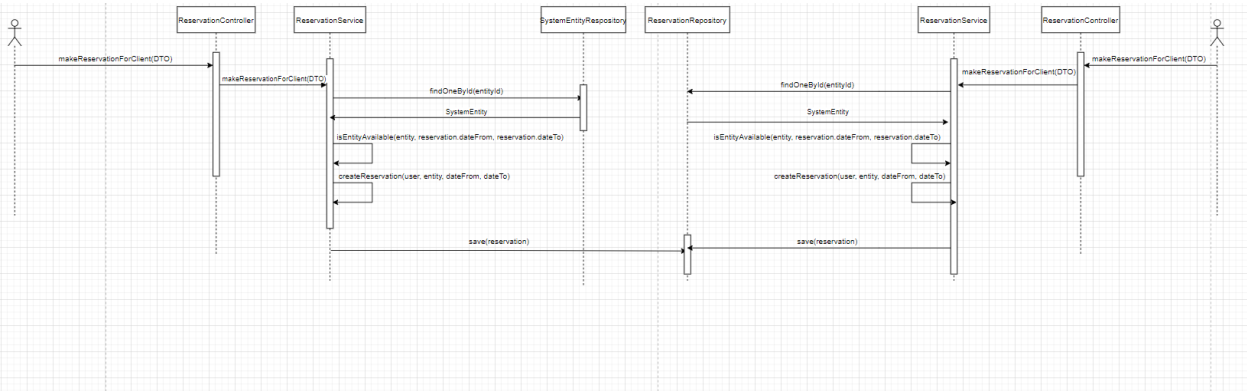
Konkurentni pristup resursima u bazi

Konflikt 1 – vlasnik vikendice/broda ili instruktor ne može da napravi rezervaciju u isto vreme kad i drugi klijent

Opis problema

Problem nastaje kada vlasnik vikendice zele da napravi rezervaciju u dogovoru sa klijentom. Postoji konflikt kada dva vlasnika ili vlasnik I klijent rezervisu isti entitet u istom ili preklapajucem periodu.

Dijagram opisanog problema.



Resenje

Problem je moguce resiti pomocu pesimisticnog ili optimisticnog zakljucavanja baze. U ovom slucaju zakljucava se baza SystemEntityRepository pri trazenju. Dodatno se dodaje anotacija *@Transactional* iznad metode *makeReservationForClient*.

Slika resenja konflikta

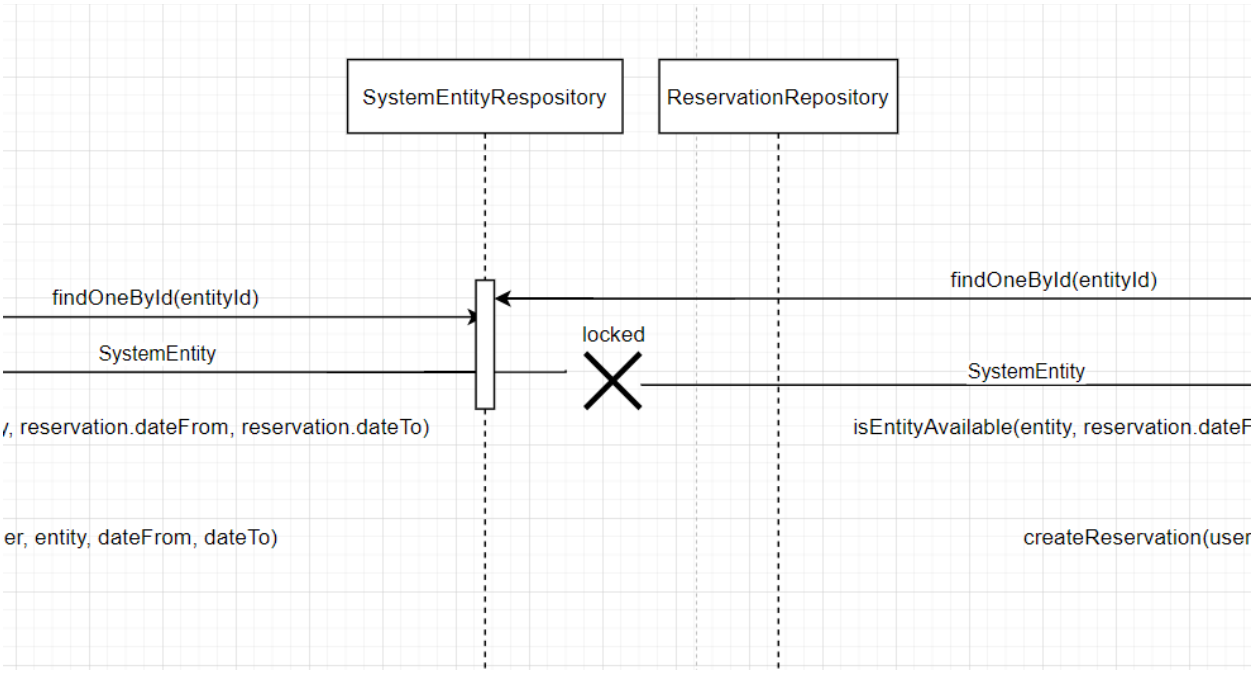
```
@Lock(LockModeType.PESSIMISTIC_WRITE)
@Query("SELECT e FROM SystemEntity e where e.id=:id")
@QueryHints({@QueryHint(name = "javax.persistence.lock.timeout", value = "1000")})
SystemEntity getLockedEntity(Integer id);
```

