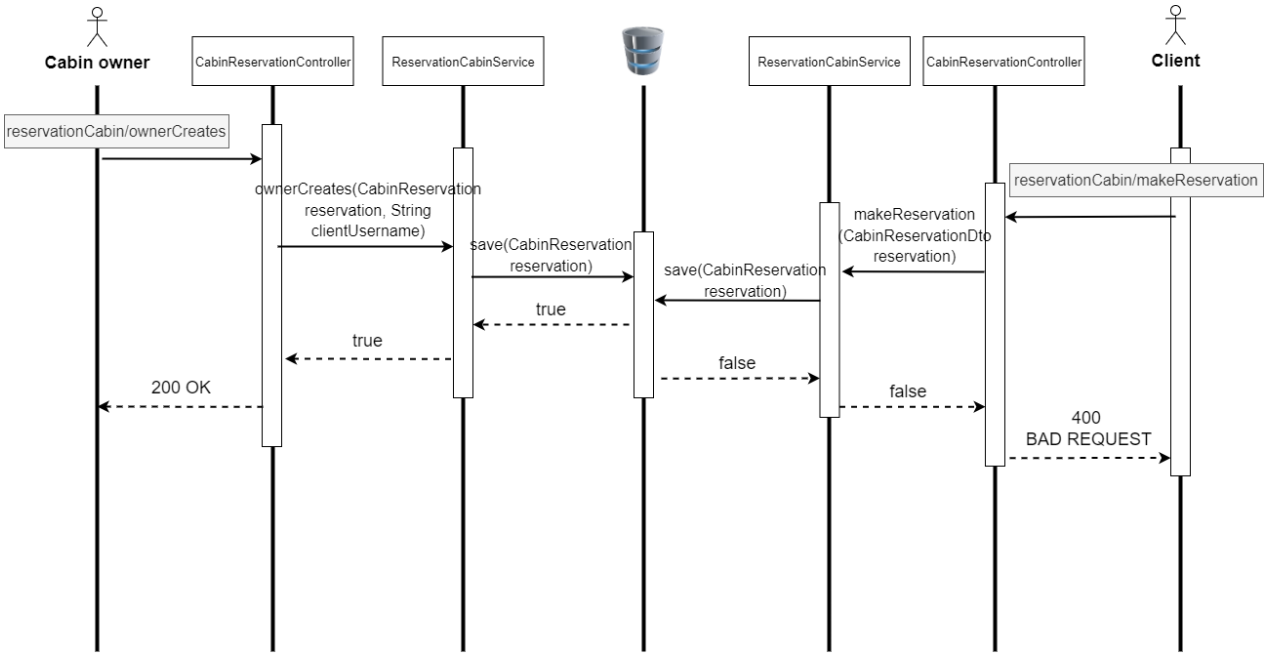
KONKURENTNI PRISTUP BAZI – Student 2

1. Kreiranje rezervacije

Mogućnost kreiranja rezervacije u sistemu imaju klijenti ali i vlasnici vikendice, broda i instruktori pecanja. Ukoliko vlasnik nekog od entiteta u isto vreme kao i klijent pokuša da napravi rezervaciju istog entiteta za isti ili preklapajući period, dolazi do konfliktne situacije. Kako bismo izbegli kreiranje dve rezervacije za isti entitet u istom ili preklapajućem terminu, potrebno je rešiti ovu konfliktnu situaciju. Na slici 1 prikazana je situacija kada vlasnik vikendice i klijent u isto vreme pokušavaju da rezervišu vikendicu u istom terminu, pri čemu je vlasnik pokrenuo transakciju nekoliko sekundi ranije. (*Napomena: na dijagramu su izostavljene metode logike servisa i validacije kako bi se pokazala suština transakcija*)



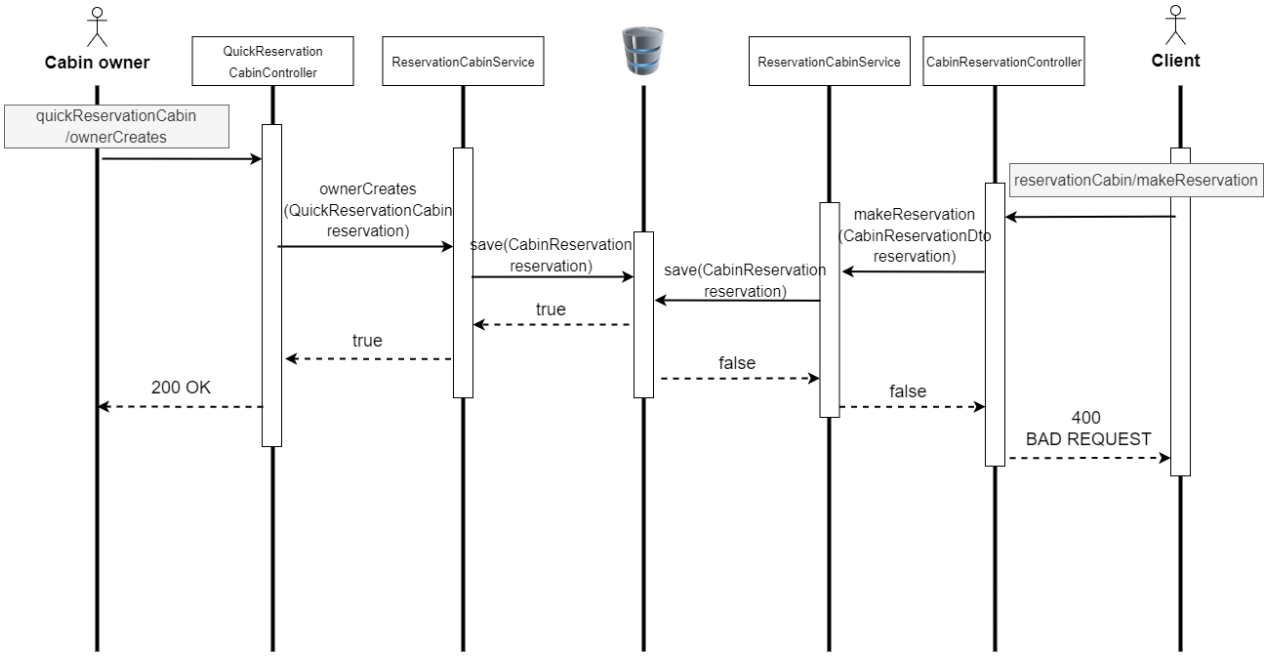
Slika 1 – Kreiranje rezervacije

Problem je rešen upotrebom optimističkog zaključavanja. U klasu Cabin dodato je polje version sa anotacijom @Version koja nam omogućava da izvršimo verzionisanje torke u bazi. Za kreiranje rezervacije sa strane vlasnika vikendice korišćena je metoda servisa CabinReservationService ownerCreates, a sa strane klijenta metoda makeReservation. Obe metode označene su kao transakcione anotacijom @Transactional. Nivo izolacije podešen je na SERIALIZABLE (rešena su četiri glavna problema konkurentnog pristupa bazi), atribut readonly postavljen na false jer vršimo izmenu u bazi, a atribut propagation na REQUIERS_NEW. Ovim smo obezbedili da metoda uvek pokreće novu transakciju, a ako postoji tekuća

transakcija ona se suspenduje. U slučaju da vlasnik vikendice i klijent u isto vreme pokrenu transakciju vezanu za rezervaciju istog entiteta, ali te rezervacije nisu u istom ili preklapajućem terminu, prvo će se izvršiti rezervacija koja je prva stigla, a nakon što ona bude sačuvana sačuvaće se druga rezervacija.

