KONFLIKTNE SITUACIJE – konkurentni pristup resursima u bazi

Rješavane su sledeće konfliktne situacije:

1. Na jedan zahtjev za brisanje naloga može da odgovori samo jedan administrator sistema
2. Na jednu žalbu može da odgovori samo jedan administrator sistema
3. Može da se napravi samo jedan loyalty program u jednom trenutku
4. Samo jedan administrator može da obriše korisnika sistema

Rješenja konfliktnih situacija:

1. Konfliktna situacija

Administrator može da potvrdi ili odbije zahtjev klijenta za brisanje naloga. Ako se desi da dva administratora istovremeno žele da potvrde ili odbiju zahtjev, doći će do konfliktne situacije gdje će jedan od njih pokušati da obriše nepostojećeg korisnika.

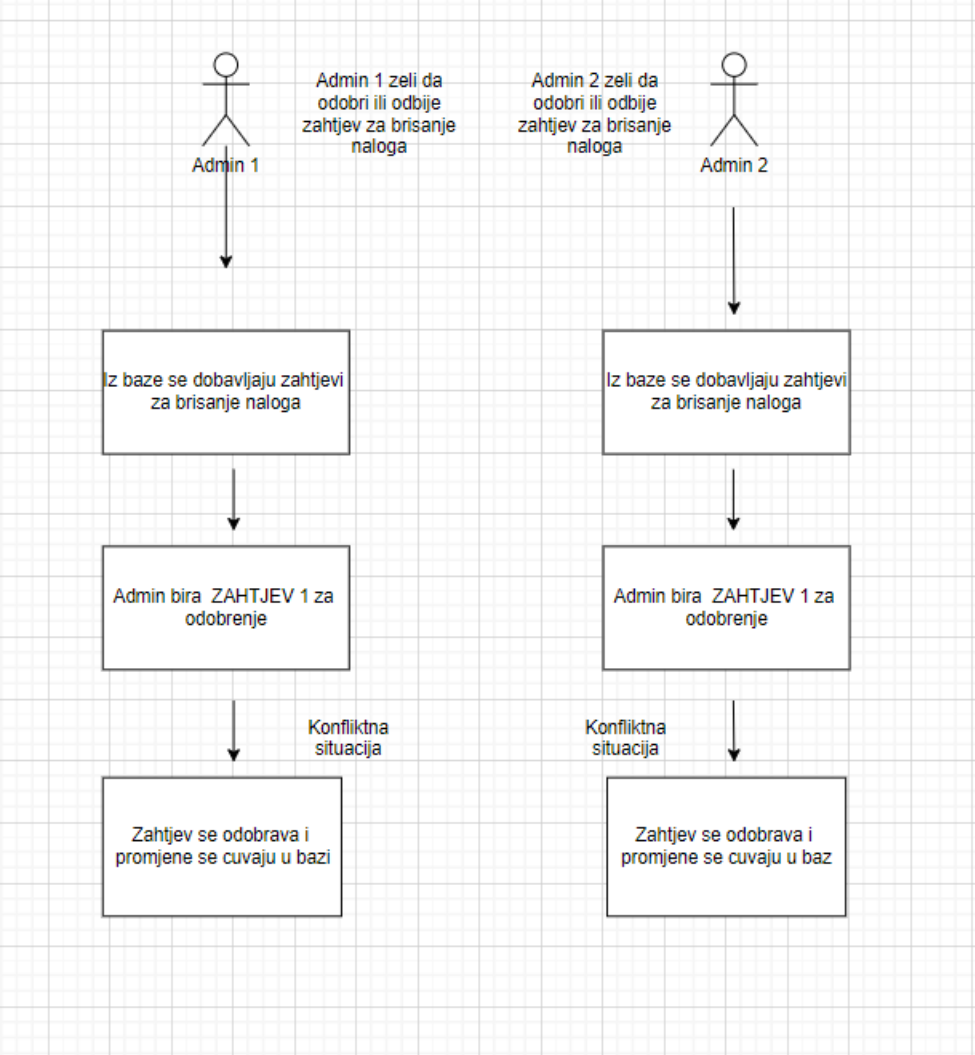
Konflikt je riješen tako što je korišćena anotacija @Transactional, pesimističko zaključavanje i provjera izuzetaka unutar metoda za prihvatanje i odbijanje zahtjeva, što je prikazano na slikama ispod.

```
@Lock(LockModeType.PESSIMISTIC_WRITE)
@Query("select u from User u where u.id = :id")
@QueryHints({@QueryHint(name = "javax.persistence.lock.timeout", value ="10000")})
public User findLockById(@Param("id")Long id);
```

```
@Override
@Transactional(readOnly = false, propagation = Propagation.REQUIRES_NEW)
public User declineDeletingAccount(DeleteAccountRequestDTO dto) throws Exception {
```

```
@Override
@Transactional(readOnly = false, propagation = Propagation.REQUIRES_NEW)
public User acceptDeletingAccount(DeleteAccountRequestDTO dto) throws Exception {
```

