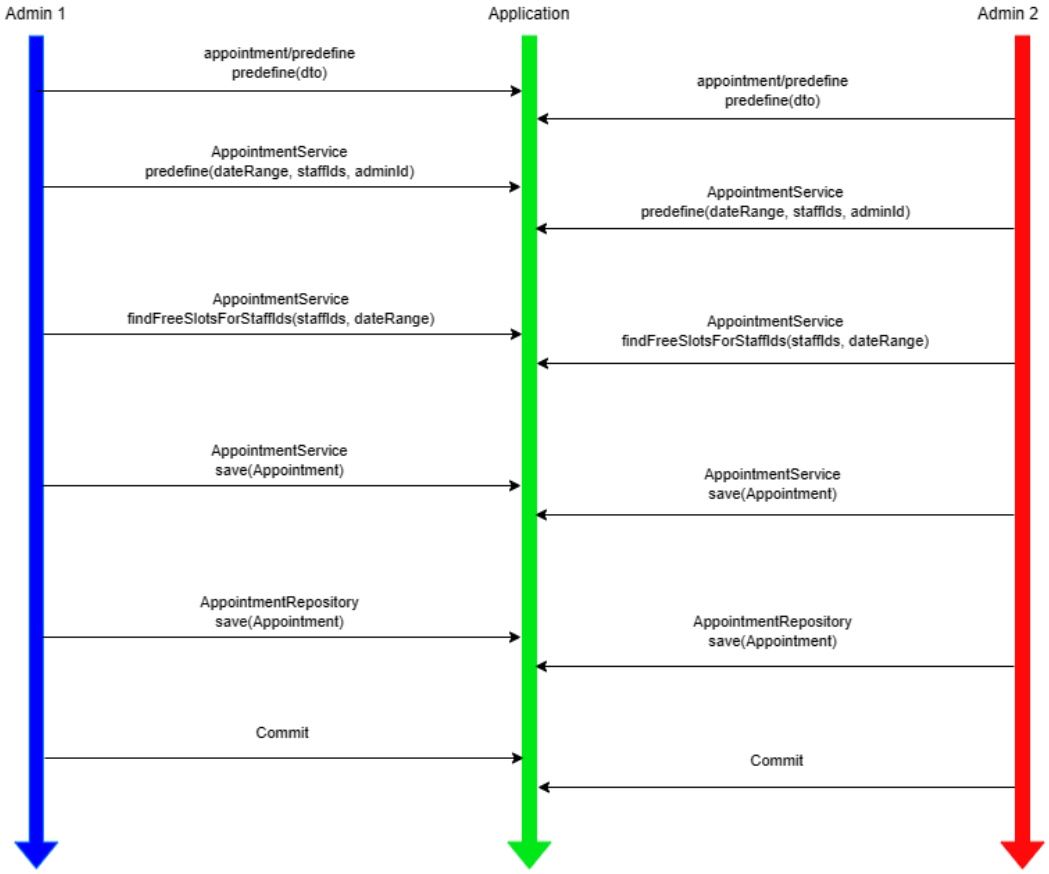
# Konfliktne situacije - student 2 (Strahinja Eraković RA152/2019)

## 1) Više administratora ne mogu unapred definisati termine u isto ili preklapajuće vreme

*Opis konfliktne situacije:* Administrator/zaposleni u centru (u daljem teksu administrator ili osoblje) ima mogućnost da predefiniše termin u određeno vreme i određenog trajanja. Problem nastaje kada 2 administratora pokušaju da zakažu termine koji počinje u isto vreme ili se ova dva termina preklapaju. Ako se opisana situacija dogodi u bazi podataka će biti kreirana dva pregleda koja se preklapaju sto je naravno neželjeno.

*Dijagram zahteva i objašnjenje:*




Administrator 1 započinje transakciju tako što gađa *endpoint* za predefinisanje termina. Administrator 2 odmah posle njega poziva isti *endpoint*. Oboje šalju isti *dto* tj pokušavaju da zakažu pregled u isto vreme. Dalje se poziva medota za predefinisanje iz servisa koja će pozvati medotu *findFreeSlotsForStaffIds*. Ova metoda proverava da li je moguće zakazati termin u prosleđeno vreme. Pošto obe metode vide isto stanje u bazi oba termina bivaju označeni kao validni. Poziva se medtoda *save* koja upisuje oba terminau bazu. Ovo narušava konzistentnost podataka u bazi.

