Milos Markovic RA132/2018

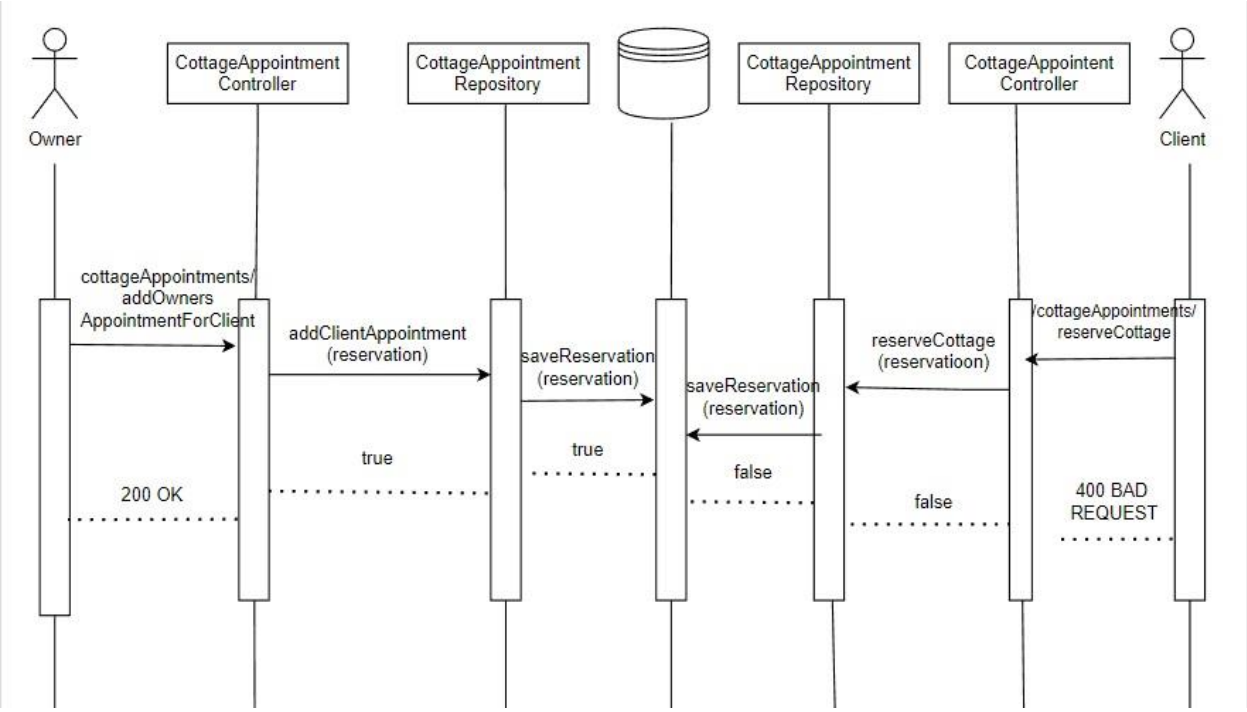
4.4 Konkurentni pristup bazi-Student 2

Resene situacije:

Sve situacije se odnose i na vlasnika broda i insturktora pecanja, ali primera radi uzimamo vlasnika vikendice.

1. Vlasnik vikendice pravi rezervaciju sa klientom u isto vreme kad i klijent pravi rezervaciju za isti entitet

Opis situacije: Kao sto klijenti mogu da zakazu rezervaciju, vlasnik vikendice moze da zakaze novu rezervaciju u dogovoru sa klijentom. Problem moze nastati ako se u isto vreme pravi rezervacija koja se odnosi za istu vikendicu u istom ili preklapajucem periodu. (Napomena: na dijagramu su izostavljene metode logike servisa i validacije kako bi se pokazala suština transakcija)

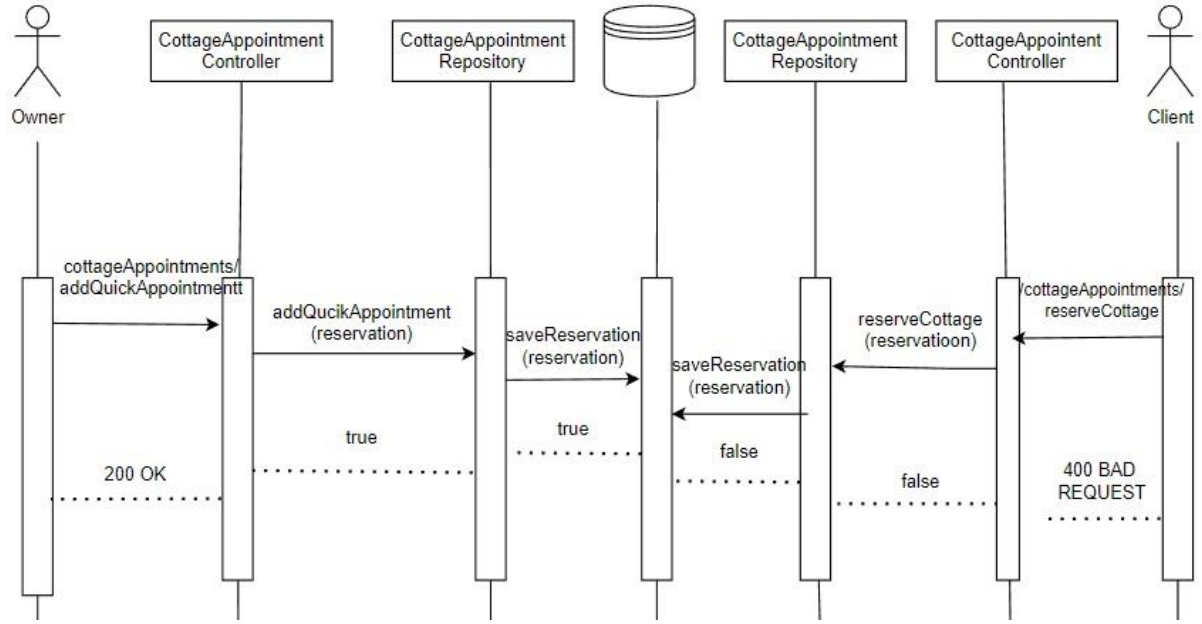


Resenje:

Postupak se nalazi u klasi CottageAppointmentController, pozivaju se metode addClientAppointment i reserveCottage, kao i u CottageAppointmentRepository klasi gde se poziva saveReservation metoda. Kako bi se izbegao konflikt koristimo optimistickim zakljucavanjem, dodali smo anotaciju Version u CottageAppointment , takodje obe metode iz kontrolera su oznacene kao Transactional i REQUIRES_NEW propagacije. Takodje je dodat exception koji se baca klijentu ili vlasniku prilikom rezervacije.

2. Vlasnik vikendice pravi akciju za klijente u isto vreme kad I klijent pravi rezervaciju za dati entitet

Opis situacije: Vlasnik moze da pravi akcije koje su brze rezervacije za koje se prijavljuje klijent. Ako se ove akcije naprave u istom ili preklapajucem periodu sa rezervacijom koju pravi klijent u isto vreme. (Napomena: na dijagramu su izostavljene metode logike servisa i validacije kako bi se pokazala suština transakcija)

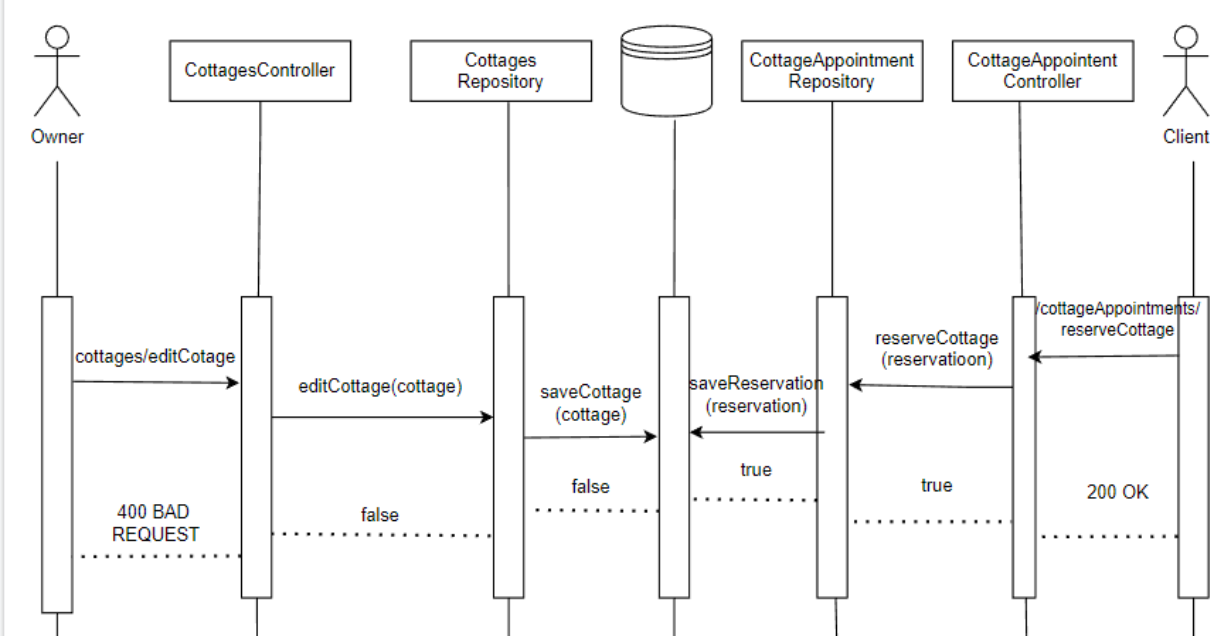


Resenje:

Postupak se nalazi u klasi CottageAppointmentController, pozivaju se metode addQuickAppointment i reserveCottage, kao i u CottageAppointmentRepository klasi gde se poziva saveReservation metoda. Kako bi se izbegao konflikt koristimo optimistickim zakljucavanjem, dodali smo anotaciju Version u CottageAppointment , takodje obe metode iz kontrolera su oznacene kao Transactional i REQUIRES_NEW propagacije. Takodje je dodat exception koji se baca klijentu ili vlasniku prilikom rezervacije.

3. Izmena postojeceg entiteta u isto vreme kad i klijent pravi rezervaciju

Opis situacije: Vlasnici vikendice ima mogućnost izmene entiteta ukoliko taj entitet nema rezervacije u tom trenutku. 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



Resenje:

Postupak se nalazi u klasama CottageAppointmentController(gde se poziva reserveCottage), CottagesController (gde se poziva editCottage) CottageAppointmentRepository (gde se poziva saveReservation) metoda I CottagesRepository(gde se poziva saveCottage) Kako bi se izbegao konflikt koristimo optimistickim zakljucavanjem, dodali smo anotaciju Version u Cottage i CottageAppointment , takodje obe metode iz kontrolera su oznacene kao Transactional i REQUIRES_NEW propagacije. Takodje je dodat exception koji se baca klijentu ili vlasniku prilikom rezervacije I kreiranja entiteta.