Dijagram 1:



2. Konfliktna situacija:

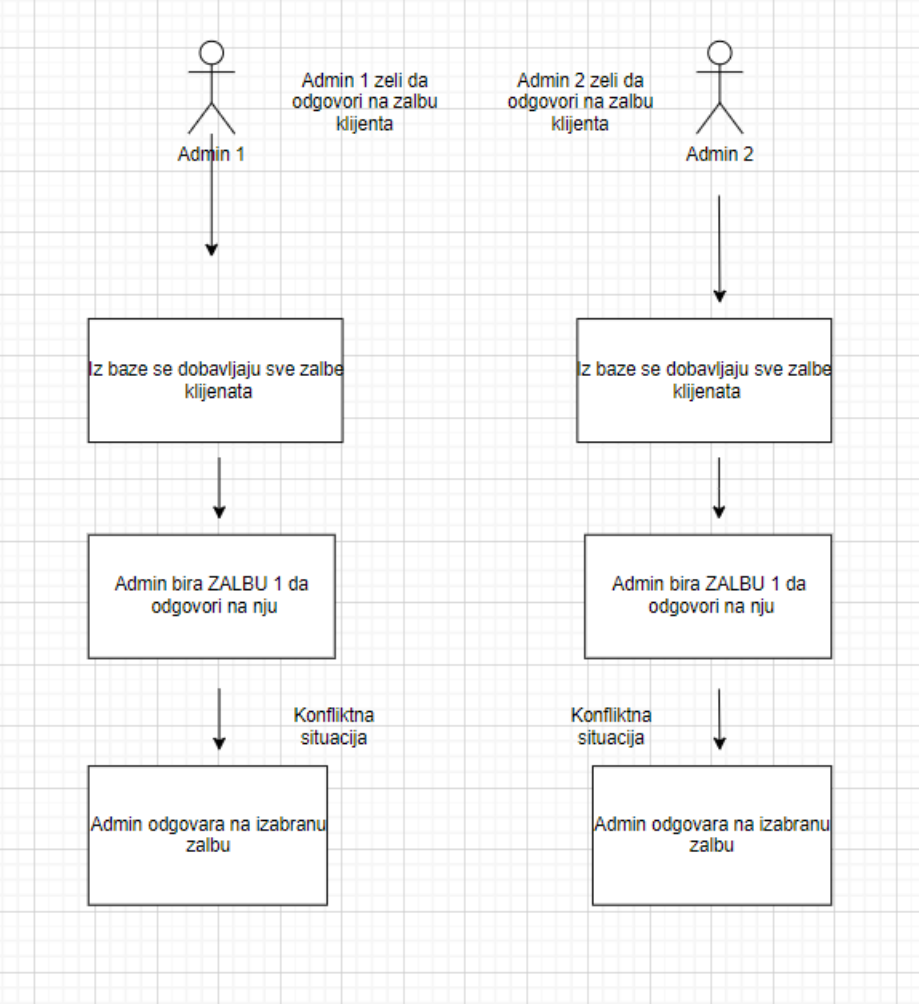
Administrator može da odgovori na žalbu klijenta. Ukoliko se desi da dva administratora žele da odgovore na istu žalbu , doći će do konfliktne situacije gdje će jedan od njih pokušati da odgovori na već odgovorenu žalbu.

