STUDENT 3 (Rajko Zagorac)

U svih primerima koji slede korišćeno je optimističko zaključavanje. Ovo zaključavanje je izabrano prvenstveno jer su u konfliktnim situacijama korišćeni entiteti koji već postoje u bazi, pa je ovakav pristup moguć. Prilikom implementacije dodavana su @Verson polja kako bi hibernate mogao automatski da poveća vrednost verzije ukoliko primeti da je došlo do promene objekta koji se zaključava. Do promene verzije dolazi nakon čuvanja promena u bazi. Svako čuvanje podrazumeva da u međuvremenu nije došlo do promene, inače se baci **ObjectOptimisticLockingFailureException** kao znak da ne radimo sa najnovijom verzijom.

Zbog jednostavnosti prikaza, u navođenju konkretnih endpointova koji su gađani, koristiće se **baseUrl** umesto http://localhost:8092/bookingApp/

Konflikt1

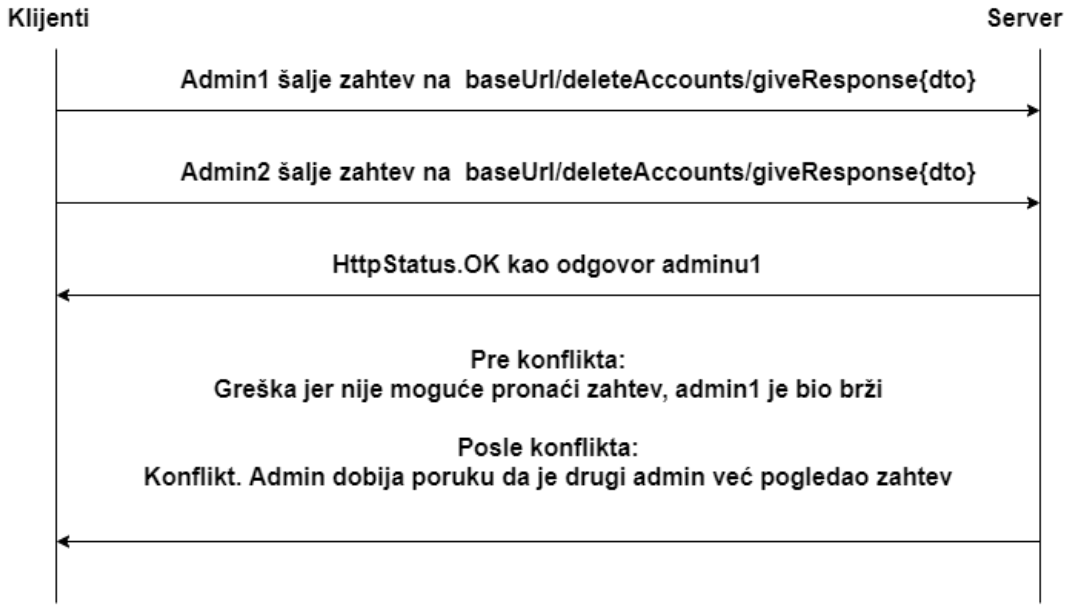
Na jedan zahtev za brisanje naloga može da odgovori samo jedan administrator sistema

Prilikom odgovaranja na zahteve za brisanjem naloga, konflikt se javlja ukoliko dva administratora pokušaju da obrade zahtev u isto vreme. Konflikt je detektovan tako što se odgovor u slučaju odbijanja zahteva slao korisniku dva puta na mejl. U slučaju brisanja, jedan admin bi stigao da obriše korisnika a drugi bi dobio grešku jer bi pokušao da briše nešto što ne postoji.

Kontroler koji se gadja: **DeleteAccountController**.

Metoda kontrolera je: **giveResponse (DeleteAccountRequestDTO d)**. U parametru funkcije se nalaze wrappovani zahtev za brisanje i odgovor administratora u vidu DTO objekta.

Ovaj problem je rešen optimističnim zaključavanjem. U klasi **DeleteAccountRequest** je dodato polje verzije, anotacijom @Version. U servisnoj metodi **giveResponse** klase **DeleteAccountService** je upotrebljeno zaključavanje postavljanjem zahteva kao obrađenog, na taj način se automatski povećava verzija, i admin koji je bio sporiji pokušaće da obradi zahtev za koji idalje ima pročitanu zastarelu verziju.



