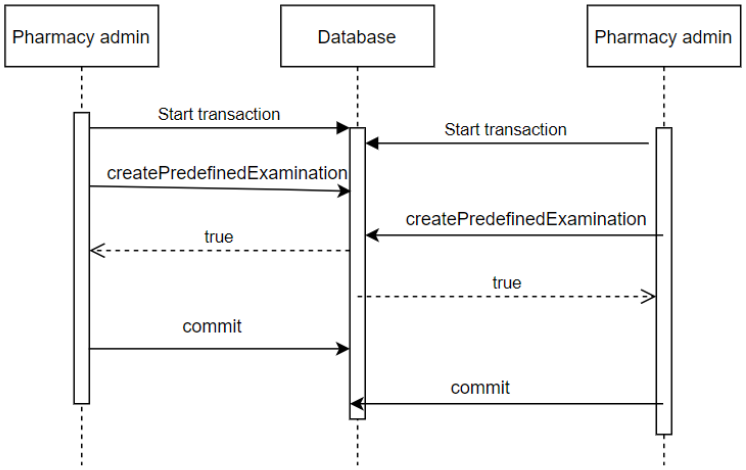
Konfliktne situacije – Student br.2

Dragana Todorović RA87-2017

1. ***Jedan dermatolog ne može istovremeno da bude prisutan na više različitih pregleda – Administrator apoteke***

Opis konfliktne situacije: Kada se uloguje administrator apoteke ima mogućnost kreiranja predefinisanih pregleda za dermatologe koji su zaposleni u apoteci u kojoj je on administrator. Problem može nastati ukoliko data apoteka ima više administratora koji istovremeno pokušavaju da definišu predefinisane pregleda za istog dermatologa u istom terminu.

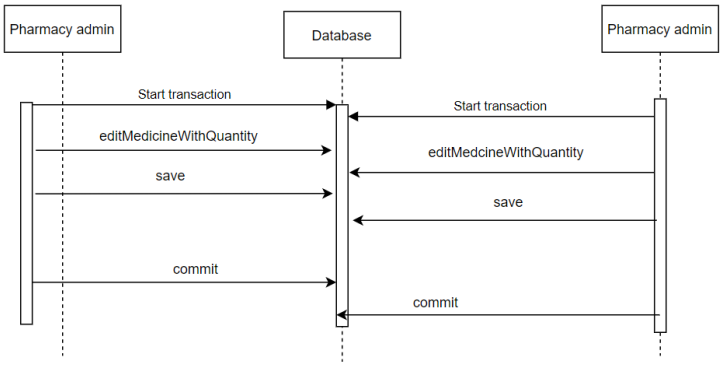


Rješenje: S obzirom da se radi o kreiranju potpuno nove torke za rješavanje ove konfliktne situacije iskorišteno je pesimističko zaključavanje nad torkom Dermatologist. Nad metolom findDermatologistByID koja se nalazi u DermatologistRepository dodata je anotacija `@Lock(LockModeType.PESSIMISTIC_WRITE)` kojom će se vršiti zaključavanje dermatologa kada se on izvlači iz baze na osnovu id-ja. Što znači da nijedna druga transakcija neće moći da se izvrši dok se ta ne završi, pa ukoliko dva administratora apoteke istovremeno pokušaju da kreiraju predefinisani pregled javiće se izuzetak.

```
@Lock(LockModeType.PESSIMISTIC_WRITE)
@Query("select d from Dermatologist d where d.id = :id")
@QueryHints({@QueryHint(name = "javax.persistence.lock.timeout", value="0")})
Dermatologist findDermatologistByID(@Param("id")Long id);
```

2. ***Više administratora apoteke pokušava da u isto vrijeme izmijeni količinu lijeka u apoteci***

   Opis konfliktne situacije: Kada se uloguje administrator apoteke ima mogućnost da pristupi lijekovima koje ta apoteka ima u ponudi kao i da im dodaje, i mijenja njihovu količinu. Prilikom izmjene lijeka unosi novu količinu za dati lijek. Konflikt može nastati ukoliko administrator apoteke pokuša da izmijeni količinu lijeka koja je prethodno izmijenjena od strane drugog administratora apoteke, ali on nije svjestan te izmjene jer se može desiti da nije osvježio stranicu, pa se verzija lijeka na frontu neće poklapati sa onom koja je u bazi .