Konflikt je riješen tako što je korišćena anotacija @Transactional, pesimističko zaključavanje i provjera izuzetaka unutar metoda za odgovaranje na žalbe, što je prikazano na slikama ispod.

```
@Lock(LockModeType.PESSIMISTIC_WRITE)
@Query("select c from CottageComplaint c where c.id = :id")
@QueryHints({@QueryHint(name = "javax.persistence.lock.timeout", value ="10000")})
public CottageComplaint findLockById(@Param("id")Long id);
```

```
@Override
@Transactional(readOnly = false, propagation = Propagation.REQUIRES_NEW)
public ComplaintResponseDTO answer(ComplaintAnswerDTO dto)  throws Exception{
```

Dijagram 2:

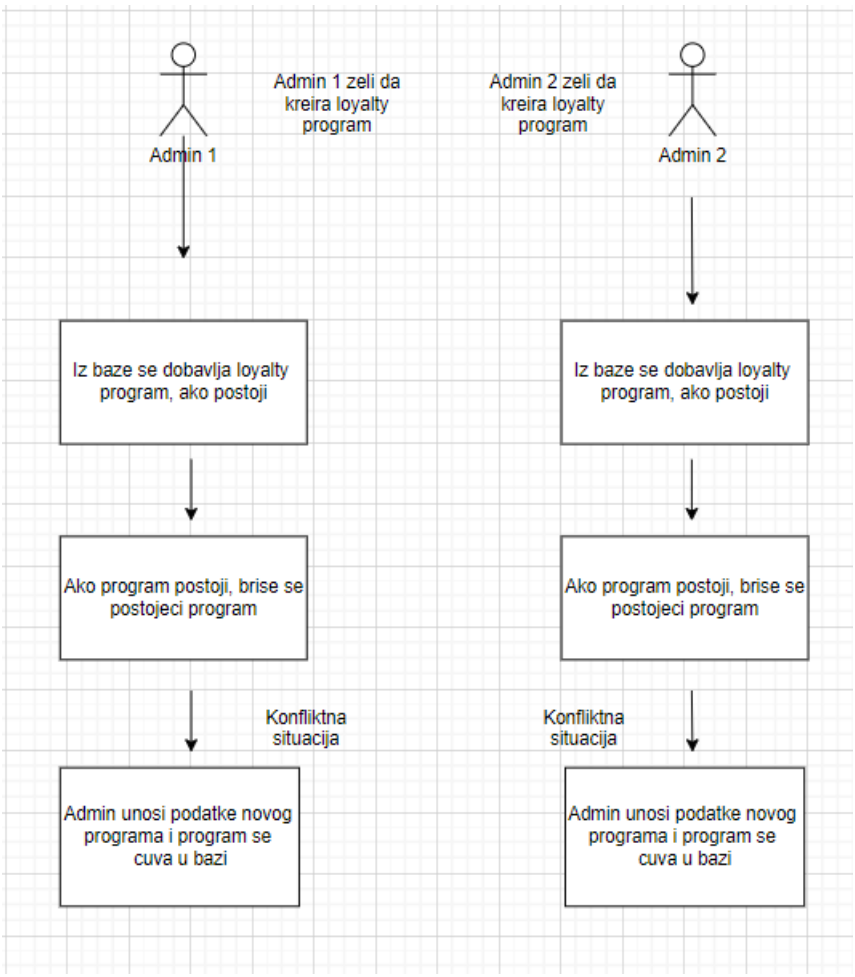


3. Konfliktna situacija:

Administrator može da kreira loyalty program. Ukoliko dva administratora pokušaju da istovremeno kreiraju loyalty program doći će do konfliktne situacije jer može da postoji samo jedan loyalty program.

Konflikt je riješen tako što je korišćena anotacija @Transactional i pesimističko zaključavanje, kao i u prethodnim primjerima.

Dijagram 3:



4. Konfliktna situacija:

Administrator može da obriše korisnika sistema. Ukoliko dva administratora istovremeno pokušaju da brišu istog korisnika, doći će do konfliktne situacije jer će jedan od admina pokušati da briše već obrisanog klijenta.

Konflikt je riješen tako što je korišćena anotacija @Transactional i pesimističko zaključavanje, kao i u prethodnim primjerima.

Dijagram 4: