Решавање конфликтних ситуација и конкурентни приступ ресурсима у бази

Студент: Матија Матовић

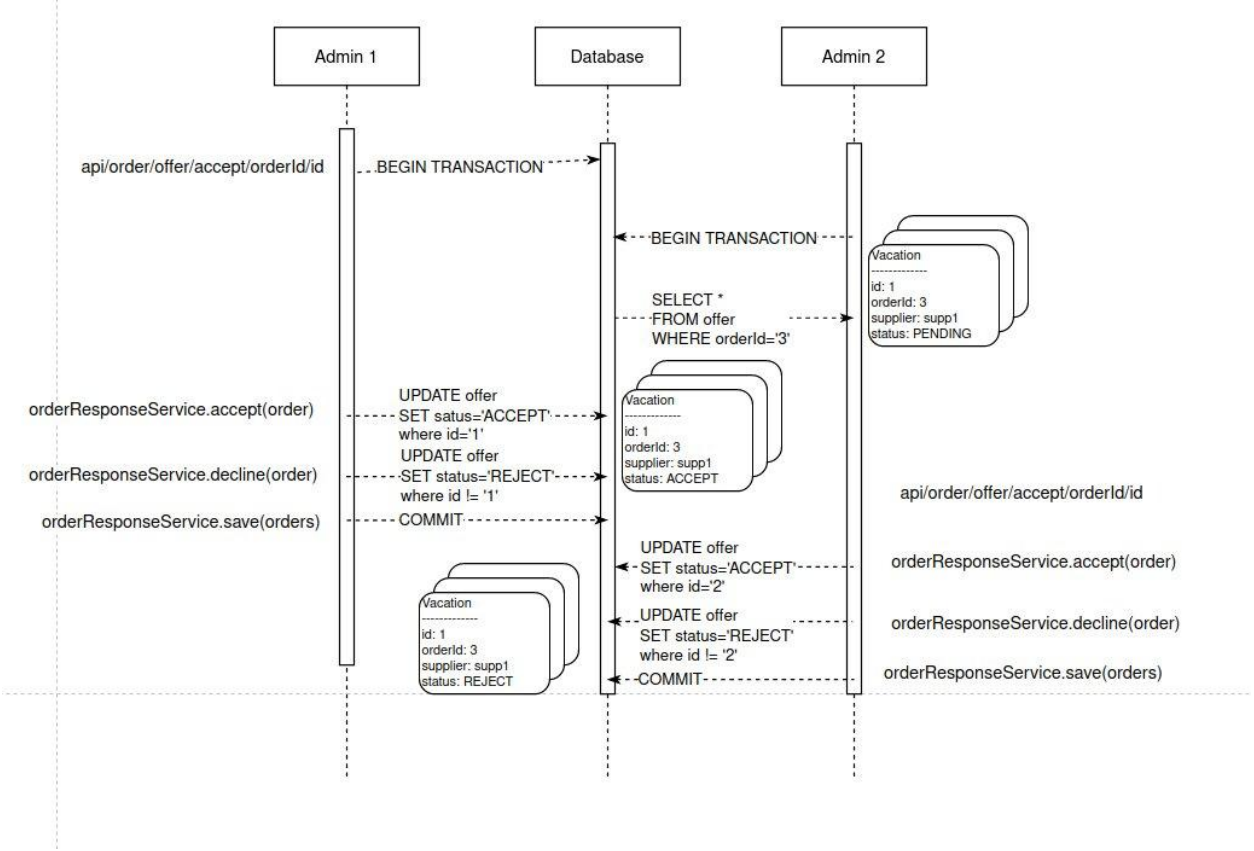
SW-6-2018

Предмет: Интернет софтверске архитектуре
Година: 2020/2021
Тим 19

Прихватање понуде за наруџбину

Могуће је да се деси да више администратора апотеке одговоре на исту наруџбину у исто време. Уколико се то деси могуће је да једно прихватање погази друго, самим тим, да се подаци неправилно сачувају. Такође, том приликом би више различитих добављача било обавештено мејлом о прихваћеној понуди, иако само једна има статус “Прихваћено”.