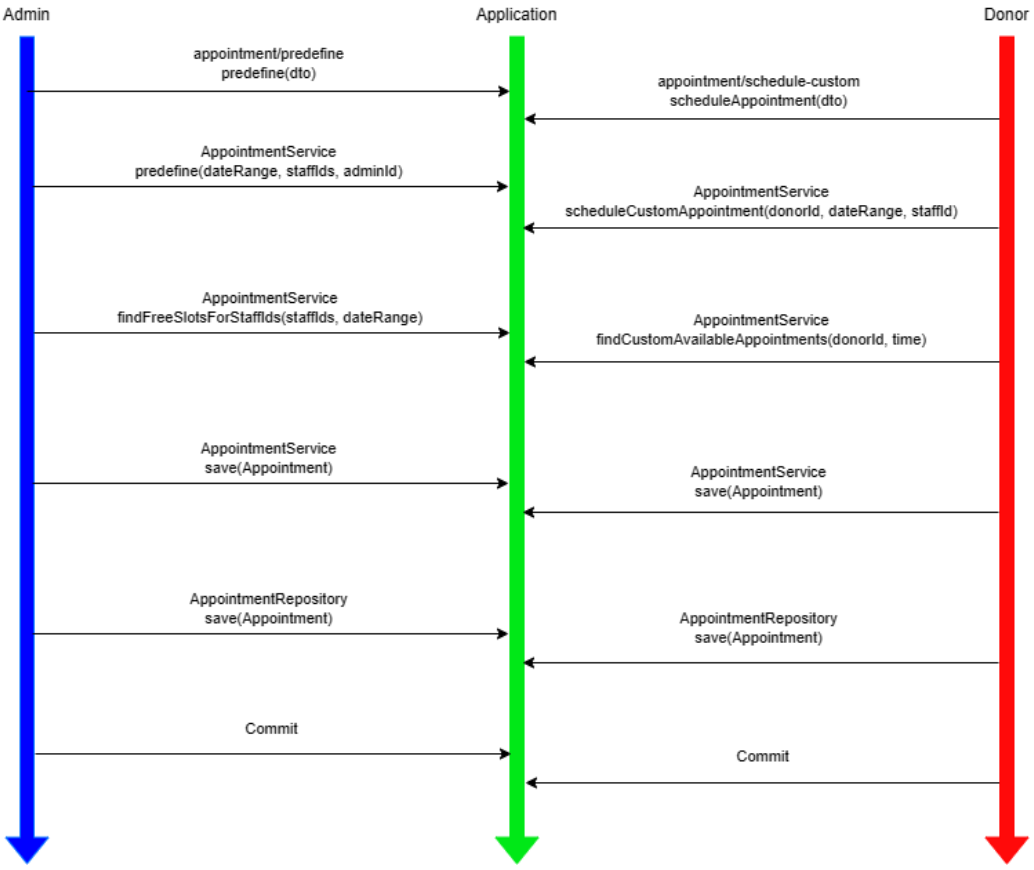
*Opis rešenja:* Kao rešenje izabrano je podizanje nivoa izolacije baze na *SERIALIZABLE.* Ovaj nivo izolacije omogućava transakcijama da se izvršavaju “serijski” tj. Jedna posla druge. Transakcija će se izvršiti tek kada transakcija pre nje komituje svoje izmene. Na ovaj način će kreiranje prvog termina biti komitovano a pokušaj kreiranja drugog termina će biti neuspešan jer će pristup bazi biti

zabranjen. Ovo rešenje je odabrano jer je kreiranje termina *INSERT* novog reda u tabeli. Ne možemo koristiti optimističko zaključavanje jer ne modifikujemo već postojeći red nego ubacujemo novi. Konkretno u kodu je ovo urađeno korišćenjem anotacije *@Transactional* i davanjem dodatnog parametra *isolation = Isolation.SERIALIZABLE*. Konfliktna situacija je testirana pomoću *testova* i *thread-ova*. Pokrene se test koji zatim pokrene dve niti koje pokušavaju da predefinišu termine u isto vreme.

### 2) Administrator ne može predefinisati termin u isto ili preklapajuće vreme za koje korisnik kreira rezervaciju termina.

*Opis konfliktne situacije:* Korisnik (odnosno donor) ima mogućnost da rezerviše termin u određeno vreme. U prethodnoj tački je objašnjeno da administrator može da predefiniše termin. Problem se javlja kada korisnik pokuša da rezerviše termin u isto vreme kada i administrator pokušava da predefiniše termin. Posledica problema je da se će se u bazi kreirati dva termina sa istim ili preklapajućim vremenom.

*Dijagram zahteva i objašnjenje:*




Administrator gađa *endpoint* za predefinisanje pregeda, a korisnik odnosno donor gađa *endpoint* za rezervisanje termina. Kontroler za predefinisanje termina kao i u prethodnoj tački poziva iste metode servisa. Kontroler za rezervisanje termina poziva medotu *scheduleCustomAppointment* iz servisa za termine kojoj prosleđuje vreme za koje želi da rezerviše termin (identično vreme kada i administrator