Konflikt2

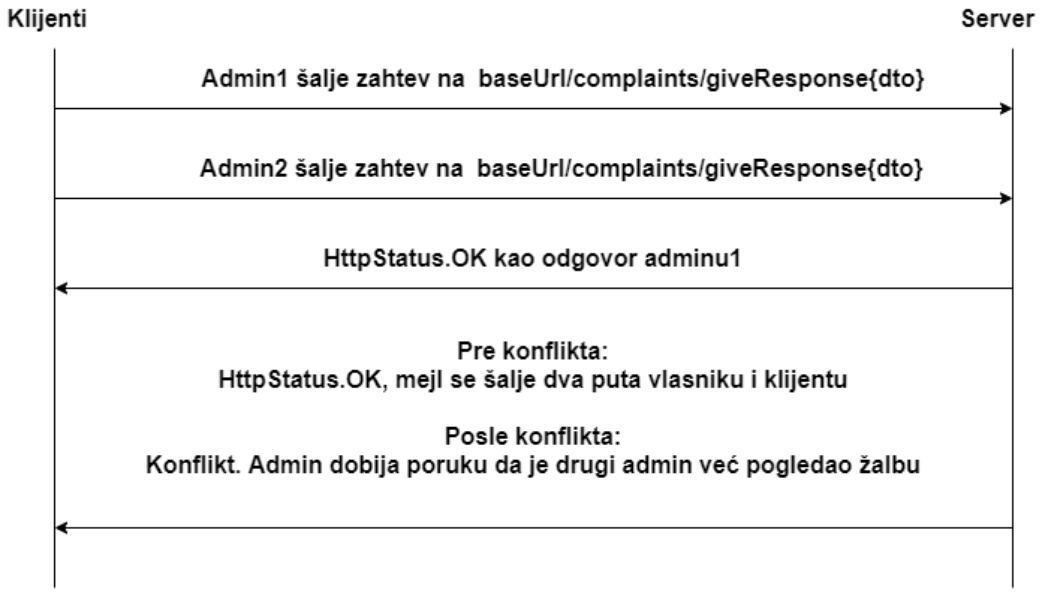
Na jednu žalbu može da odgovori samo jedan administrator sistema.

Prilikom odgovaranja na žalbe, konflikt se javlja ukoliko dva administratora pokušaju da odgovore na žalbu u isto vreme. Konflikt je detektovan tako što bi se vlasnicima i klijentima mejl slao više puta jer bi oba administratora detektovala zahtev kao neobrađen.

Kontroler koji se gađa: **ComplaintController**

Metoda kontrolera: **giveResponse(ComplaintReviewDTO c)**. U parametru funkcije se nalaze wrappovana žalba i odgovor administratora u vidu DTO objekta.

Ovaj problem je rešen optimističnim zaključavanjem. U klasi **Complaint** je dodato polje verzije, anotacijom @Version. U servisnoj metodi **giveResponse** klase **ComplaintService** je upotrebljeno zaključavanje postavljanjem žalbe kao obrađene, na taj način se automatski povećava verzija, i admin koji je bio sporiji pokušaće da obradi žalbu za koju idalje ima pročitanu zastarelu verziju.



Konflikt3

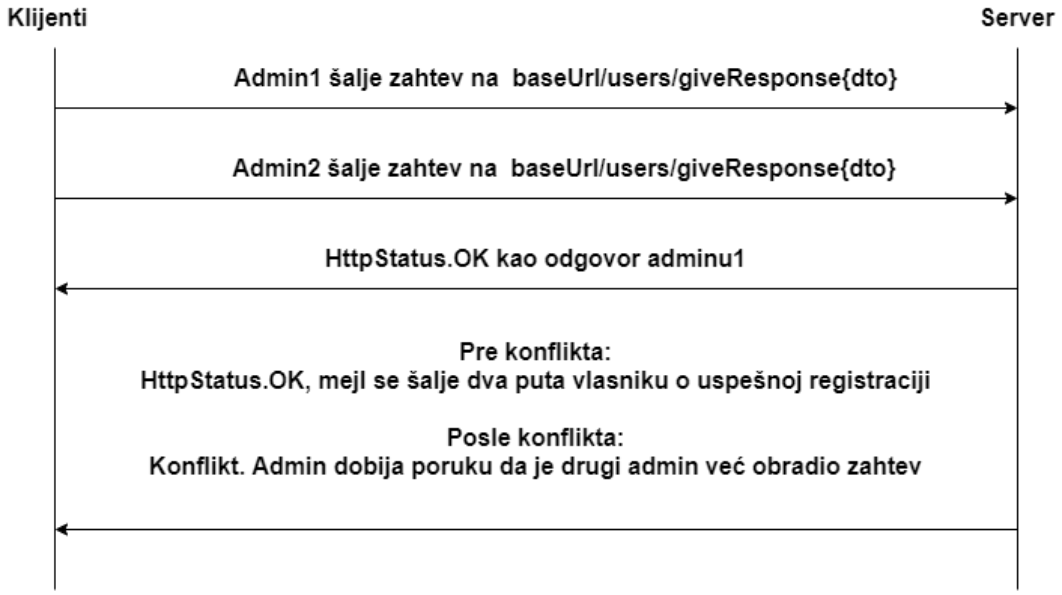
Na jedan zahtev za registracijom može da odgovori samo jedan administrator sistema.

Prilikom odgovaranja na zahteve za registracijom, konflikt se javlja ukoliko dva administratora pokušaju da odgovore na zahtev u isto vreme. Konflikt je detektovan tako što bi se vlasnicima mejl slao više puta jer bi oba administratora detektovala zahtev kao neobrađen.

