Aca Simić RA-175/2017

Anotacije u kodu za sve transakcije I end pointi:

```
@Override
@Transactional(readOnly = false)
public Pharmacy change(PharmacyChangeDTO dto) {
```

```
@PreAuthorize("hasAnyRole('ROLE_PH_ADMIN')")
@PostMapping("/change")
public ResponseEntity<?> changePharmacy(@RequestBody PharmacyChangeDTO dto) {
```

```
@Lock(LockModeType.PESSIMISTIC_READ)
@QueryHints({@QueryHint(name = "javax.persistence.lock.timeout", value ="2000")})
MedicalStuff findByUserId(UUID userId);
```

```
@Lock(LockModeType.PESSIMISTIC_READ)
@QueryHints({@QueryHint(name = "javax.persistence.lock.timeout", value ="1000")})
Dermatologist findByLoginInfoEmail(String email);
```

```
@Version
private Long version;
```

```
@Service
@Transactional(readOnly = false)
public class DermatologistServiceImpl implements DermatologistService {
```

```
@PreAuthorize("hasAnyRole('ROLE_PH_ADMIN')")
@PostMapping("/createExamination")
public ResponseEntity<?> createExaminationForDermatologist(@RequestBody ExaminationCreateDTO dto){
```

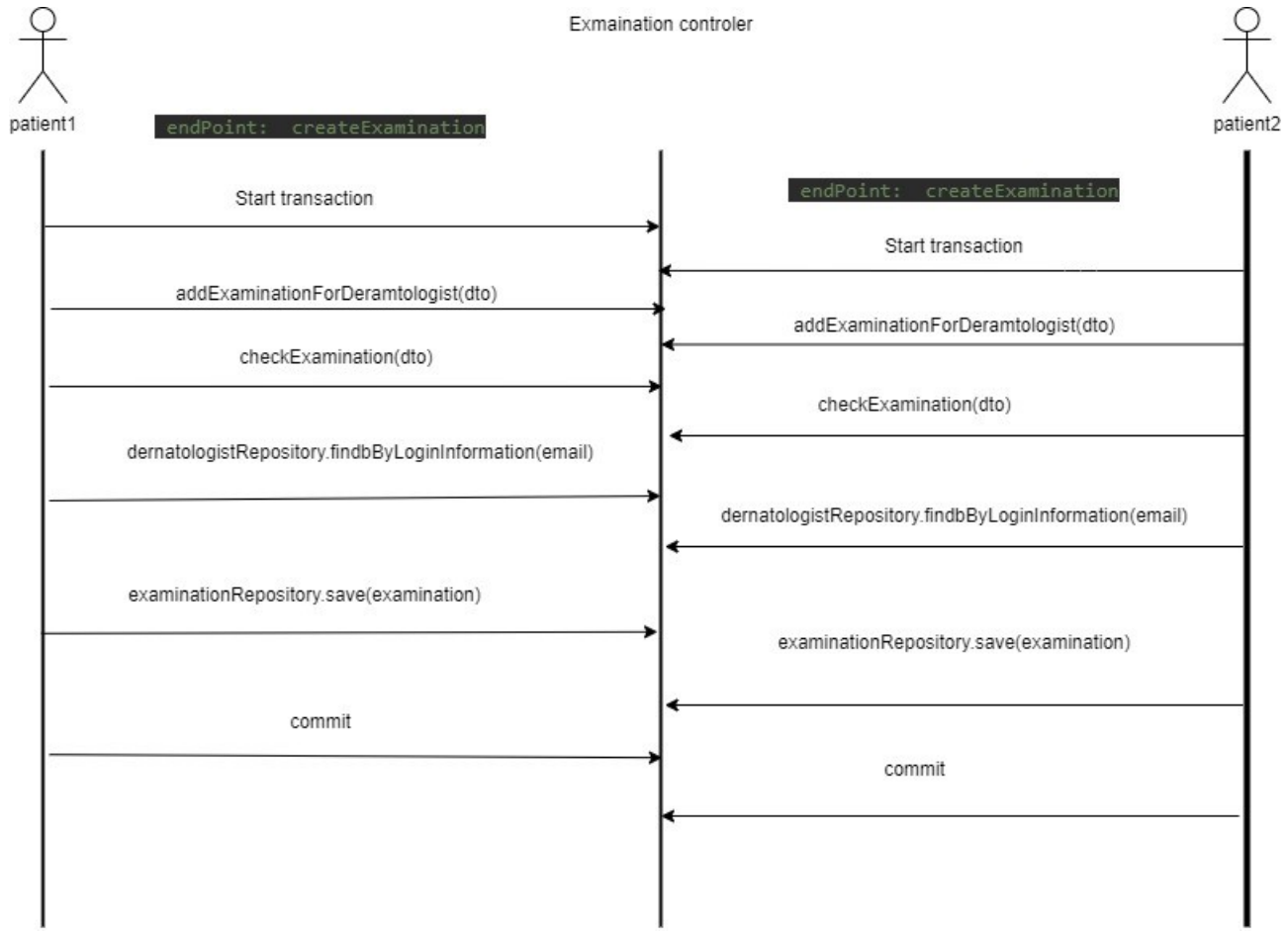
```
@PreAuthorize("hasAnyRole('ROLE_PATIENT')")
@PostMapping("/newExamination/")
@Transactional(readOnly = false)
public ResponseEntity<Examination> scheduleExamination(@RequestBody DermatologistExamScheduleDTO dto)  {
```

