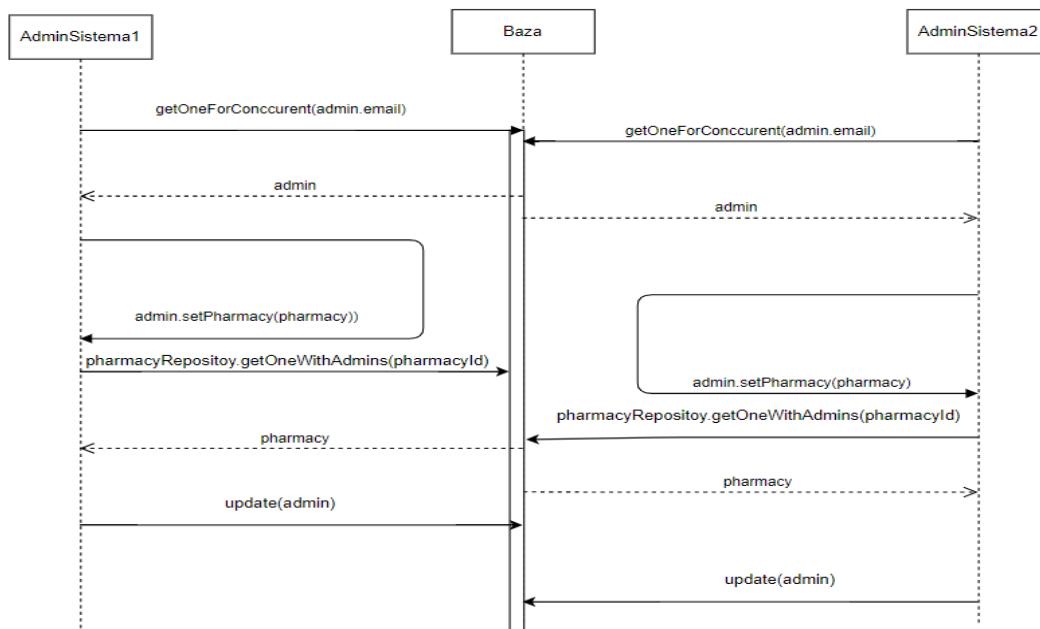
Više administratora sistema u isto vrijeme mogu da različitim apotekama da dodijele istog admina apoteke, ali to ne smije da se desi, jer jedan admin apoteke može da radi u samo jednoj apoteci.

Crtež zahtjeva i odgovor klijenta

Dva administratora su prijavljena i pokušaju da istog admina apoteke dodijele apotekama.

1. Endpoint: POST api/updatePharmacyToAdmin
2. Funckija: updatePharmacy(AdminPharmacyDTO dto) u PharmacyAdminService

Dobave se admini iz baze, dodijeli im se apoteka, te se sačuvaju u bazi, na ovaj način se pregazi prvi administrator sistema i njegovo zaposlenje administratora apoteke.



Rješenje

Ovaj problem je riješen pomoću pesimističkog zaključavanja tako što je u **PharmacyAdminRepositoryDB** iznad funkcije **AdminPharmacy getOneForConcurrent(String id)** dodato **@Lock(LockModeType.PESSIMISTIC_READ)**, kao i **@QueryHints({@QueryHint(name = „javax.persistence.lock.timeout“, value = „0“)})**. Dok je u **PharmacyAdminServiceImpl** iznad funkcije **updatePharmacy** dodana anotacija **@Transactional**.

Na ovaj način prvi admin će da doda apoteku adminu apoteke, dok drugi admin neće imati pristup tome.