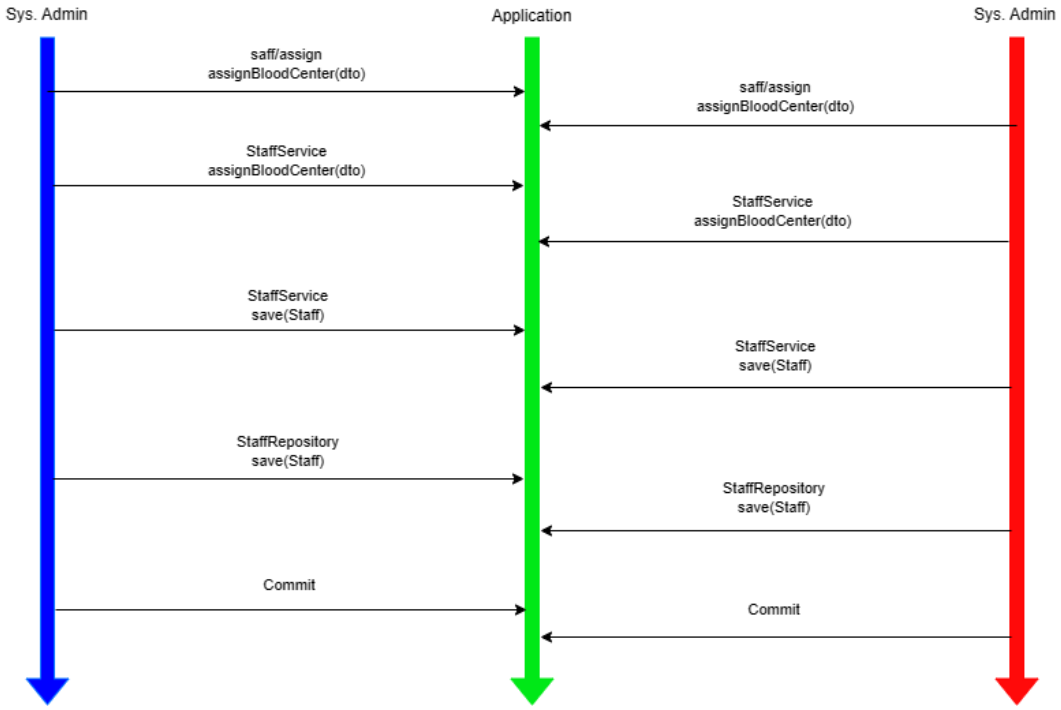
pokušava da predefiniše termin). Zatim se poziva metoda *findCustomAvailableAppointments* koja proverava da li je moguće rezervisati željeni termin. U nastavku se poziva *save* metoda koja će komitovati izmene. Kada se transakcije završe u bazi bivaju kreirana dva termina u identičnom vremenu.

*Opis rešenja:* Slično kao u prethodnoj tački kao rešenje je izabrano podizanje nivoa izolacije na *SERIALIZABLE.* Ova konfliktna situacija je u konkretnoj implementaciji ovih funkcionalnosti jako slična kao prošla pa je samim tim i rešenje identično. Za testiranje se takoće koriste dve niti. Prva nit pokušava da predefiniše termin a druga pokušava da rezerviše termin. U zavisnosti od toga koja nit prva započne transakciju zavisi da li će uspešno biti predefinisan termini ili će uspešno biti rezervisan termin.

## 3) Sistemski administrator ne mogu zaposliti osbolje u dve različite banke krvi

*Opis konfliktne situacije:* Sistemski administrator centra ima mogućnost da administratoru tj. osoblju dodeli banku krvi u kojoj će oni biti zaposleni. Problem nastaje ako dva sistemska administratora pokušaju u isto vreme da dodele različite banke krvi istom administratoru. Posledica konflikta je da će poslednja izmena biti upisana u bazu, tj onaj sistemski administrator koji je poslednji dodelio banku je “pobedio” u trci.

*Dijagram zahteva I objasnjenje:*




Oba sistemska administratora gađaju *endpoint* za dodelu banke krvi osoblju. Kontroler poziva servis za dodelu banke krvi. Na kraju se poziva metoda *save* koja komituje promene u bazi. Komit koji se poslednji desi, u ovom slučaju crveni sistemski admin, će “overwritovati” prošlo stanje u bazi. Na primer ako plavi administrator pokuša da mu dodeli “Centar 1” a crveni administrator “Centar 2” na kraju će mu biti dodeljen “Centar 2”.

ISA 2022/23 - Strahinja Erakovic RA152/2019

*Opis rešenja:* Kao rešenje u ovom slučaju je izabrano optimističko zaključvanje. Ovo rešenje možemo iskoristiti u ovom slučaju jer modifikujemo red u tabeli *staff*. Uvodimo novu kolonu *version* koja će služiti da vodimo evidenciju o verziji reda u tabeli. Ovo rešava problem na sledeći način: Kada plavi administrator pokuša da izmeni osoblje on iz baze dobije na izmenu osoblje sa verzijom 1. Kada se njegova promena komituje verzija se inkrementuje na 2. Crveni administrator je u međuvremenu takođe na izmenu dobio osoblje sa verzijom 1. Kada on pokuša da komituje svoje promene porede se verzije u bazi i verzija koja pokušava da bude komitovana. Pošto se verzije razlikuju komit se odbija. Dodavanje verzije je u kodu odrađeno tako što je u *modelu* dodato polje *int version* koje je anotirano sa *@Version*. Testiranje rešenja je takođe urađeno preko 2 niti koje u isto vreme pokušavaju jedon administratoru da dodele dva različita centra.