Slika resenja konflikta unutar metode

```
@Transactional
public boolean makeReservationForClient(ReservationRequestDTO reservationRequestDTO) {
    String email = reservationRequestDTO.getUsername();
    User u = userRepository.findOneByUsername(email);
    if(u == null) return false;
    if(!u.getRoles().get(0).getName().equals("ROLE_CLIENT")) return false;
    if(u.getUserPenalties() >=3) return false;
    SystemEntity entity = systemEntityRepository.findOneById(reservationRequestDTO.getEntityId());
    SystemEntity entityToReserve;
```

Ovako blokiramo pristup bazi dok prvi zahtev nije zavrsen. Tek onda se izvrsava drugi zahtev.

Dijagram koji prikazuje situaciju unutar baze

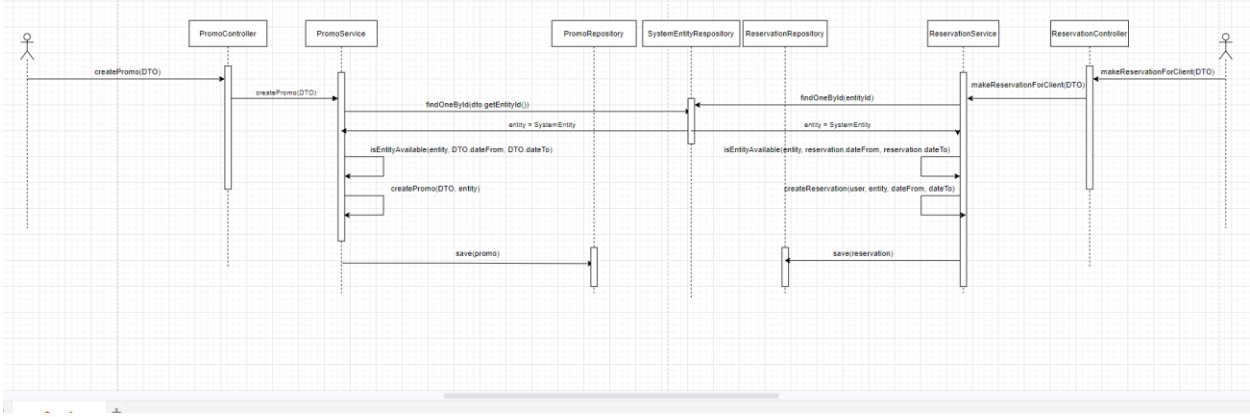


Konflikt 2 – klijent ne moze da rezervise entitet sve dok vlasnik vikendice/broda instruktor ne zavrsi sa kreiranjem akcije

Opis problema

Problem nastaje kada klijent zeli da napravi rezervaciju za neki period a u isto vreme vlasnik vikendice takodje kreira akciju za taj isti period.

