Konkurentni pristup bazi podataka

Konfliktne situacije

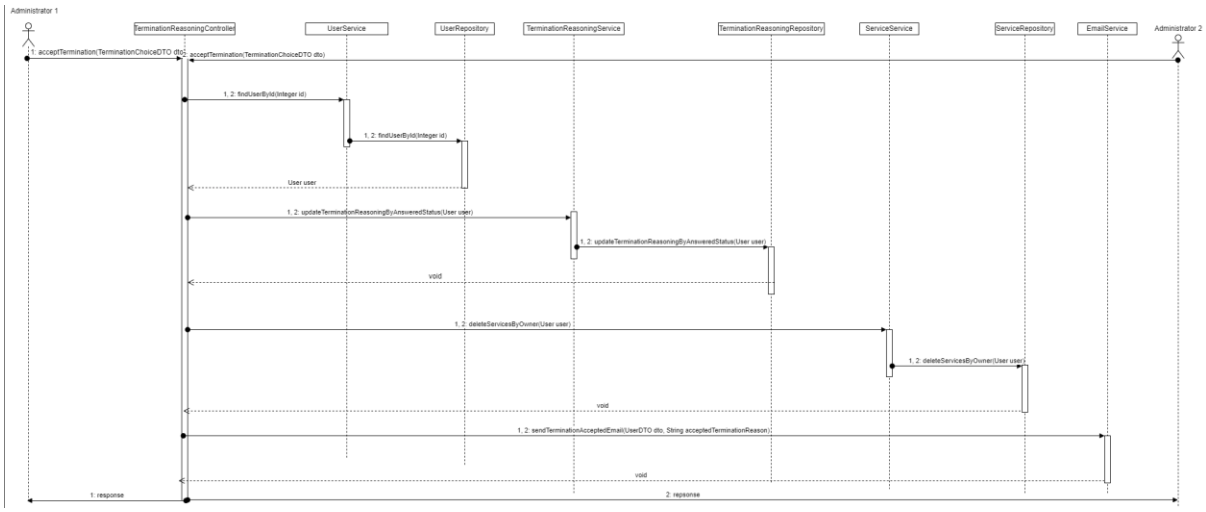
Student: Teodor Sakal Franciković

Broj indeksa: SW 22/2019

Situacija 1:

Dva (ili više) administratora pokušavaju u isto vreme da odgovore na jedan zahtev za brisanje, što podrazumeva da ili prihvataju ili odbijaju korisnikov zahtev za brisanje. Primer posledice ovog konflikta može biti da se korisniku prvobitno odobri zahtev za brisanje (što bi impliciralo da korisnik više nije aktivan u sistemu), a nakon toga da se korisniku ne odobri zahtev, pa bi opet trebalo da postane aktivan.

Tok akcije: Dva administratora istovremeno pritiskaju na dugme za prihvatanje/odbijanje zahteva za brisanje naloga. Na server stižu oba zahteva, a ako oba prođu, postoji mogućnost da će se stanje aktivnosti korisnikovog naloga promeniti u odnosu na prvobitnu odluku i da će korisniku stići više mejlova sa različitim razlozima prihvatanja/odbijanja razloga brisanja naloga.