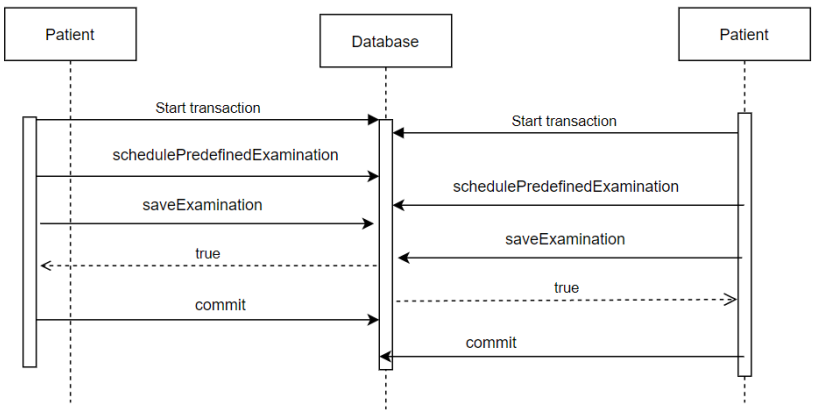
Rješenje: Kako ne bi došlo do gore navedene situacije primijenjeno je optimističko zaključavanje. U klasi MedicineQunatity dodat je atribut version (sa anotacijom @Version) koji čuva broj izmjena objekta klase MedicineQuantity pa samim tim prilikom izmjene lijeka u toj apoteci provjeravaće se da li se verzija pristigla sa frontenda poklapa sa već postojećom u bazi. Ukoliko se ne poklapa javiće izuzetak.

```java
public class MedicineWithQuantity {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private long id;

    @ManyToOne(fetch = FetchType.LAZY, cascade = CascadeType.PERSIST)
    private Medicine medicine;

    @Column(name = "quantity", nullable = false)
    private int quantity;

    @Version
    @Column(name = "version", nullable = false, columnDefinition = "int default 1")
    private long version;
```

```java
@Override
@Transactional(readOnly = false)
public void editMedicineWithQuatityInPharmacy(String email, long id, int quantity, Long v) {
    PharmacyAdmin pa = pharmacyAdminService.findPharmacyAdminByUser(userService.findByEmail(email));

    Pharmacy pharmacy = pa.getPharmacy();

//  MedicineWithQuantity medicineWithQuantity = this.medicineWithQuantityRepository.findByMedicineId(id);
    MedicineWithQuantity medicineWithQuantity = this.medicineWithQuantityRepository.findById(id).get();

    if(medicineWithQuantity.getVersion() != v) {
        throw new ObjectOptimisticLockingFailureException("Versions don't match", MedicineWithQuantity.class);
    }


    for(MedicineWithQuantity mq : pharmacy.getMedicineWithQuantity()) {
        if(mq.equals(medicineWithQuantity)) {
            mq.setQuantity(quantity);
        }
    }

    this.pharmacyRepository.save(pharmacy);
}
```

3. ***Pregledi koji su unaprijed definisani ne smiju biti rezervisani od strane više različitih korisnika***

   Opis konfliktne situacije: Kada se uloguje administrator apoteke ima mogućnost kreiranja predefinisanih pregleda za dermatologe koji su zaposleni u apoteci u kojoj je on administrator. Pacijentima se nudi mogućnost zakazivanja tih predefinisanih pregleda. Konfliktna situacija može nastati ukoliko više pacijenata pokušava u isto vrijeme da zakažu isti predefinisani pregled.



Rješenje: S obzirom da se radi o izmjeni pregleda, a ne o kreiranju novog, ovu konfliktnu situaciju je moguće riješiti koristeći optimističko zaključavanje, kao i u prethodnoj konfliktnoj situaciji dodavanjem version atributa u klasi DermatologistAppointment i dodavanjem Transactional anotacije za metode za zakazivanje predefinisanih pregleda.

```java
public class DermatologistAppointment {

    @Id
     @Column(name = "id")
     @GeneratedValue(strategy = GenerationType.IDENTITY)
      private Long id;

    @OneToOne(fetch = FetchType.EAGER, cascade = CascadeType.ALL)
    private Dermatologist dermatologist;

    @OneToOne(fetch = FetchType.EAGER, cascade = CascadeType.ALL)
    private Pharmacy pharmacy;

    @OneToOne(fetch = FetchType.EAGER, cascade = CascadeType.ALL)
    private Patient patient;

    @Column(name = "startDateTime", nullable = false)
    private LocalDateTime startDateTime;

    @Column(name = "duration", nullable = false)
    private int duration;

    @Column(name = "description")
    private String description;

    @ManyToOne(fetch = FetchType.EAGER, cascade = CascadeType.ALL)
    private Therapy therapy;

    @Version
    @Column(name = "version", nullable = false)
    private Long version;
```