Dijagram opisanog konflikta



Resenje

Problem je moguce resiti pomocu pesimisticnog ili optimisticnog zakljucavanja baze. U ovom slucaju zakljucava se baza SystemEntityRepository pri trazenju. Dodatno se dodaje anotacija *@Transactional* iznad metode *createPromoFromDTO* unutar servisa.

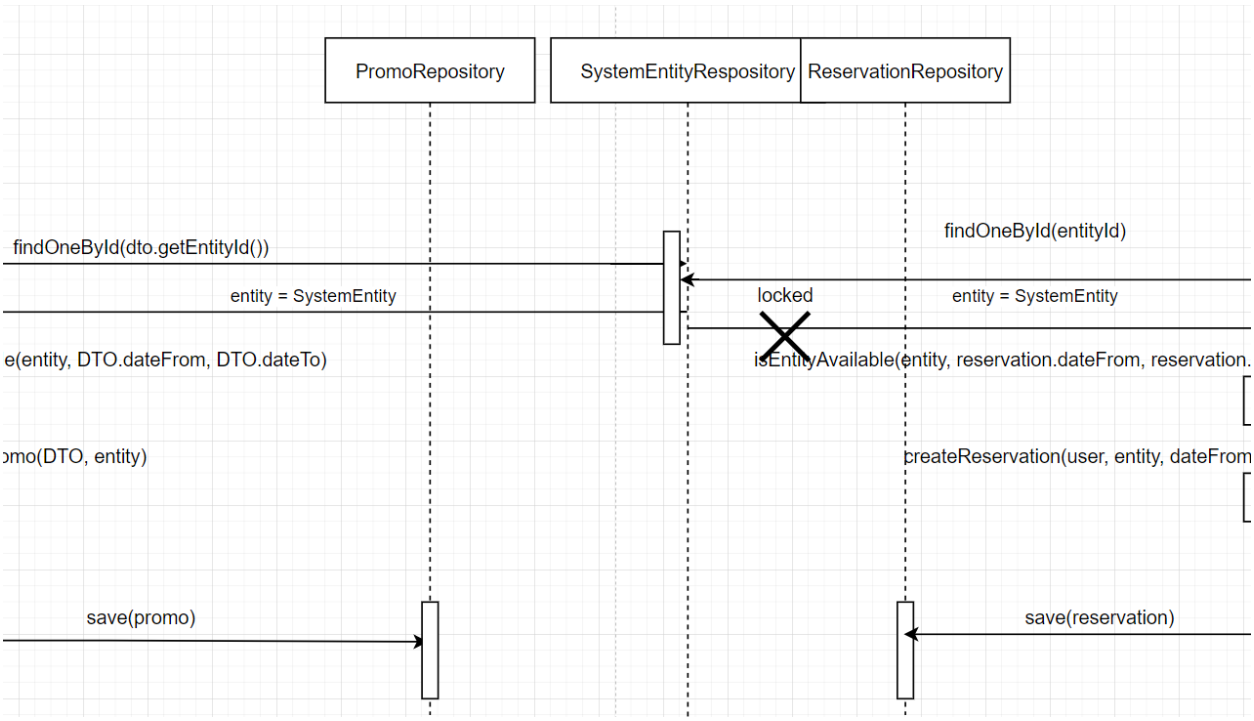
Slika resenja konflikta unutar baze

```
@Lock(LockModeType.PESSIMISTIC_WRITE)
@Query("SELECT e FROM SystemEntity e where e.id=:id")
@QueryHints({@QueryHint(name = "javax.persistence.lock.timeout", value = "1000")})
SystemEntity getLockedEntity(Integer id);
```