2. Kreiranje brze rezervacije od strane vlasnika i kreiranje obične rezervacije od strane klijenta

Vlasnici vikendica, brodova i instruktori pecanja imaju mogućnost da za svoje entitete kreiraju brze rezervacije/akcije. Ukoliko vlasnik nekog od entiteta pokuša da kreira brzu rezervaciju u isto vreme kada klijent vrši običnu rezrvaciju tog entiteta za isti ili preklapajući termin, dolazi do konfliktne situacije. Kako bismo izbegli kreiranje brze akcije i rezervacije za isti entitet u istom ili preklapajućem terminu, potrebno je rešiti ovu konfliktnu situaciju. Na slici 2 prikazana je situacija kada vlasnik vikendice kreira brzu akciju, a klijent u isto vreme pokuša da rezerviše vikendicu u istom terminu, pri čemu je vlasnik pokrenuo transakciju nekoliko sekundi ranije. (*Napomena: na dijagramu su izostavljene metode logike servisa i validacije kako bi se pokazala suština transakcija*)

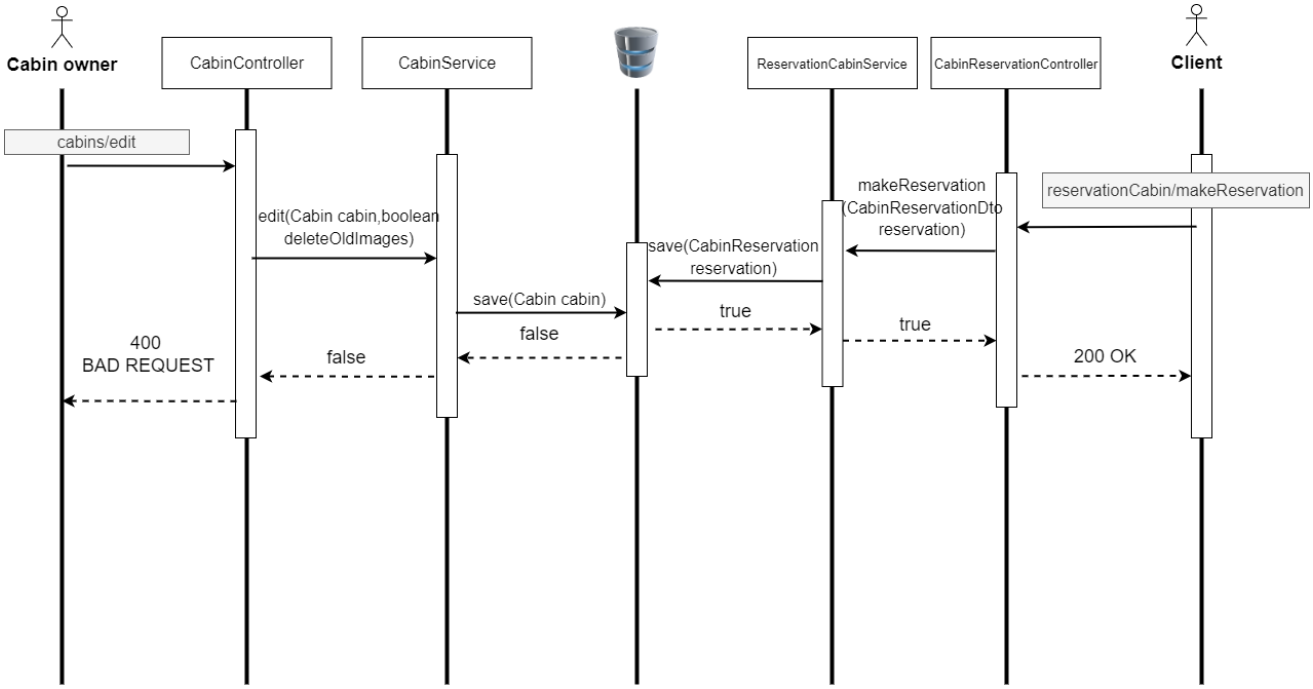


Slika 2 – Kreiranje brze rezervacije od strane vlasnika vikendice i obične rezervacije od strane klijenta

