Rešenja konfliktnih situacija – student 4

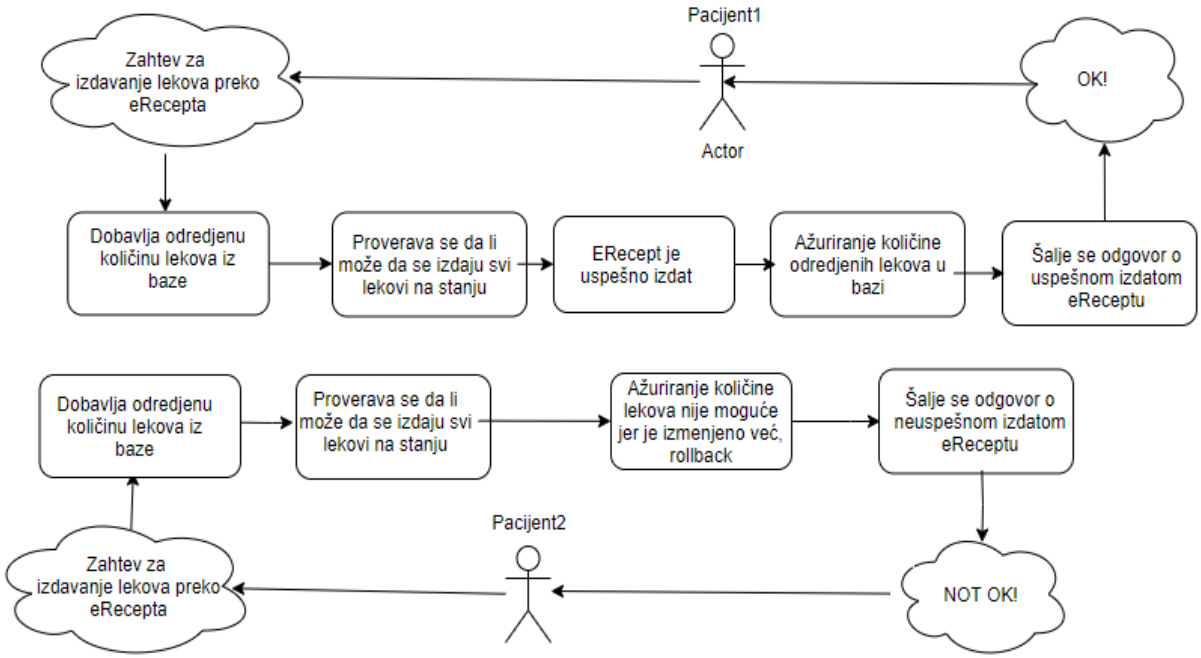
SLUČAJ 1: Prilikom izdavanja eRecepta se izdaju ili svi ili ni jedan lek i stanje leka u apoteci se ažurira

Pacijent vrši upload QR koda. QR kod sadrži listu lekova koje je potrebno izdati pacijentu. U slučaju da pacijent izabere validan QR kod (QR kod koji nije prethodno iskorišćen) prikazuje mu se lista svih apoteka koje na stanju imaju sve lekove i količine tih lekova, koji se nalaze na eReceptu. Za svaku stavku se prikazuje ukupna cena svih lekova, naziv i mesto gde se apoteka nalazi, kao i ocena apoteke. Pacijentu se daje mogućnost da sortira/filtrira listu rezultata po bilo kom kriterijumu. Pacijent bira apoteku u kojoj želi da preuzme sve lekove klikom na dugme pored željene stavke. Nakon odabira apoteke, šalje se zahtev na servis da se ažuriraju koliične lekova u izabranoj apoteci.