Slika resenja unutar metode

```
@Transactional
public boolean createPromoFromDTO(PromoDTO promoDTO ) {
    SystemEntity entity = systemEntityRepository.getLockedEntity(promoDTO.getSystemEntityId());
    DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm");
    LocalDateTime dateFrom = LocalDateTime.parse(promoDTO.getDateFrom().replace( target: "T", replacem
    LocalDateTime dateTo = LocalDateTime.parse(promoDTO.getDateTo().replace( target: "T", replacement: "
    if(reservationService.isEntityAvailable(entity, dateFrom, dateTo)) {
        promoRepository.save(new Promo(promoDTO, entity));
        for (User sub: entity.getSubscribers()) {
```

Ovako je zatvoren pristup bazi dok se ne zavrsi prvi zahtev. Dijagram konflikta

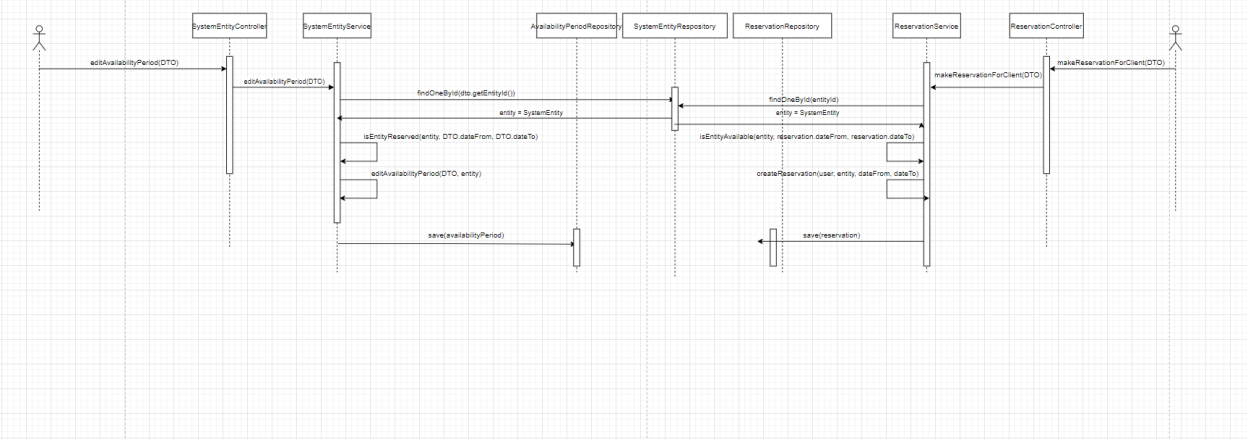


Konflikt 3 – rezervacija od strane klijenta ili vlasnika vikendice/broda I instuktora kada se istovremeno azurira period dostupnosti

Opis problema

Problem se javlja kada vlasnik vikendice/broda instruktor azurira period dostupnosti entiteta I istovremeno se kreira rezervacija od strane korisnika.

Dijagram opisanog problema



Resenje

Problem je moguce resiti pomocu pesimisticnog ili optimisticnog zakljucavanja baze. U ovom slucaju zakljucava se baza SystemEntityRepository pri trazenju. Dodatno se dodaje anotacija *@Transactional* iznad metode *editAvailabilityPeriod* unutar servisa.

Slika resenja unutar baze

```
@Lock(LockModeType.PESSIMISTIC_WRITE)
@Query("SELECT e FROM SystemEntity e where e.id=:id")
@QueryHints({@QueryHint(name = "javax.persistence.lock.timeout", value = "1000")})
SystemEntity getLockedEntity(Integer id);
```

