Konkurentni pristup resursima u bazi

Nikola Damjanović SW 71/2019

Situacija 1)

Postoji mogućnost da dva ili više administratora pokušaju da odgovore na isti zahtev za brisanje naloga istovremeno. Dolazi do konflikta u slučaju da su dva admina doneli različitu odluku i onda se dešava data-race čija će odluka biti finalna. Da bi ovo sprečili, koristićemo optimistični lock.

Tok akcija koji dovodi do ove konfliktne situacije je prihvatanje ili odbijanje zahteva za brisanje naloga od strane dva administratora istovremeno. Na server stižu oba zahteva i ako se oba izvrše finalno stanje korisničkog naloga nije determinističko i korisniku će stići više email-ova sa različitim porukama.

Za rešenje ovog problema implementirani su elementi za optimističko zaključavanje. U AdminController klasi kod funkcija za obradu zahteva za brisanje naloga dodati su try/catch blokovi koji hvataju ObjectOptimisticLockingFailureException i obaveštavaju admina da je došlo do konflikta. Takođe, da bi optimistično zaključavanje funkcionisalo, u User i DeletionRequest klasu su dodati Long version atributi sa anotacijom @Version kako bi JPA Hibernate imao mogućnost da baci ObjectOptimisticLockingFailureException.

Situacija 2)

Postoji mogućnost da dva ili više administratora pokušaju da odgovore na istu žalbu od poslovnih korisnika za brisanje naloga istovremeno. Dolazi do konflikta u slučaju da su dva admina doneli različitu odluku i onda će se desiti da se klijent penalizuje iako je drugi admin doneo odluku da se ne prihvati žalba poslovnog klijenta.

Tok akcija koji dovodi do ove konfliktne situacije je prihvatanje ili odbijanje žalbe od poslovnih korisnika od strane dva administratora istovremeno. Na server stižu oba zahteva i ako se oba izvrše finalno stanje klijenta koji je prijavljen nije determinističko, korisnik će biti penalizovan i korisniku i poslovnom korisniku će stići više email-ova sa različitim porukama.

Za rešenje ovog problema implementirani su elementi za optimističko zaključavanje. U AdminController klasi kod funkcija za obradu žalbi poslovnih korisnika dodati su try/catch blokovi koji hvataju ObjectOptimisticLockingFailureException i obaveštavaju admina da je došlo do konflikta. Takođe je dodat Long version atribut sa anotacijom @Version, u Report klasu, kako bi se mogao baciti ObjectOptimisticLockingFailureException.

Situacija 3)

Postoji mogućnost da dva ili više administratora pokušaju da odgovore na isti zahtev za registraciju poslovnog korisnika istovremeno. Dolazi do konflikta u slučaju da su dva admina doneli različitu odluku i onda će se desiti da se poslovni klijent obavesti o (ne)uspešnosti zahteva za registraciju putem email-a.

Tok akcija koji dovodi do ove konfliktne situacije je prihvatanje ili odbijanje zahteva za registraciju poslovnog korisnika istovremeno od strane dva administratora. Na server stižu oba zahteva i ako se oba izvrše finalno stanje poslovnog klijenta čiji zahtev za registraciju nije deterministično, korisnik će biti „uključen“ i stići će mu više email-ova sa različitim porukama.

Za rešenje ovog problema implementirani su elementi za optimističko zaključavanje. U AdminController klasi kod funkcija za obradu registracija poslovnih korisnika dodati su try/catch blokovi koji hvataju ObjectOptimisticLockingFailureException i obaveštavaju admina da je došlo do konflikta. Takođe je dodat Long version atribut sa anotacijom @Version, u RegistrationRequest klasu pored postojećeg @Version Long version atributa u User klasi, kako bi se mogao baciti ObjectOptimisticLockingFailureException.