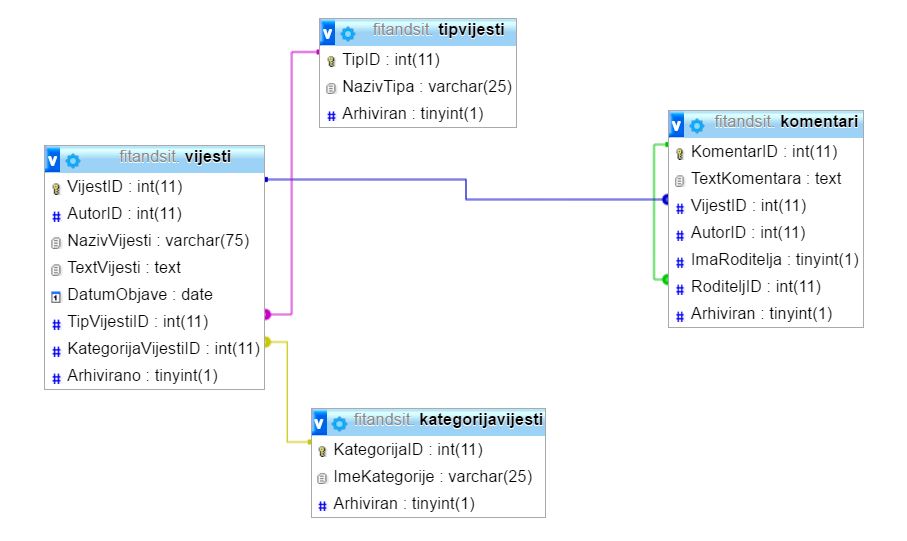
**PROJEKTNA DOKUMENTACIJA**

**-NAPREDNE WEB TEHNOLOGIJE -**

Sakib Kurtic

Zadatak 1.

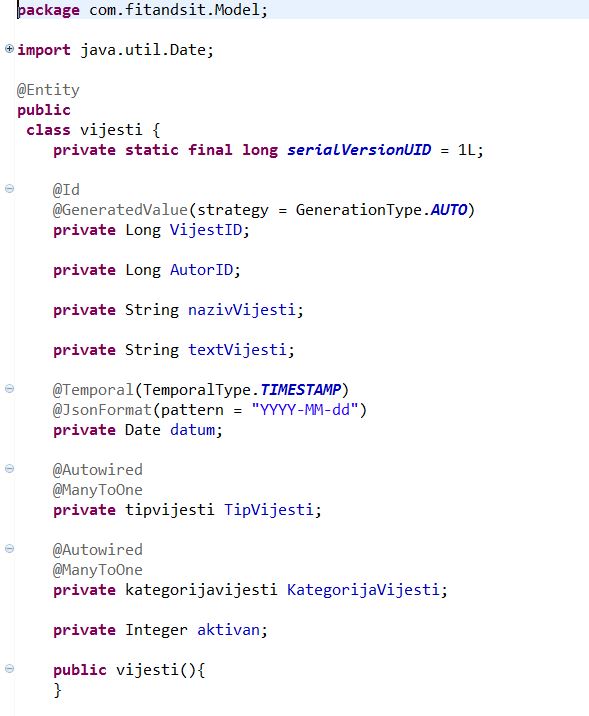
Moj prvi zadatak bio je osmisljavanje erd-a te kreiranje projekta i baze za mikroservis koji je dio aplikacije FitAndSit. Naziv modula, odnosno mikroservisa, koji mi je dodijeljen je „Vijesti“.



