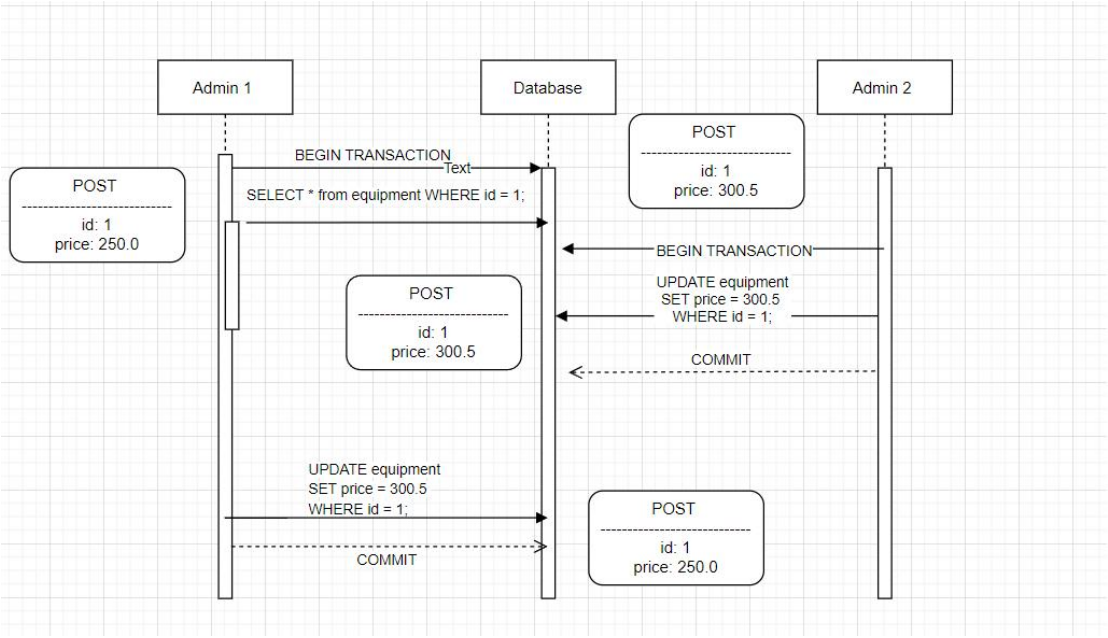
6.5. Konkurentni pristup resursima u bazi

- Konfliktna situacija: više administratora kompanije ne može u isto vreme da menja podatke o istoj opremi (npr. cenu)

- Crtež toka koji dovodi do konfliktne situacije:

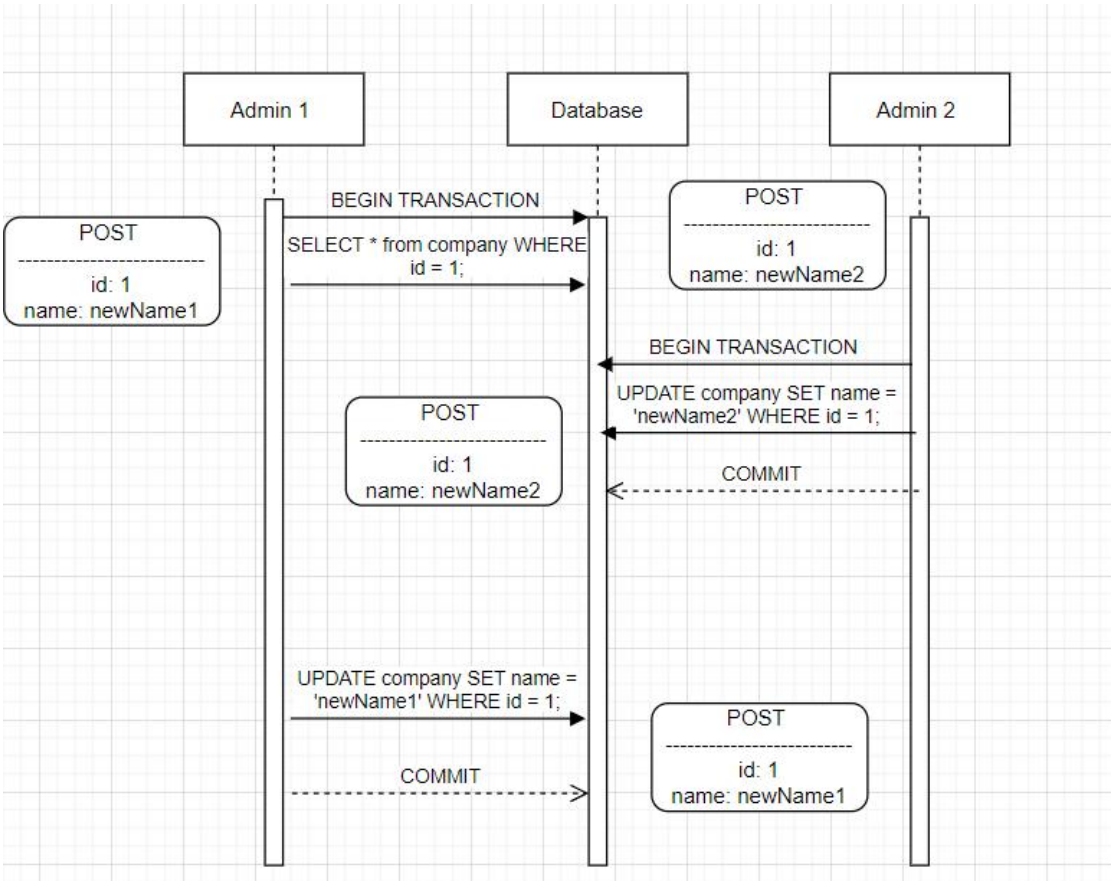


- Rešenje:
- Napisala sam test CompanyEquipmentTests.java u projektu. U ovom testu, korišćeno je optimističko zaključavanje koje se primenjuje na objektu Equipment, kako bi se rešio potencijalni konflikt između dva administratora kompanije koji pokušavaju ažurirati cenu (price) iste torke tabele Equipment u isto vreme.
- Svaka torka tabele Equipment sadrži polje version koje se koristi kog optimističkog zaključavanja za proveru da li je došlo do konflikta prilikom ažuriranja. Optimističko zaključavanje omogućava paralelno izvršavanje transakcija sve dok ne dođe do sukoba.
- Implementacija u kodu:
1. Dodato je polje version u klasu Equipment kako bi se omogućilo optimističko zaključavanje.
2. U prvoj niti (Thread 1), oprema se učitava sa određenim ID-jem, a zatim menja cena nakon kratkog sleepa, kako bi se mogla izvršiti druga nit.
3. U drugoj niti (Thread 2), oprema se takođe učitava sa istim ID-jem, ali cena se menja na drugu vrednost.

4. Obe niti pokušavaju ažurirati opremu. Druga nit koja prvi put sačuva izmenjeni objekat neće naići na problem. Međutim, prva nit će naići na izuzetak ObjectOptimisticLockingFailureException kada pokuša sačuvati svoje izmene zbog sukoba u verziji.
- Kako je testirana konfliktna situacija:
1. Kroz ExecutorService se obezbeđuje izvršavanje dve niti istovremeno.
2. Nastaje konflikt kada prva nit pokušava sačuvati izmenjeni objekat nakon što je druga nit već izmenila i sačuvala svoje promene.
3. Očekivano je da prva nit dobije izuzetak ObjectOptimisticLockingFailureException prilikom pokušaja čuvanja, što se i proverava u testu.

- Konfliktna situacija:
- više administratora kompanije ne može u isto vreme da menja podatke o njoj (npr. naziv, adresu, ...)

- Crtež toka koji dovodi do konfliktne situacije:



- Rešenje:

- Slično kao i za prvu konfliktnu situaciju, napisala sam test u klasi CompanyTests, koji proverava da li moj sistem rešava sledeću konfliktnu situaciju.
- 2 niti (Threada koji simuliraju administratore) pokušavaju da u isto vreme promene podatke o istoj kompaniji, tj. istoj torki tabele Company. Koristila sam optimističko zaključavanje resursa, dodavajući polje Version u klasu Company koja se menja svakom promenom torke tabele.
- Nastaje konflikt kada prva nit pokušava sačuvati izmenjeni objekat nakon što je druga nit već izmenila i sačuvala svoje promene.
- Očekivano je da prva nit dobije izuzetak ObjectOptimisticLockingFailureException prilikom pokušaja čuvanja, što se i proverava u testu.