Kontroler koji se gađa: **UserController**

Metoda kontrolera: **giveResponse(NewAccountRequestDTO c)**. U parametru funkcije se nalaze wrappovan nov zahtev za registracijom i odgovor administratora u vidu DTO objekta.

Ovaj problem je rešen optimističnim zaključavanjem. U klasi **User** je dodato polje verzije, anotacijom @Version. U servisnoj metodi **giveResponse** klase **UserService** je upotrebljeno zaključavanje postavljanjem zahteva za registracijom kao obrađenog, na taj način se automatski povećava verzija, i admin koji je bio sporiji pokušaće da obradi zahtev za koji idalje ima pročitanu zastarelu verziju.



Konflikt4

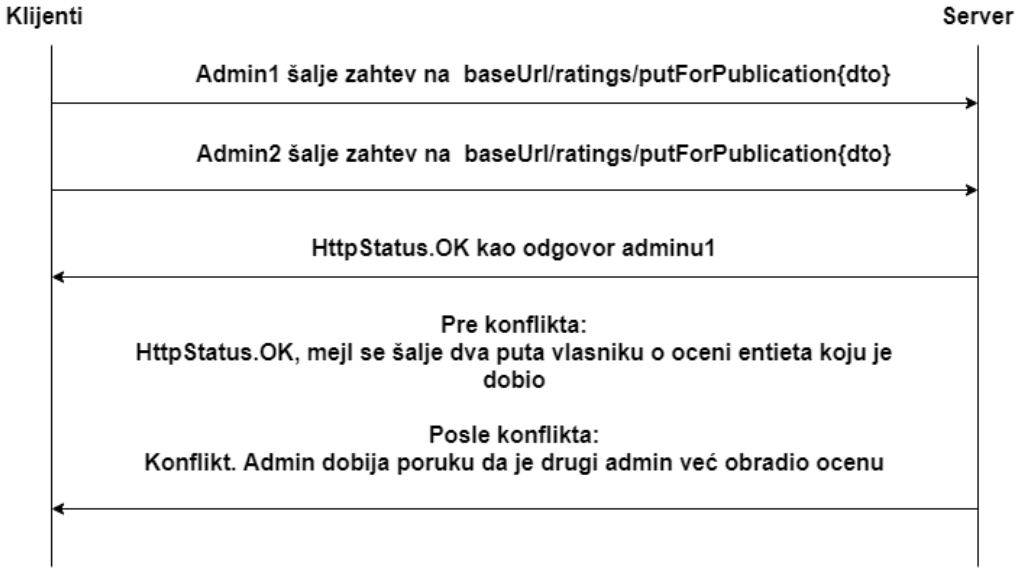
Na jednu ocenu klijenta može da odgovori samo jedan administrator sistema.

Prilikom pregledanja ocena klijenata, konflikt se javlja ukoliko dva administratora pokušaju da obrade ocenu u isto vreme. Konflikt je detektovan tako što bi se vlasnicima mejl slao više puta jer bi oba administratora detektovala ocenu kao neobrađenu.

Kontroler koji se gađa: **RatingController**

Metoda kontrolera: **puClientReviewForPublication(RatingReviewDTO c)**. U parametru funkcije se nalazi wrappova ocena i odgovor administratora u vidu DTO objekta.

Ovaj problem je rešen optimističnim zaključavanjem. U klasi **Rating** je dodato polje verzije, anotacijom @Version. U servisnoj metodi **setRatingForPublicationAndNotifyOwner** klase **RatingService** je upotrebljeno zaključavanje postavljanjem ocene kao obrađene, na taj način se automatski povećava verzija, i admin koji je bio sporiji pokušaće da obradi ocenu za koju idalje ima pročitanu zastarelu verziju.



Konflikt5

Na jedan izveštaj o rezervaciji može da odgovori samo jedan administrator sistema.

Prilikom pregledanja izveštaja rezervacija, konflikt se javlja ukoliko dva administratora pokušaju da obrade ocenu u isto vreme. Konflikt je detektovan tako što bi se vlasnicima i klijentima mejl slao više puta jer bi oba administratora detektovala izveštaj kao neobrađen.

Kontroler koji se gađa: **ReportController**

Metoda kontrolera: **giveResponse(CreatedReportReviewDTO c)**. U parametru funkcije se nalazi wrappov izveštaj i odgovor administratora u vidu DTO objekta.

Ovaj problem je rešen optimističnim zaključavanjem. U klasi **Report** je dodato polje verzije, anotacijom @Version. U servisnoj metodi **giveResponse** klase **ReportService** je upotrebljeno zaključavanje postavljanjem izveštaja kao obrađenog, na taj način se automatski povećava verzija, i admin koji je bio sporiji pokušaće da obradi izveštaj za koji idalje ima pročitanu zastarelu verziju.