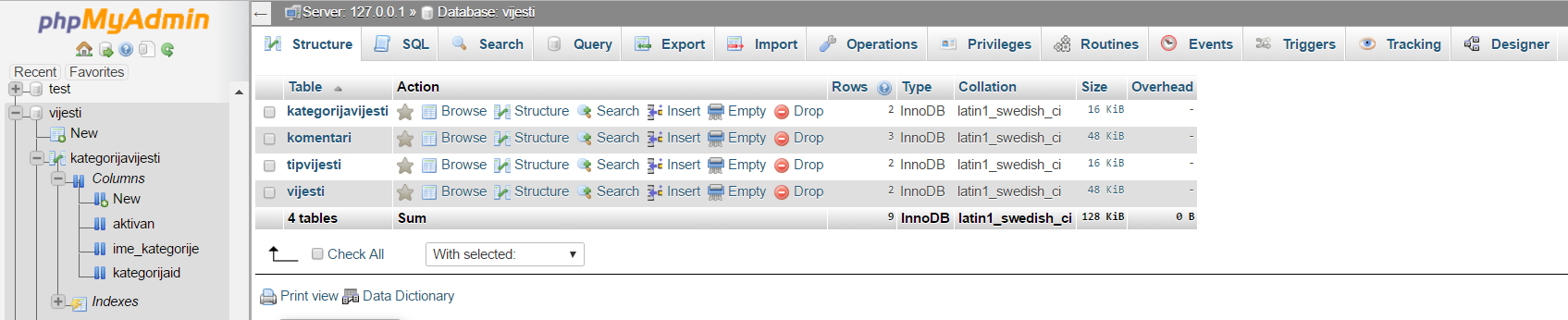
Slika 1.1. ERD baze podataka Mikroservisa Vijesti.

Mikroservis Vijesti ce trebati da ostvari povezanost sa mikroservisom korisnici, ali on nece imati direktnu vezu sa tom bazom, nego ce to biti dvije odvojene baze, ali ce mirkoservisi izmedju sebe razmjenjivati podatke. Nakon Osmisljenog erd dijagrama, Sljedeci korak bio je kreiranje projekta, te kreiranja baze podataka na mysql bazi. Najvece prepreke tokom ovog zadatka bile su u postavljanju okruzenja java-spring-jpa te svih ostalih potrebnih dependency-ja koji su bili neophodni da bi sve proradilo.

Sljedeci korak bio je kreiranje modela osmisljenih u erd-u, u projektu fitandsit za modul Vjesti. Znajuci da ce spring uz pomoc JPA-toolsa raditi code-first pristup, te nakon kreiranja svih modela, i uspostavljanja konekcije sa bazom podataka, i pokretanja projekta, znamo da bi nam se trebala kreirati baza podataka, pod uslovom da smo predhotne korake uspjesno uradili. Na slici 1.2 mozemo vidjeti model Vijesti, koji se mapira automatski uz pomoc jpa-tools-a i anotacija koje smo koristili. Na slici 1.3 mozemo vidjeti uspjesno kreiranu bazu podataka pristupom code-first.



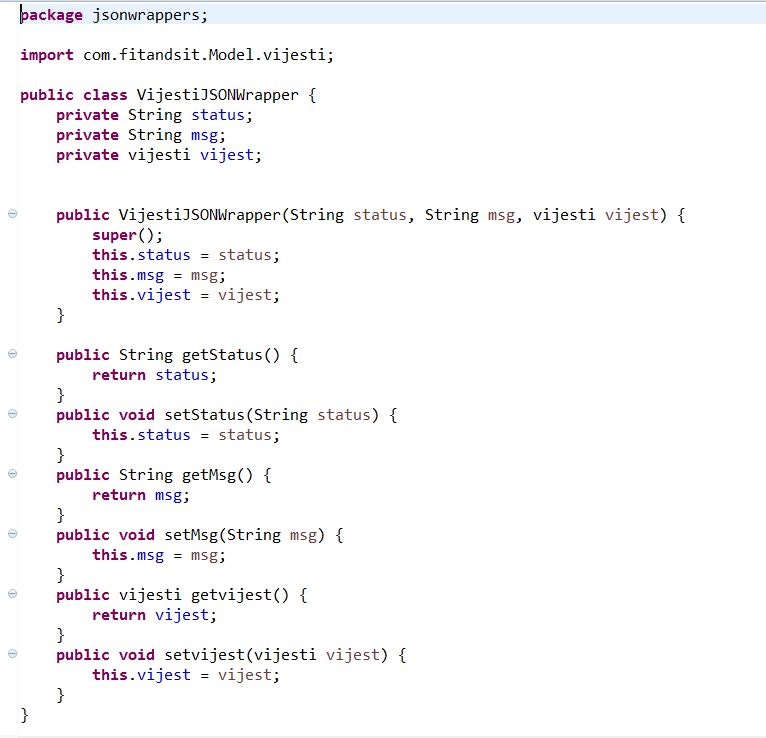
Slika 1.2 Model Vijesti (jpa entitet) sa anotacijama



Slika 1.3 Baza podataka u phpMyAdmin za mysql bazu podataka.

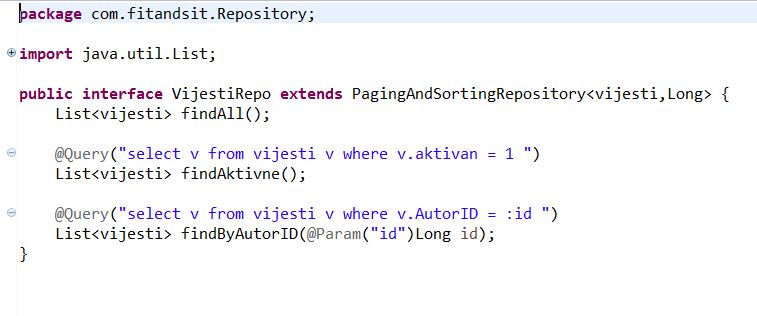
Nakon proteklih koraka, sljedeci korak bio je implementacija REST servisa za sve enititete kreirane, a koje su prikazni na erd-u na slici 1.1. U nastavku ce biti pokazana implementacija REST servisa kroz primjer klase Vijesti.

Prvi korak implementacije REST servisa bio je kreiranje wrappera oko, to jeste za svaku klasu, odnosno entitet. Wrapper nam sluzi da bi smo otklonili ne zeljene akcije i situacije, te da bi korisnik servisa bio obavjesten o responsu koji se desio. Svaki wrapper sadrzi, pored entiteta na koji se odnosi, odnosno kojeg omotava, i atribute status i poruka kako bi korisnik mogao shvatiti koji je odgovor te da li se je njegov request uspjesno izvrsio ili ne, te koji je problem ako jeste. Na slici 1.4 prikazana je implementacija wrappera za vijesti.



Slika 1.4 wrapper za klasu vijesti.

Sljedeci korak bio je kreiranje repozitorija, servisa i kontrolera, koji nam odredjuju za koji put ce koji wrapper biti vracen i koji ce podaci biti vraceni od kojeg puta, i kojeg http requesta. Odnosno svi servisi oslanjaju se na repozitorije, koji su pozvani iz kontrolera, te kada korisnik uputi zahtjev na odredjeni path, ako se on nalazi definisan u kontroleru, bit ce pozvan odredjeni servis koji ce vratiti wrapper, sa responosom.



Slika 1.5 Repozitoriji za klasu vijesti.