1. Jedan dermatolog ne može istovremeno da bude prisutan na više različitih pregleda.

Opis situacije: Administrator apoteke kreira termina kod dermatologa, pri kreiranju mu definise: datum, vreme i cenu. Može da se desi da dva administratora istovremeno pokušaju da kreiraju termine jednom dermatologu i tada dolazi do konfliktne situacije

Opis problem:

Kada dva administratora apoteke započnu kreiranje pregleda kod dermatologa, i započnu unos datuma, vremena, trajanje pregleda, dermatologa, cenu. Moguće je da se istom dermatologu definišu dva pregleda u istom terminu istog dana. Programski se rade provere za već definisane preglede i vreme kada se održavaju, da li taj dan radi dermatolog i da li je na odmoru . U slučaju da kreiranje dešava isto vremeno od strane dva administratora apoteke provere da li je termin već definisan nam ne može sprečiti konflikt i može doći do kreiranja pregleda u istom terminu .




## Resešenje problema:

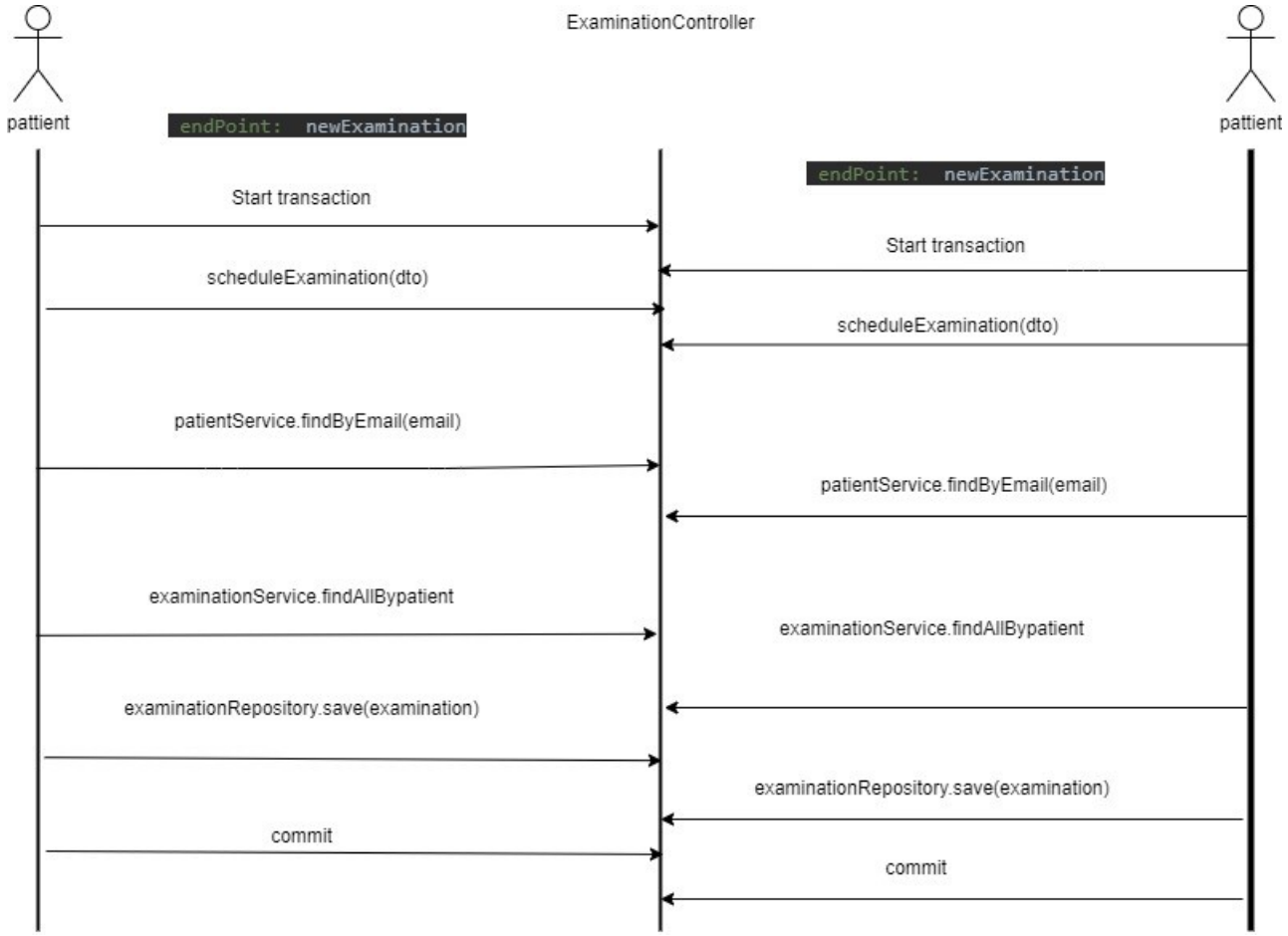
Strategija primenjena na resavanje ove konfliktne situacije je pesimiticko zakljucavanje klase Dermatologist. Dermagologa koji se dobavlja u metodi zakazivanja saveotavanja. U DermatologistRepostitori metoda findByLoginInfoEmail ide anotacija @Lock(LockModeType.*PESSIMISTIC_READ*) efekat koji dobijamo je da prvi koji pristupi resursu može da ga menja ostali mogu samo da ga čitaju. Kada prvi administrator apoteke dobavi resurs samo će on moći da ga menja ostali će moći samo da ga čitaju. Na taj način je obezbeđeno da ne dođe do konfliktne situacije u kojoj više administratora istovremeneo dodaju isti termin kod jendog dermatologa. Dodata je anotacija @QueryHints({@QueryHint(name="javax.persistence.lock.timeout", value ="1000")}) koja omogućava lokovanje resusra od 1 sekunde za koje vreme samo jedan administrator može da ga menja.

