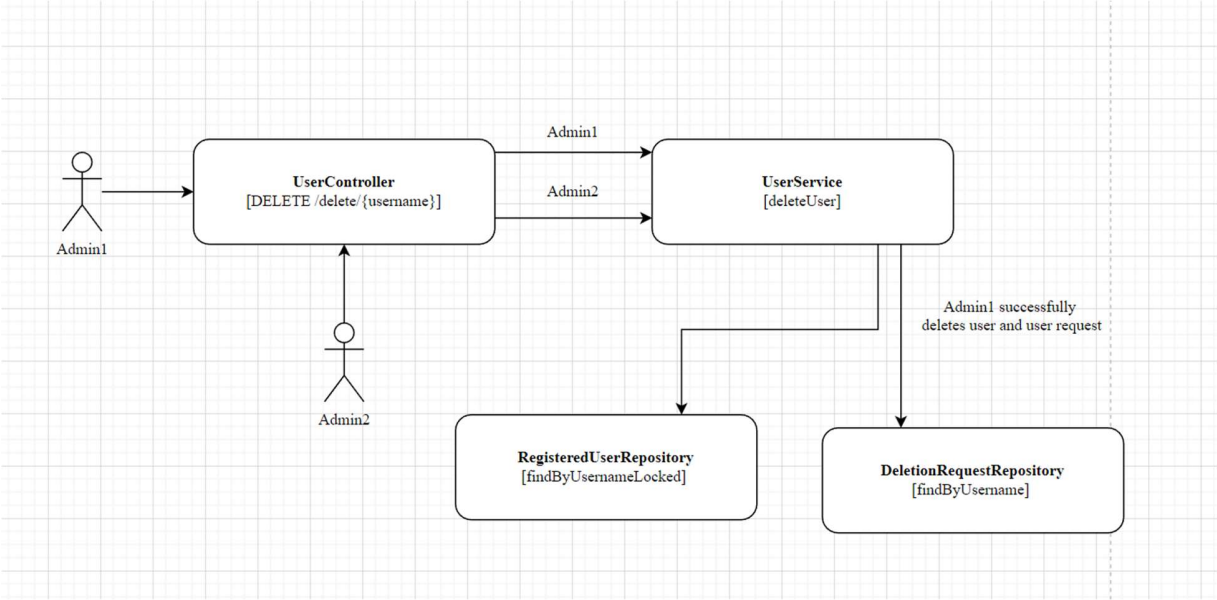
4.4 Konkurentni pristup resursima u bazi

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

Nakon što korisnici sistema naprave zahtev za brisanje naloga uz objašnjenje, on se čuva u bazi I čeka na administratorovo odobrenje. Administrator može da odbija ili prihvati zahtev.

Administratoru se ne prikazuju već odobreni zahtevi jer se brišu iz baze nakon njegove odluke. Ukoliko dva različita administratora pokušaju u isto vreme da reše slučaj isti slučaj zahteva za brisanje naloga dolazi do konfliktne situacije.



Konfliktna situacija je rešena pomoću:

- Metoda koju poziva kontroler je označena sa anotacijom @Transactional, a telo metode je obuhvaćeno sa blokom try/catch

```
3 usages  KisicM
@Transactional
@Override
public boolean deleteUser(String username) {
    try {
        logger.info("Deleting user: " + username);
        RegisteredUser user = userRepository.findByUsernameLocked(username);
        logger.info("User found: " + user.getUsername());
        ProfileDeletionRequest request = deletionRequestRepository.findByUsername(username);
        if (request != null)
            deletionRequestRepository.delete(request);
        if(user == null) return false;
        userRepository.delete(user);
        return true;
    } catch (Exception e) {
        throw new RuntimeException(e);
    }
}
```

- Metode userRepository.findByUsernameLocked I deletionRequestRepository.findByUsername su pesimistički zaključane po sledećem principu:

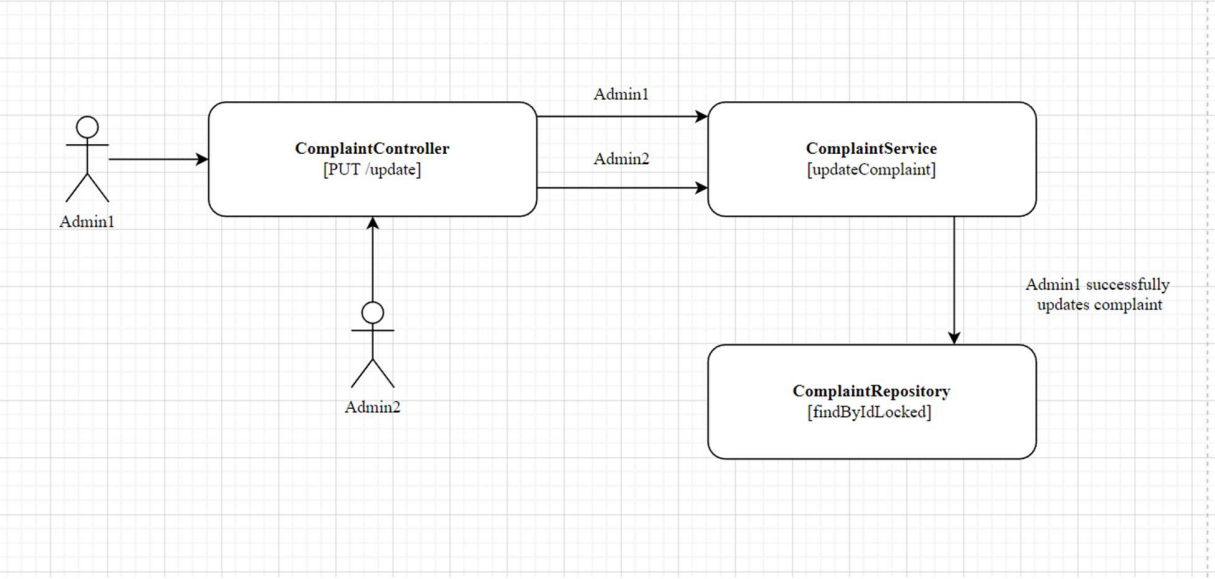
```
KisicM
@Lock(LockModeType.PESSIMISTIC_WRITE)
@QueryHints(@QueryHint(name = "javax.persistence.lock.timeout", value = "0"))
ProfileDeletionRequest findByUsername(String username);
```

2. Na jednu žalbu može da odgovori samo jedan administrator sistema

Klijenti mogu da naprave žalbu za žavršene rezevacije kao što i vlasnici i instruktori mogu da naprave žalbu za klijenta (ukoliko se klijent nije pojavio automatski dobija 1 penal).

Sve žalbe odlaze kod administratora na odobravanje. Ako admin odobri žalbu vlasnika ili instruktora, obaveštavaju se preko emaila klijent i vlasnik/instruktor, a klijent dobija 1 penal. Ako admin odobri žalbu klijenta, admin mora da odgovori na žalbu i prosledi ga na email klijentu i vlasniku/instruktoru.

Do konfliktne situacije dolazi ukoliko dva administratora pokušaju istovremeno da odgovore na istu žalbu.



Konfliktna situacija je rešena pomoću:

- Metoda koju poziva kontroler je označena sa anotacijom @Transactional, a telo metode je obuhvaćeno sa blokom try/catch

```
3 usages  KisicM
@Transactional
@Override
public Complaint updateComplaint(Complaint complaint) {
    try {
        Complaint updatedComplaint = complaintRepository.findByIdLocked(complaint.getId());
        if(updatedComplaint == null) return null;
        updatedComplaint.setReviewed(true);
        if(updatedComplaint.isForPenalty())
            penaltyService.createPenalty(new Penalty(complaint.getSubjectUsername()));

        return complaintRepository.save(updatedComplaint);
    } catch (Exception e) {
        throw new RuntimeException(e);
    }
}
```