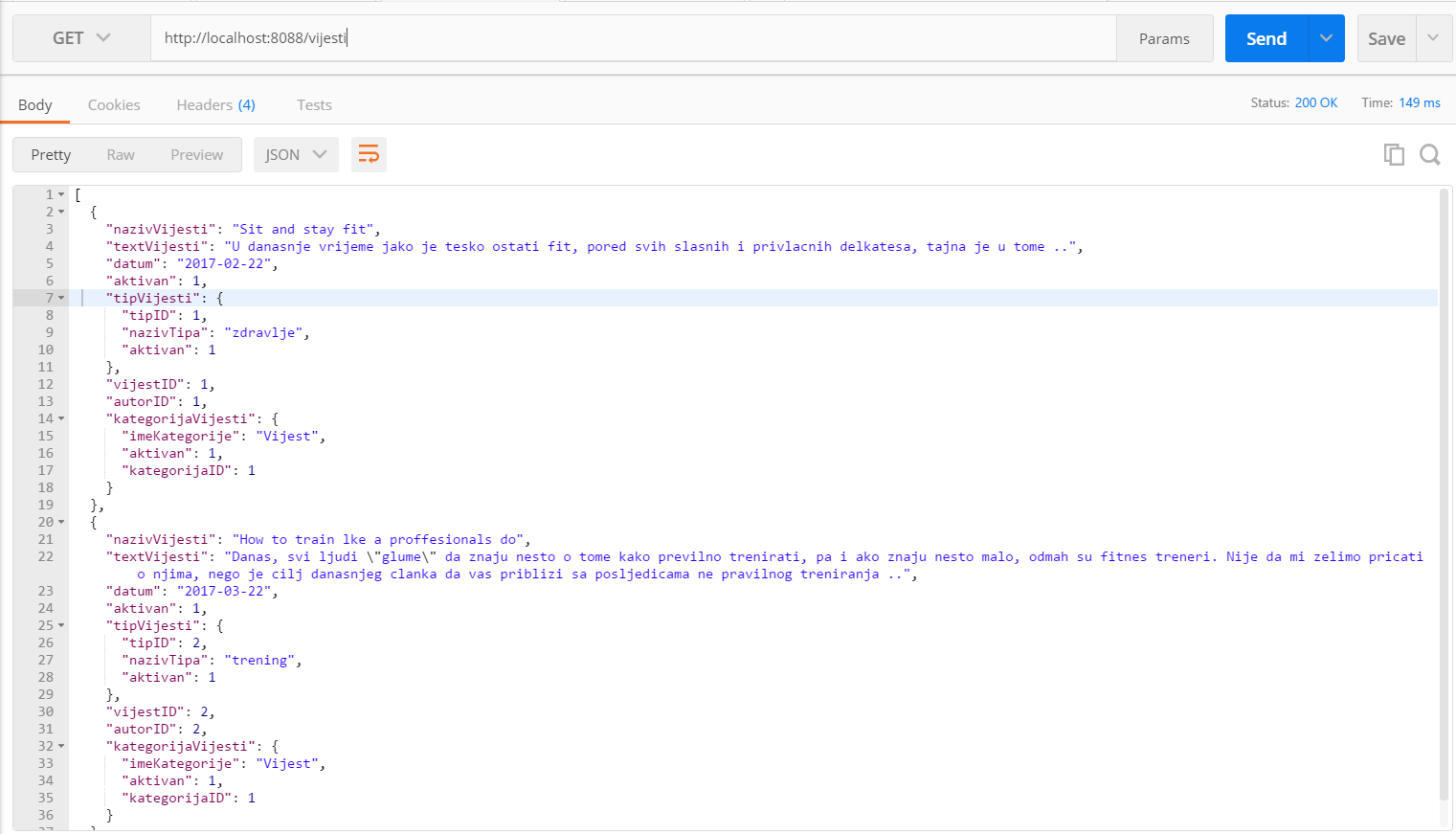
Slika 1.6 Servis za klasu vijesti



Slika 1.7 Kontroler za klasu vijesti

Nakon implementacije svih koraka, te nakon pokretanja aplikacije, trebali bi smo biti u mogucnosti za odredjeni path i request, da dobijemo podatke, sa odredjenog patha, odnosno da izvrsimo odredjene CRUD akcije uz pomoc odredjenog http requesta. Za testiranje Servisa koristen je alat postman.