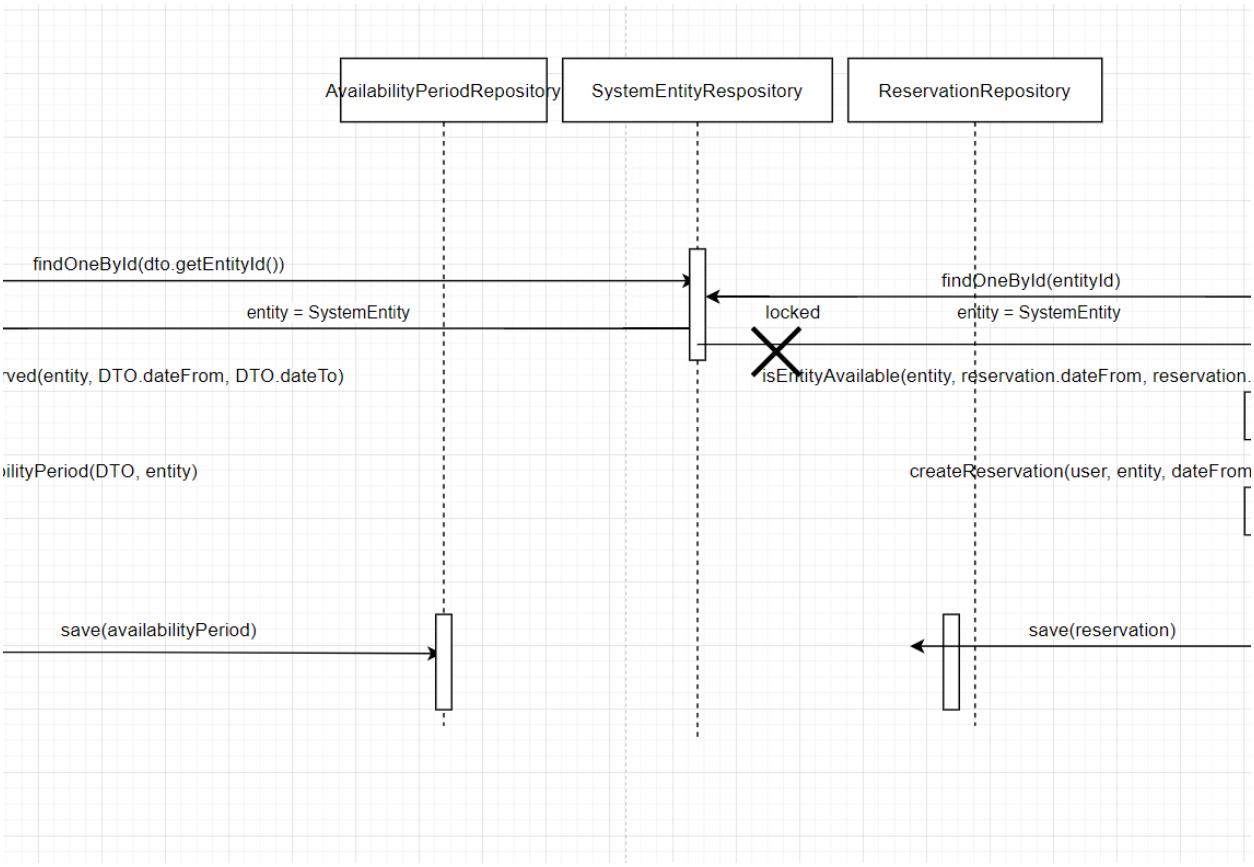
Slika resenja unutar metode

```
@Transactional
public boolean editAvailabilityPeriod(PeriodsDTO periodsDTO) {
    SystemEntity entity = systemEntityRepository.getLockedEntity(periodsDTO.getServiceID());

    Set<AvailabilityPeriod> availabilityPeriods = entity.getAvailabilityPeriod();
    for(AvailabilityPeriod period: availabilityPeriods) {
        if(!reservationService.isEntityAvailable(entity, period.getDateFrom(), period.getDateTo())) {
            return false;
        }
    }
```

Ovako se zakljucava pristup bazi dok se prvi zahtev ne obradi.

Dijagram resenja


AvailabilityPeriodRepository
SystemEntityRespository
ReservationRepository
findOneById(dto.getEntityId())
findOneById(entityId)
entity = SystemEntity
locked
entity = SystemEntity
rved(entity, DTO.dateFrom, DTO.dateTo)
isEntityAvailable(entity, reservation.dateFrom, reservation.
ilityPeriod(DTO, entity)
createReservation(user, entity, dateFrom
save(availabilityPeriod)
save(reservation)