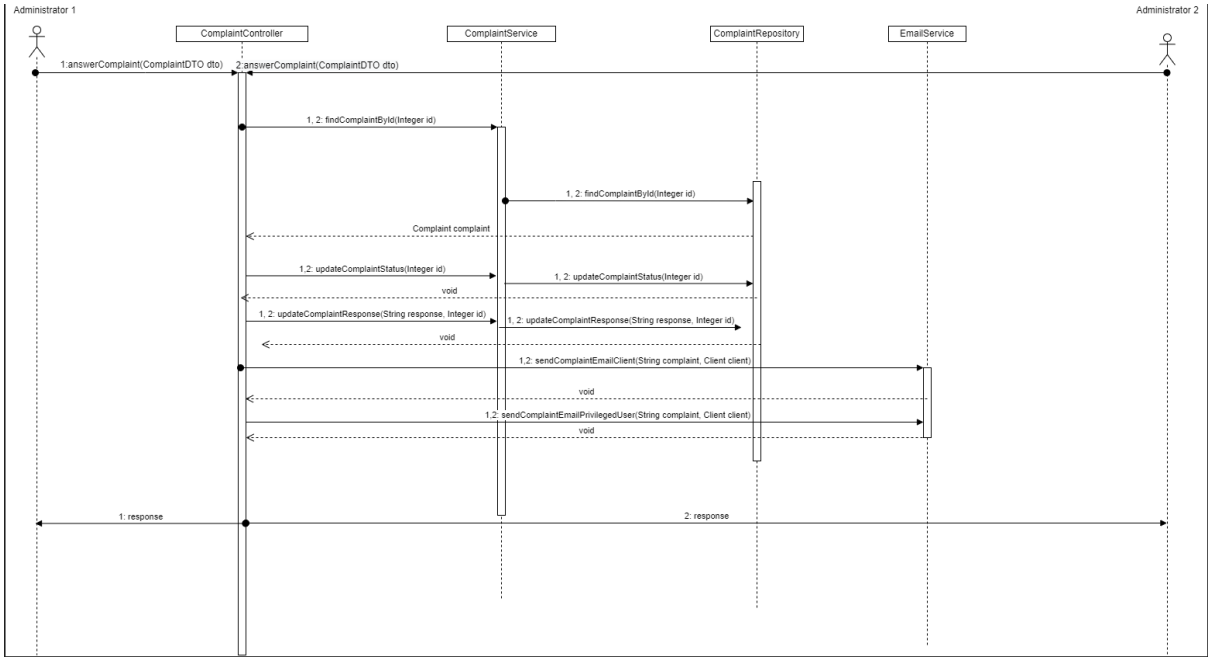


Rešenje: Ovaj problem je rešen korišćenjem metode pesimističnog zaključavanja. U klasi *TerminationReasoningController*, u funkcijama *acceptTermination(TerminationChoiceDTO dto)* i *declineTermination(TerminationChoiceDTO dto)* je dodat *try/catch* blok nad funkcijom *findUserById(int id)* iz klase *UserService*. U *UserRepository*, iznad funkcije koja biva pozvana od strane malopre spomenute funkcije iz servisa, dodata je anotacija *@QueryHints({@QueryHint(name = "javax.persistence.lock.timeout", value ="0")})*, koja će baciti grešku *PessimisticLockingFailureException* (koja je obrađena u prethodno spomenutom *try/catch* bloku) u slučaju da više korisnika sistema odjednom pokuša da dobavi podatke iz baze. Ovo je omogućeno dodavanjem anotacije *@Lock(LockModeType.PESSIMISTIC_WRITE)*, koja zabranjuje i čitanje i pisanje podataka sve dok se ne završi prethodna transakcija. U slučaju da dođe do prethodno spomenutog slučaja, vraća se greška 409 (konflikt), koja se aktivira prilikom dolaska do *PessimisticLockingFailureException* greške.

Situacija 2:

Dva (ili više) administratora pokušavaju u isto vreme da odgovore na jednu žalbu, što podrazumeva da odgovore na žalbu slobodnim tekstom i da se nakon toga pošalje email. Primer posledice ovog konflikta može biti da korisniku, koji je podneo žalbu, na email adresu stignu dva različita odgovora administratora, iako samo treba jedan administrator da odgovori na jednu žalbu.

Tok akcije: Dva administratora istovremeno pritiskaju na dugme za odgovaranje na žalbu korisnika nakon što su uneli svoj odgovor u slobodnoj formi. Na server stižu oba zahteva, a ako oba prođu, postoji mogućnost da će se odgovoriti na žalbu dva ili više puta, što bi značilo da će se korisniku poslati više mejlova sa odgovorima na žalbe, kao i da će se odgovor žalbe ,kao atribut u bazi, nekoliko puta izmeniti.