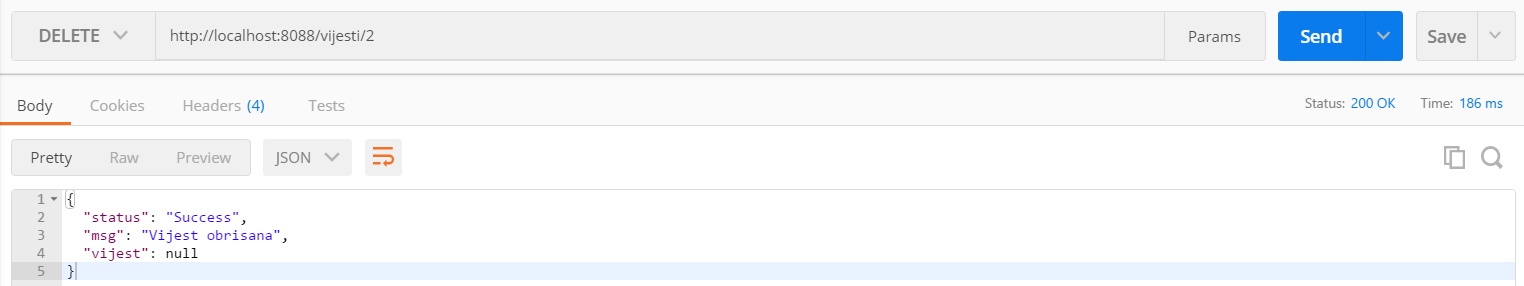
Na sljedecim slikama bit ce prikazan rad servisa vezanih za vijesti.



Slika 1.8 response na http get request na path „http:localhost:8088/vijesti“, koji ce nam vratiti sve vijesti koje nisu obrisane, tj koje su aktivne.



Slika 1.9 response na http get request na path „http:localhost:8088/vijesti/2“, koji ce nam vratiti odredjenu vijest pod odredjenim id-jem, u ovom slucaju id je 2.

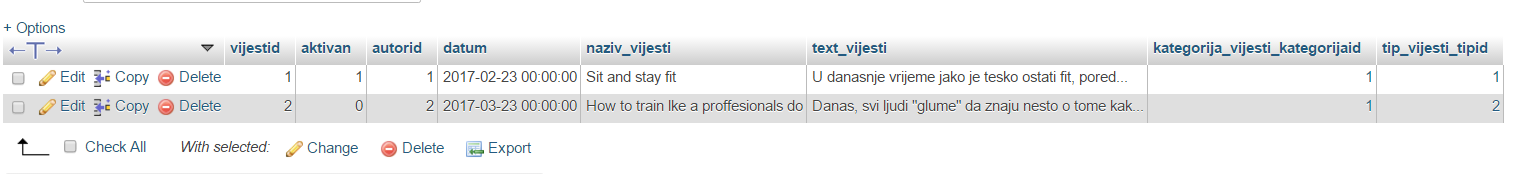


Slika 1.10 response na http delete request na path „http:localhost:8088/vijesti/2“, koji ce nam vratiti da je odredjena vijest pod odredjenim id-jem izbrisana, tj da je njen atribut aktivan podesen sa 1 na 0, u ovom slucaju id je 2.



Slika 1.11 response na http get request na path „http:localhost:8088/vijesti/2“, koji ce nam vratiti vijest pod id-jem 2, koju smo prethodno izbrisali, odnosno podesili atribut aktivan na 0.

Nakon ovih akcija trenutna situacija u bazi, u tabeli vijesti je prikazana na slici 1.12.