Problem je rešen upotrebom optimističkog zaključavanja. U klasu Cabin dodato je polje version sa anotacijom @Version koja nam omogućava da izvršimo verzionisanje torke u bazi. Za kreiranje rezervacije sa strane vlasnika vikendice korišćena je metoda servisa QuickReservationCabinService ownerCreates, a sa strane klijenta metoda makeReservation servisa CabinReservationService. Obe metode označene su kao transakcione anotacijom @Transactional. Nivo izolacije podešen je na SERIALIZABLE (rešena su četiri glavna problema konkurentnog pristupa bazi), atribut readonly postavljen na false jer vršimo izmenu u bazi, a atribut propagation na REQUIERS_NEW. Ovim smo obezbedili da metoda uvek pokreće novu transakciju, a ako postoji tekuća transakcija ona se suspenduje. U slučaju da vlasnik vikendice i klijent u isto vreme pokrenu transakciju vezanu za rezervaciju odnosno brzu rezervaciju istog entiteta, ali te rezervacije nisu u istom ili preklapajućem terminu, prvo će se izvršiti rezervacija koja je prva stigla, a nakon što ona bude sačuvana sačuvaće se druga rezervacija.

3. Izmena entiteta i kreiranje rezervacije

Vlasnici vikendica, brodova ili instruktori pecanja imaju mogućnost izmene entiteta ukoliko taj entitet nema rezervacije u budućnosti. Ukoliko vlasnik vikendice pokuša da izmeni entitet u trenutku kada neki klijent pokuša da kreira rezervaciju, može se desiti da se rezervacija kreira po novim uslovima (nova cena, promenjena pravila, uslovi korišćenja ili dodatni servisi) jer će vikendica možda biti izmenjena pre nego što se kreira rezervacija. Kako bismo ovo izbegli, potrebno je rešiti konfliktnu situaciju. Na slici 3 prikazana je situacija kada vlasnik vikendice pokušava da izmeni vikendicu, a klijent pokušava da kreira rezervaciju. (*Napomena: na dijagramu su izostavljene metode logike servisa i validacije kako bi se pokazala suština transakcija*)



Slika 3 – Izmena vikendice od strane vlasnika i kreiranje rezervacije od strane klijenta

Problem je rešen upotrebom optimističkog zaključavanja. U klasu Cabin dodato je polje version sa anotacijom @Version koja nam omogućava da izvršimo verzionisanje torke u bazi. Za izmenu vikendice od strane vlasnika korišćena je metoda edit servisa CabinService, a za kreiranje rezervacije od strane klijenta metoda makeReservation servisa CabinReservationService. Obe metode označene su kao transakcione anotacijom @Transactional. Nivo izolacije podešen je na SERIALIZABLE (rešena su četiri glavna problema konkurentnog pristupa bazi), atribut readonly postavljen na false jer vršimo izmenu u bazi, a atribut propagation na REQUIERS_NEW. Ovim smo obezbedili da metoda uvek pokreće novu transakciju, a ako postoji tekuća transakcija ona se suspenduje. Kako je u ovom primeru klijent kreirao rezervaciju nekoliko sekundi pre nego što je vlasnik pokušao da izmeni vikendicu, sistem obaveštava vlasnika da za tu vikendicu postoje rezervacije u budućnosti pa je ne može izmeniti. U suprotnom slučaju, sistem će obavestiti klijenta da je vikendica izmenjena i da može ponovo da pokuša da kreira rezervaciju.

Napomena: Navedeni problemi rešeni su na isti način u slučaju brodova i avantura, ali je zbog jednostavnosti detaljno opisana vikednica. Metode, servisi i endpointi vezani za ove entitete nazvani su isto, samo imaju drugačije prefikse (Boat/ Adventure).