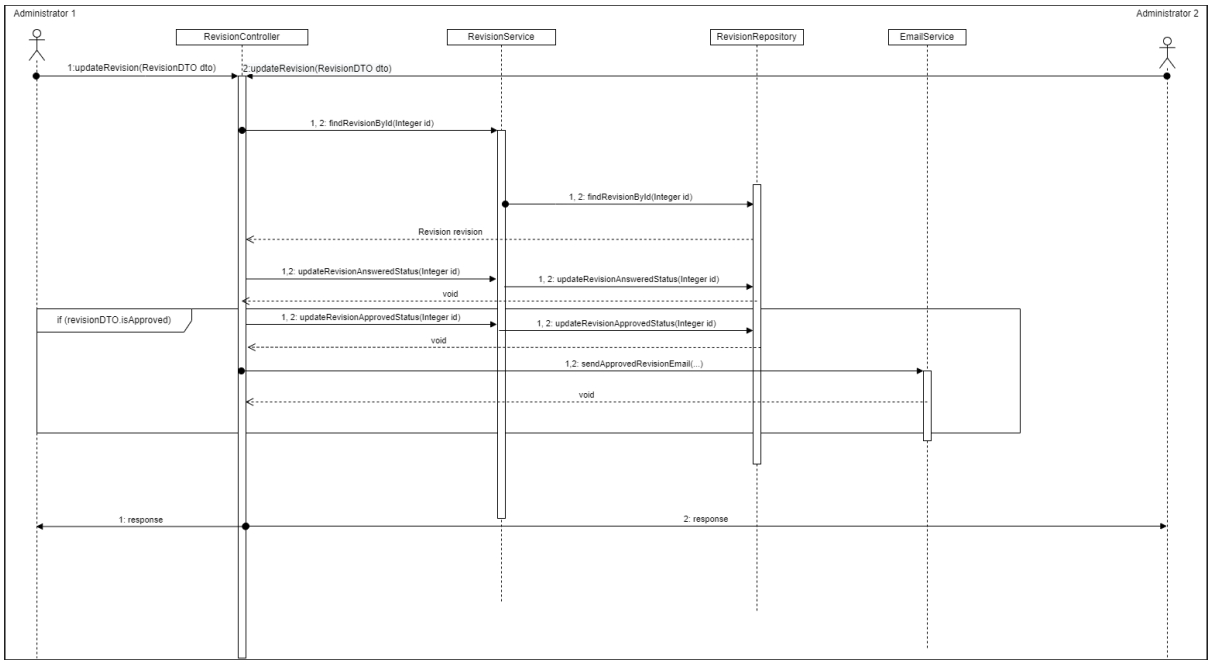
Rešenje: Ovaj problem je rešen korišćenjem metode pesimističnog zaključavanja. U klasi *ComplaintController*, u funkciji *answerComplaint(ComplaintDTO dto)* je dodat *try/catch* blok nad funkcijom *findComplaintById(int id)* iz klase *ComplaintService.* U *ComplaintRepository*, iznad funkcije koja biva pozvana od strane malopre spomenute funkcije iz servisa, dodata je anotacija *@QueryHints({@QueryHint(name = "javax.persistence.lock.timeout", value ="0")})*, koja će baciti grešku *PessimisticLockingFailureException* (koja je obrađena u prethodno spomenutom *try/catch* bloku) u slučaju da više korisnika sistema odjednom pokuša da dobavi podatke iz baze. Ovo je omogućeno dodavanjem anotacije *@Lock(LockModeType.PESSIMISTIC_WRITE)*, koja zabranjuje i čitanje i pisanje podataka sve dok se ne završi prethodna transakcija. U slučaju da dođe do prethodno spomenutog slučaja, vraća se greška 409 (konflikt), koja se aktivira prilikom dolaska do *PessimisticLockingFailureException* greške.

Situacija 3:

Dva (ili više) administratora pokušavaju u isto vreme da odgovore na jednu reviziju, što podrazumeva da pritisnu dugme da prihvataju ili odbijaju reviziju. U slučaju da je revizija prihvaćena, šalje se mejl klijentu koji je napisao reviziju i korisniku, na čiji je servis revizija napisana. Primer posledice ovog

konflikta može biti da jedan administrator prihvati reviziju, dok drugi administrator može istu da odbije. U ovom slučaju bilo bi potrebno razrešiti da li će se u prosečnu ocenu servisa i korisnika uračunati ocena prihvaćene revizije, kao i da li će se poslati odgovor na reviziju na mejl.

Tok akcije: Dva administratora istovremeno pritiskaju dugme za prihvatanje ili odbijanje revizije. Na server stižu oba zahteva, a ako oba prođu, postoji mogućnost da će se jedna revizija prihvatiti, a druga odbiti, što bi predstavljalo problem iz razloga što bi se prvobitno uračunala ocena revizije u prosečnu ocenu servisa i korisnika čiji je servis. U obrnutom slučaju, prvobitno bi se poslali mejlovi da je revizija prihvaćena, a naknadno bi ipak ispalo da nije.



Rešenje: Ovaj problem je rešen korišćenjem metode pesimističnog zaključavanja. U klasi *RevisionController*, u funkciji *updateRevision(RevisionDTO dto)* je dodat *try/catch* blok nad funkcijom *findRevisionById(int id)* iz klase *RevisionService.* U *UserRepository*, iznad funkcije koja biva pozvana od strane malopre spomenute funkcije iz servisa, dodata je anotacija *@QueryHints({@QueryHint(name = "javax.persistence.lock.timeout", value ="0")})*, koja će baciti grešku *PessimisticLockingFailureException* (koja je obrađena u prethodno spomenutom *try/catch* bloku) u slučaju da više korisnika sistema odjednom pokuša da dobavi podatke iz baze. Ovo je omogućeno dodavanjem anotacije *@Lock(LockModeType.PESSIMISTIC_WRITE)*, koja zabranjuje i čitanje i pisanje podataka sve dok se ne završi prethodna transakcija. U slučaju da dođe do prethodno spomenutog slučaja, vraća se greška 409 (konflikt), koja se aktivira prilikom dolaska do *PessimisticLockingFailureException* greške.