Slika 1.12 stanje tabele vijesti

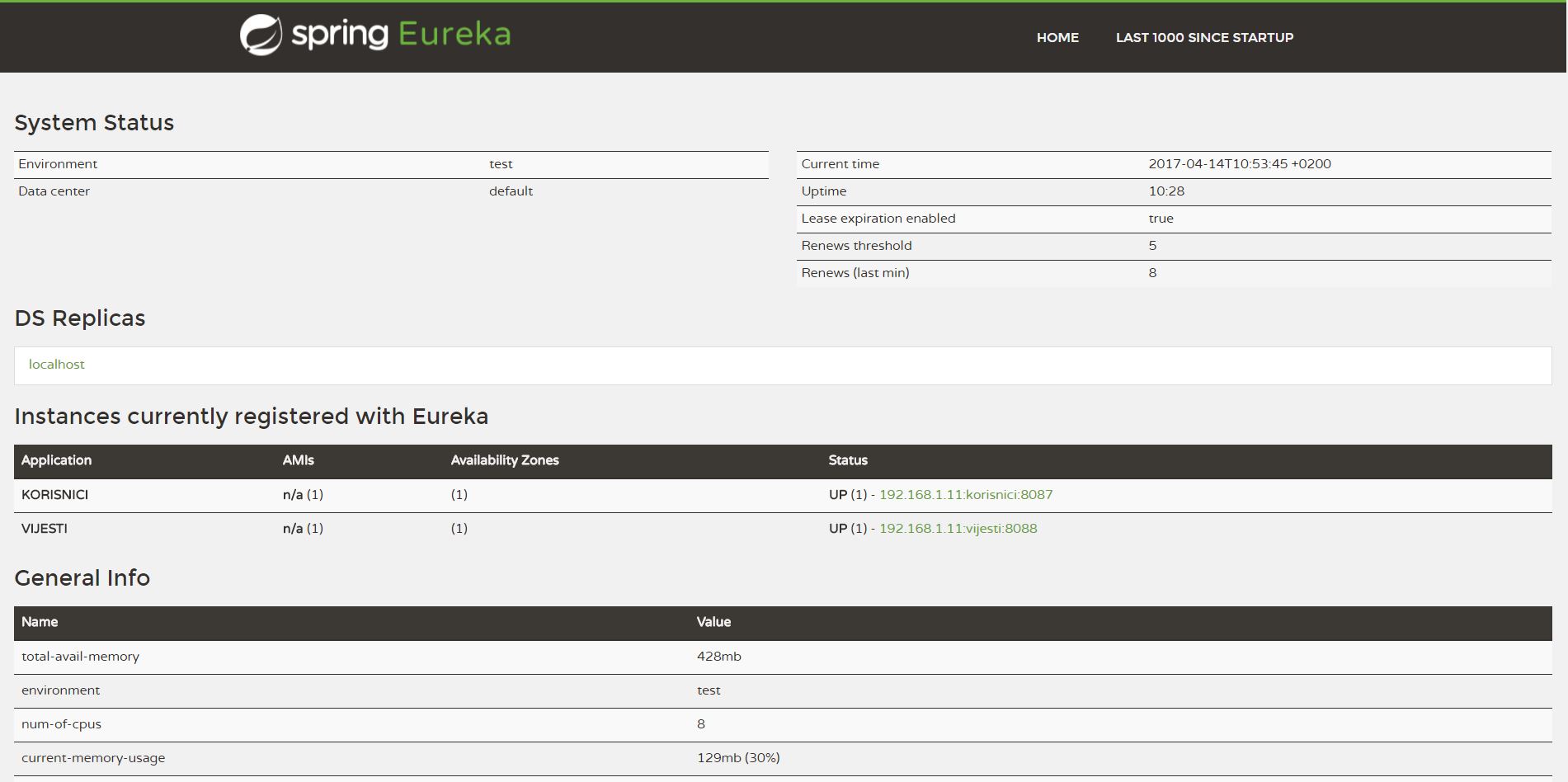
Zadatak 2.

Zadatak 2 bio je kreiranje dodatnih servisa koji ce biti potrebni kasnije, tokom implementacije frontenda, u dobavljanju, pretrazivanju, pokazivanju i izmjeni podataka. Te implementacija zaostalih stvari iz prvog zadatka. Ali za taj korak bio nam je potreban drugi modul, te ovaj zadatak je implementiran zajedno sa implementacijom 3. zadatka

Zadatak 3.