Овај проблем је описан следећим дијаграмом:



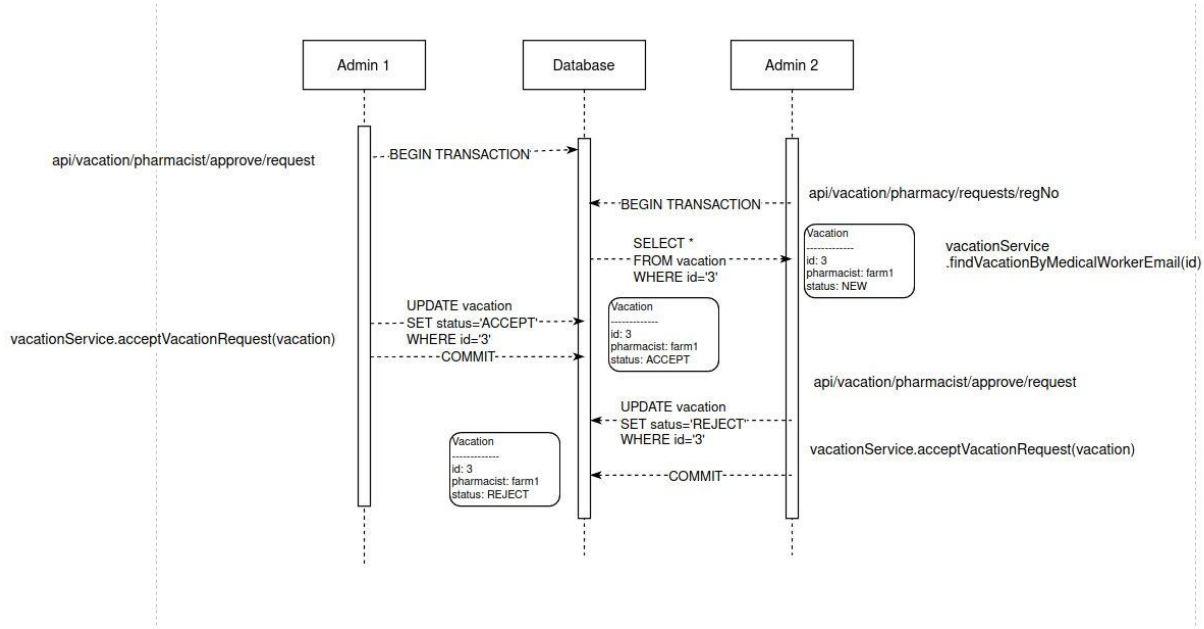
Може се приметити да је једна понуда која је првобитно одбијена, прихваћена, а она која је првобитно прихваћена је сада одбијена. Добављачу је стигло 2 мејла, један који тврди да је наруџбина прихваћена, а други да је одбијена. Такође, при прихватању понуде ажурира се број лекова на стању апотеке, те би се у овом случају број лекова више пута повећао, јер су понуде на исту наруџбину прихваћене више пута. Тада број лекова у бази не би био формално конзистентан.

Овај проблем се може решити оптимистичким закључавањем. Вероватноћа да се овако нешто деси је мала, те се не мора база закључавати сваки пут. Довољно је да класе које представљају понуду за наруџбину (OrderResponse) и количину лекова у апотеци (MedicineQuantity) прошире са атрибутом Version и анотирају са @Version. Такође део кода који прихвата или одбија понуде потребно је анотирати са @Transactional. Тако, уколико дође до промене количине лека, или стања понуде, окружење ће бацити optimisticLockException.

Прихватање/Одбијање захтева за годишњи одмор

Могуће је да се деси да два администратора апотеке одговоре на исти захтев за годишњим одмором у исто време. Ово је врло мало вероватно да се деси, међутим, како се здравственом раднику шаље мејл да му је захтев за одмором потврђен/одбијен, ова операција траје поприлично дуго, што проширује временски прозор у коме би одговарање на исти захтев изазвало конфликт.