Pretpostavimo da Pacijent1 traži zahtev za izdavanje lekova preko eRecepta u odabranoj apoteci, a neposredno nakon njega i Pacijent2 traži zahtev za izdavanje lekova u istoj apoteci u kojoj su svi lekovi bili dostupni. Pacijent1 uspešno završava izdavanje lekova preko eRecepta. Za Pacijent2 se i dalje obradjuje zahtev za izdavanje lekova preko eRecepta u istoj apoteci kao i za Pacijent1, u kojoj više ne bi smeo da može da se preuzme jer je u tom vremenskom trenutku već izmenjena dostupna količina određenog leka od strane Pacijent2. Međutim Pacijent2, preuzetu količinu određenog leka od strane Pacijent1 ne vidi, te on smatra da može da preuzme lekove u istoj apoteci u kojoj je prethodno izvršena izmena količina odredjenih lekova, te mu oni više nisu dostupni. Na ovaj način dobijamo da jedan pacijent dobija podatke koji nisu više validni i takav problem se zove *Non-repeatable reads*.

Ovaj problem je rešen pomoću transakcija i ISOLATION_READ_COMMITED. Transakcije omogućavaju da se izmene podataka u BP unutar jedne transakcije ili sačuvaju (commit) ili odbace (rollback). Ovakvo ponašanje nam odgovara za rešavanje ovog problema. Metoda proccedEReceipt u servisu EPrescriptionServiceImpl je anotirana sa anotacijom @Transactional.

Metoda save(..) u EPrescriptionServiceImpl-u je anotirana sa @Transactional(readOnly = false, propagation = Propagation.REQUIRES_NEW), pri čemu je sa readOnly omogućena izmena podataka, a sa propagacijom REQUIRES_NEW je omogućeno kreiranje nove transakcije i ako neka već postoji. Metoda updateMedicineQuantityEreceipt(..) u MedicinePriceServiceImpl je takodje anotirana sa @Transactional(readOnly = false, propagation = Propagation.REQUIRES_NEW). Na ovaj način je omogućeno da ili sve akcije budu izvršene, ili sve budu poništene.



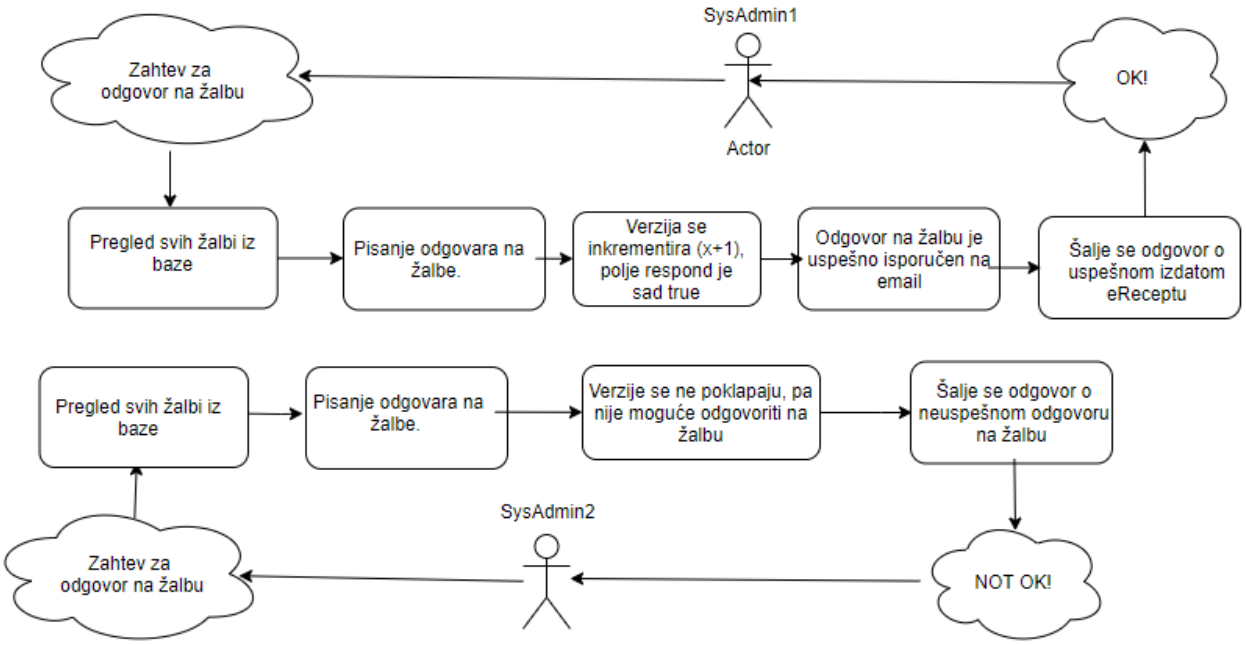
SLUČAJ 2: Na jednu žalbu može da odgovori samo jedan admnistrator sistema