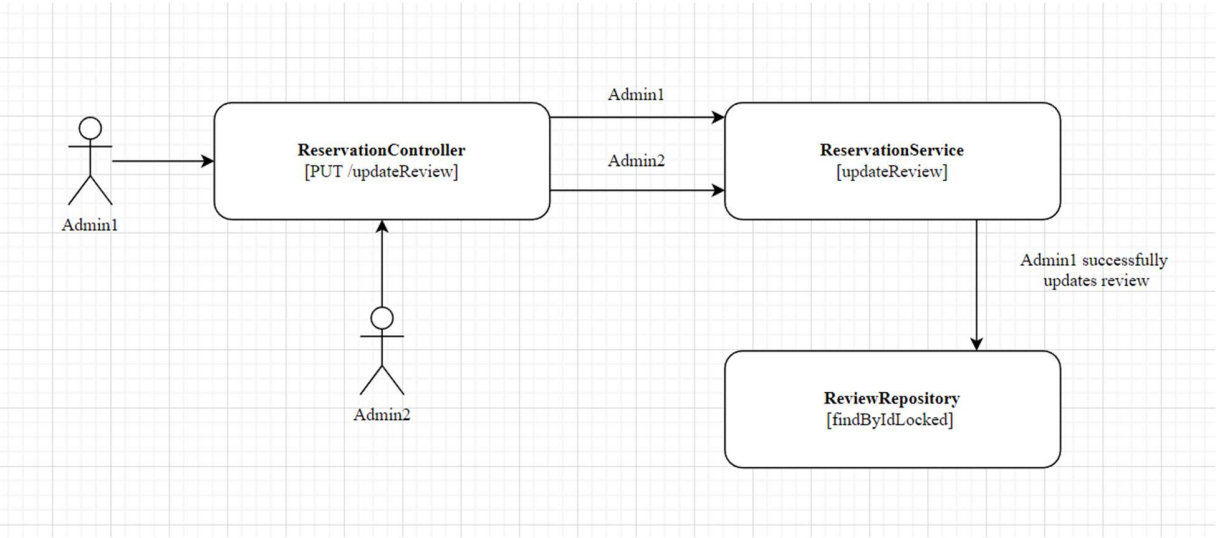
- Metoda findByIdLocked je zaključana optimistički po sledećem principu:

```
1 usage   KisicM
@Lock(LockModeType.OPTIMISTIC_FORCE_INCREMENT)
@Query("SELECT c FROM Complaint c WHERE c.id = :id")
Complaint findByIdLocked(Long id);
```

3. Na jednu reviziju može da odgovori samo jedan administrator sistema

Nakon završene rezervacije klijenti mogu da naprave reviziju za vikendicu i instruktora. Administratori vide sve revizije koje još nisu ocenili i mogu da dopuste njihovo prikazivanje ili ga odbiju.

Do konfliknte situacije dolazi ukoliko dva različita admina pokušaju da menjaju istu reviziju.



Konfliktna situacija je rešena pomoću:

- Metoda koju poziva kontroler je označena sa anotacijom @Transactional, a telo metode je obuhvaćeno sa blokom try/catch

```java
@Transactional
@Override
public void updateReview(Review review) {
    try {
        Review reviewToUpdate = this.reviewRepository.findByIdLocked(review.getReservationId());
        if(reviewToUpdate == null) return;
        reviewToUpdate.setApproved(review.isApproved());
        reviewToUpdate.setPublic(review.isPublic());
        this.reviewRepository.save(reviewToUpdate);
    } catch (Exception e) {
        throw new RuntimeException(e);
    }
    if(!review.isPublic()) return;
```

- Metoda findByIdLocked je zaključana optimistički po sledećem principu:

```java
    1 usage  KisicM
    @Lock(LockModeType.OPTIMISTIC_FORCE_INCREMENT)
    @Query("select r from Review r where r.reservationId = :id")
    Review findByIdLocked(Long id);
}
```