Овај проблем је описан следећим дијаграмом

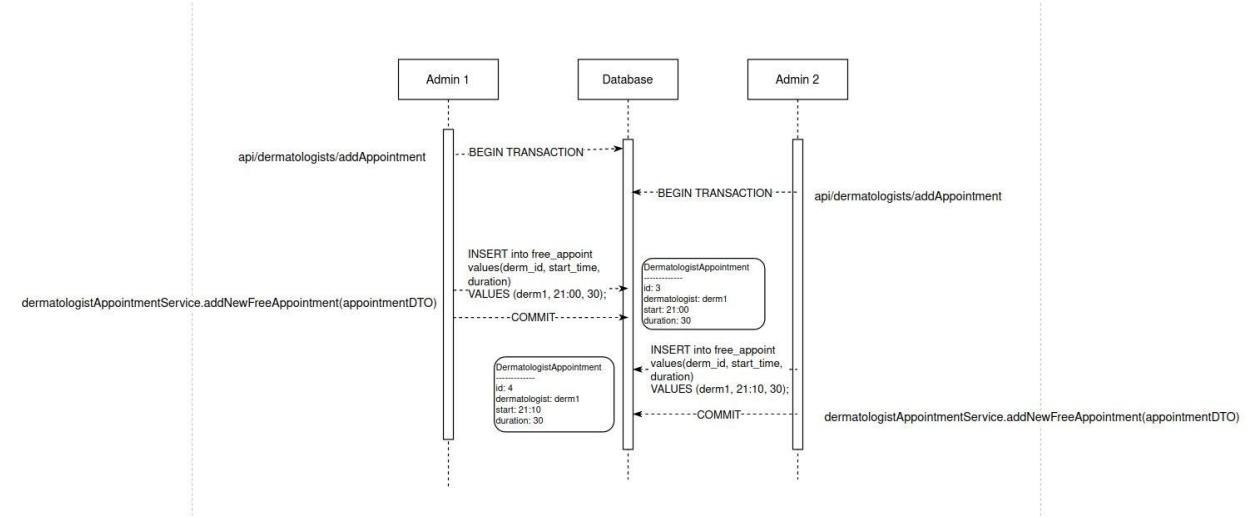


Може се приметити да између BEGIN TRANSACTION и UPDATE постоји велика временска разлика. То је зато што се у том периоду шаље мејл кориснику о статусу његовог захтева. Ту настаје простор за грешке, где други админ чита почетни захтев, мења га, и након COMMIT-а првог админа, COMMIT-ује своју измену, поништавајући прву. Здравствени радник прима мејл од оба админа, што је веома непожељно.
Како би се овај проблем решио потребно је онемогућити админу да сачува одговор на захтев уколико је дошло до промене тог захтева у међувремену. То се може постићи чувањем података о верзији захтева, додавањем поља Version, анотираног са @Version, и смештањем извршног кода у функију која је анотирана са @Transactional. Сада, ако се у међувремену деси промена захтева, приликом чувања систем ће упоредити вредности Version атрибута, и ако се разликују, баciће OptimisticLockException.

Додавање слободног термина за дерматолога

Могуће је да се деси да два администратора апотеке у исто време додају слободан термин за дерматолога. Ово иначе не би представљао проблем, међутим јесте уколико се деси да се ова два термина подударају. Програм врши проверу да ли у време новог термина дерматолог ради у задатој апотеци, или већ има неки постојећи термин. Проблем настаје када се нови термин, који би иначе изазвао да додавање тренутног не прође, дода након провере, а пре чувања новог термина. У том временском прозору могуће је да дође до конфликта, где би дерматолог у исто време имао два термина

Овај проблем може се описати следећим дијаграмом:



Овде није довољно проверавати базу пре додавања новог термина, јер то захтева време у коме би се могао додати нови конфликтан термин. Такође, закључавање није довољно, јер се не ради ажурирање неког појма, већ додавање новог. Једно решење би било да се сматра да се дерматолог у процесу додавања новог термина мења. Проширићемо дерматолога са пољем version, и анотирати га са @Version. Сада, сваки пут када се додаје нови термин, повећава се вредност овог поља. Међутим, ова акција не мења директно дерматолога, те се Version не инкрементује само. Зато ћемо у UserRepository додати функцију getAndIncrement(String email), анотирану са @Lock(OPTIMISTIC_FORCE_INCREMENT). Ова функција повећава вредност version атрибута при чувању дерматолога у базу. Функцију позивамо на самом почетку трансакције, како би смо знали верзију дерматолога на самом почетку, пре свих провера доступности термина. Дерматолога чувамо у бази на крају (иако се он сам није мењао). Уколико је неки други админ позвао такође функцију getAndIncrement то значи да је он такође у том тренутку заказивао термин за дерматолога, и променио му верзију. Овде ће настати OPTIMISTIC_LOCK_EXCEPTION и окружење ће бацити грешку.
Овде постоји проблем, а то је да се грешка дешава, иако 2 нова термина нису у преклапајућим временским интервалима. Међутим, то је цена која се мора платити, јер би

супротно захтевало да се поново прођу сви термини и провери доступност, што је временски скупо.