Amidnistrator sistema ima mogućnost da pregleda sve žalbe, koji su pacijenti napisali, kao i da odgovori na iste. Unošenjem odgovora na žalbu i klikom na dugme za slanje, šalje se zahtev ka serveru za update ove žalbe. Server zatim pronadje žalbu na koju je administrator sistema odgovarao, promeni respond Boolean vrednost na true i čuva u bazu.

Pretpostavimo da SystemAdmin1 traži zahtev za odgovor na žalbu, a skoro u isto vreme nakon njega i SystemAdmin2 traži zahtev za odgovor na istu žalbu. SystemAdmin1 uspešno završava odgovaranje na žalbu. Za SystemAdmin2 se i dalje obradjuje zahtev za odgovaranje na žalbu za istu onu na koju je SystemAdmin1 već odgovorio. Međutim SystemAdmin2, odgovor na žalbu od strane SystemAdmin1 ne vidi, te on smatra da može da ogovori na istu žalbu na koju je odgovorio SystemAdmin1, te mu odgovor postaje onemogućen.

Ovaj problem je rešen pomoću optimističkog zaključavanja. Optimističko zaključavanje je izabrano zbog boljih performansi u odnosu na pesimističko zaključavanje, ali i zbog toga što nije neophodno zaključavanje podataka. Zbog ovoga je u tabelu Complaint, dodato polje private Long version i dodeljena mu je anotacija @Version. Pomoću @Version anotacije postiže se to, da se pre čuvanja izmene, polje Boolean respond dobija vrednost true, proveri verzija podatka koji se čuva. Ako je verzija podatka ista, kao u trenutku kada je učitana iz baze, podatak se ažurira, a polje version se inkrementira. U slučaju da SysAdmin1 malo brže klikne na zahtev za slanje odgovora, njegov zahtev će pre stići do servera, pri čemu će se njegova izmena sačuvati i njegov će odgovor biti poslat. A za SysAdmin2 će se proveravati verzija podatka, i ona neće biti ista kao u trenutku preuzimanja informacija iz baze, jer je SysAdmin1 u

medjuvremenu uradio izmenu. Iz tog razloga dolazi do ObjectOptimisticLockingFailureException-a kada se u okviru ComplaintService klase, u metodi respond(..) pozove complaintRepository.save(complaint).



SLUČAJ 3: Dobavljač ne sme da da ponudu za narudžbenicu koja ne postoji, ili je u medjuvremenu izmenjena, odnosno administrator apoteke ne može da vrši izmenu nad narudžbenicom za koju već postoje ponude

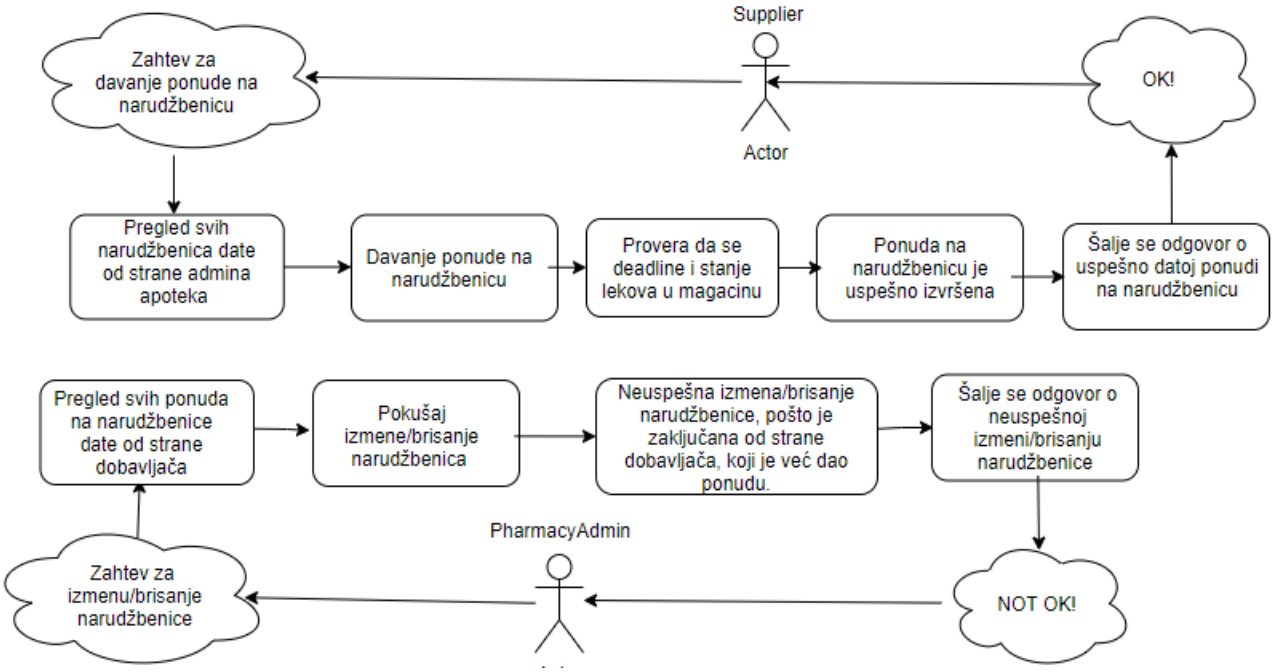
Nakon što se uloguje, dobavljač ima pristup svim pristiglim narudžbenicama na koje, klikom na dugme, može da da svoju ponudu. Do problema može doći ukoliko dodje do izmene same narudžbenice od strane administratora apoteke, ili pak do njenog brisanja, u istom trenutku u kojem je dobavljač dao ponudu na nju.

Pretpostavimo da Supplier šalje ponudu za izabranu narudžbenicu, a skoro u isto vreme nakon njega i PharmacyAdmin traži zahtev za proveru o broju pristiglih ponuda za narudžbenicu. Supplier uspešno završava davanje ponude za narudžbenicu. Za PharmacyAdmin se i dalje obradjuje zahtev za proveru o broju pristiglih ponuda za narudžbenicu. Medjutim PharmacyAdmin, ponudu za narudžbenicu od strane Supplier ne vidi, te on smatra da narudžbenica nema ponuda i da se uspešno može ažurirati/obrisati, što vodi do neispravnog rada aplikacije.

Ovaj problem je rešen pomoću pesimističkog zaključavanja narudžbenice, kako bismo jednom dobavljaču zabranili davanje ponude za narudžbenicu u trenutku u kom se ona ažurira od strane administratora apoteke. Metoda findErrandById(Id), koja se nalazi u ErrandRepository, koja se poziva prilikom izmene narudžbenice i davanja ponude za odredjenu narudžbenicu, ali usput radi i zaključavanje narudžbenice koja se u datom trenutku apdejtuje ili nad kojom se vrši ponuda (LockModeType.PESSIMISTIC_WRITE). Tako da prilikom izmene narudžbenice od strane administratora apoteke ni jedan dobavljač ne može da joj pristupi i da svoju ponudu.

Takodje, prilikom davanja ponude od strane dobavljača, administrator ne može da vrši izmene nad istom.

Slučaj 1.



Slučaj 2.