2. Jedan farmaceut ne može istovremeno da bude prisutan na više različitih savetovanja.

Opis situacjie: Savetovanja kod farmaceuta može da zakažu pacijenti. Prilikom zakazivanja prelgeda kod farmaceuta pacijent bira datum prikazujumu se apoteke u kojima ima slobodnih temrina i nakon odabira apoteke dobija prikaz farmaceuta. Nakon toga on bira kod kog farmaceuta će zakazazi pregled. Konflikta situacija se dešava kada to urade dva pacijenta istovremeno

Opis problem:

Kada dva pacijenta istovremenmo započnu zakazivanje prelgeda kod farmaceuta može doći do konfliktne situacije. Prilikom zakazivanja pregleda kada se unosi datum i vreme prikazuju mu se sve apoteke koje imaju slobodnog farmaceuta u tom periodu. Kada izaberu apoteku dobijaju prikaz farmaceuta. Ako oba pacijenta odaberu istog farmaceuta dolazi do situacije gde će jedan farmaceut imati dva pregleda u isto vreme kod dva različita pacijenta. Programski je zaštićeno da se prikažu farmaceutu koji su slobodni u tom terminu tj nemaju definisan pregled. Pošto se sam pregled kreira u bazi prilikom potvrde rezervacije dolazi do konflikta gde dva pacijenta kreiraju pregled istovremeno kod istog farmaceuta.



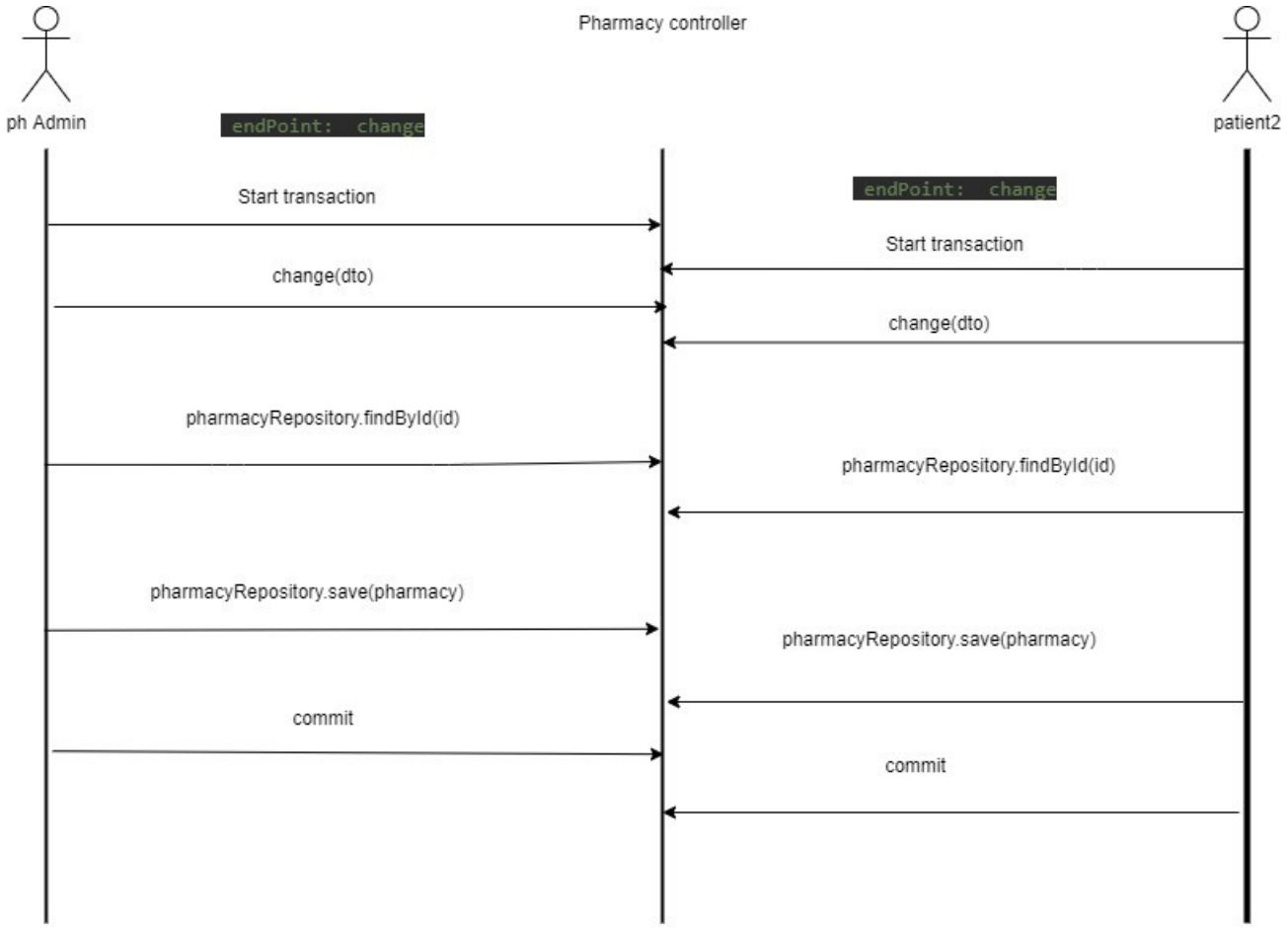
Resešenje problema:
Strategija primenjena na resavanje ove konfliktne situacije je pesimiticko zakljucavanje klase MedicalStuf. Farmaceut koji se dobavlja u metodi zakazivanja saveotavanja. U MedicalStufRepostitori metoda findByUserId(UUID userId);

ide anotacija @Lock(LockModeType.*PESSIMISTIC_READ*) efekat koji dobijamo je da prvi koji pristupi resursu može da ga menja ostali mogu samo da ga čitaju. Kada prvi pacijent dobavi resurs samo će on moći da ga menja ostali će moći samo da ga čitaju. Na taj način je obezbeđeno da ne dođe do konfliktne situacije u kojoj više pacijenta istovremeneo zakazuje isti termin kod jendog farmaceuta. Dodata je anotacija @QueryHints({@QueryHint(name="javax.persistence.lock.timeout", value ="2000")}) koja omogućava lokovanje resusra od 3 sekunde za koje vreme samo jedan administrator može da ga menja.

3. Izmenda podatka apoteke od strane administratora apoteke

Opis situacjie: Kako jedna apoteka može da ima više administratora apoteke. Može da se desi da dva administratora žele da menjau podatke istovremeno. Tada dolazi do konfliktne situacije
Opis problem:

Kada dva administratora apoteke započnu izmenu podataka apoteke, dolazi do problema gde jedan administrator izmeni neki podatak drugi administrator izmeni takođe neki podataka. Prilikom slanja na server izmena prvog administratora će se izvršiti i on dobija potvrdu o uspešnosti, a zahtev drugog administratora pristiže i on pregazi podatke od prvog administratora, takođe i drugi administrator dobija potrvdu o uspešnoj izvršenoj izmeni. Ovde se dešava gubitak podataka.




Problem resavamo: pomoću optimistic locking-a. U apoteku dodajemo novo polje, private Long version i dodeljujemo mu anotaciju @Version.

Pomoću Version @Version anotacvije postiže to da pri izmeni podataka apoteke proveri verzija kako bi se desilo da su isto vremeno više korisnika pokušali da menjaju isti podataka. Ako je verzija ista kao kada se učitao podatak, onda se izvrši čuvanje i polje version se inkrementira. Ovo nam je omogućilo da ako administrator apoteke1 pošalje svoje izmene i administrator apoteke2 pošalje svoje izmene, sačuvane će biti od onog administratora koje prve pristignu tj čija verzija se podudara sa trenutnom u bazi. Iz tog razloga dolazi do ObjectOptimisticLockingFailureException-a kada se u okviru PharmacyService klase, u metodi change pozove pharmacyRepository.save()