Zadatak 3 bio je kreiranje novog projekta koji ce se koristiti kao server koji ce olaksati povezivanje mikroservisa i kreiranje novog projekta koji se se koristiti kao centralni server sa kojeg ce ostali projekti kupiti postavke za pojedinacne mikroservise. Preciznije receno, zadatak 3 bio je povezivanje mikroservisa koji ce trebati komunicirati, u ovom slucaju mikroservisi Vijesti ce trebati komunicirati sa mikroservisom Korisnici. Tako da je kreiran Eureka server, kojem su dodane odredjene anotacije, a na svaki od mikroservisa je dodana anotacija koji ce omoguciti eureki da pronadje klijenta, tj modul.

Na slici 3.1 prikazan je uspostavljen eureka server na kojem je uocljivo da su pokrenute po 1 instanca mikroservsa Vijesti i mikroservisa korisnici.



Slika 3.1 Eureka server sa pronadjenim instancama mikroservisima

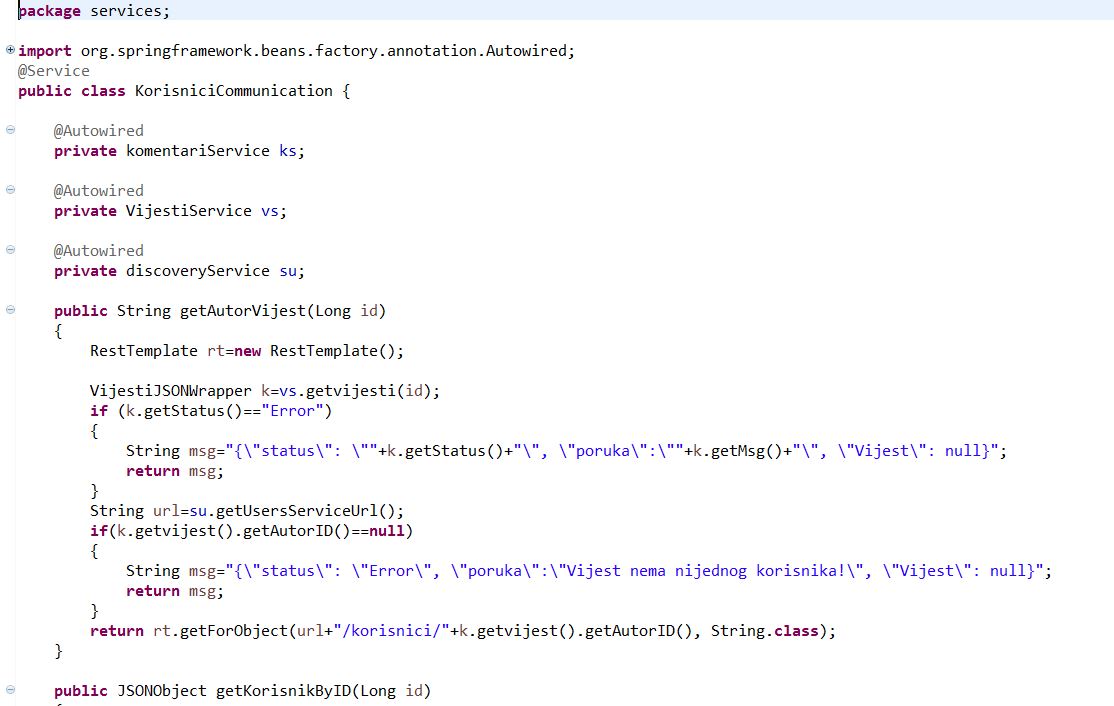
Nakon uspjesnog „podizanja“ instanci 2 mikroservisa, uslijedilo je spajanje ta dva mikroservisa, odnosno uspostavljanje komunikacije izmedju njih. Bilo je potrebno napraviti servis, koji ce vratiti odredjene podatke, a komunikacija izmedju korisnika i vijesti ostvarena je implementacijom discovery servisa koji je dostavio url koji treba discovery servisu, te uz pomoc eureke, preusmjeren je zahtjev gdje treba. Implementaciju discovery servisa prikazan je na slici 3.2



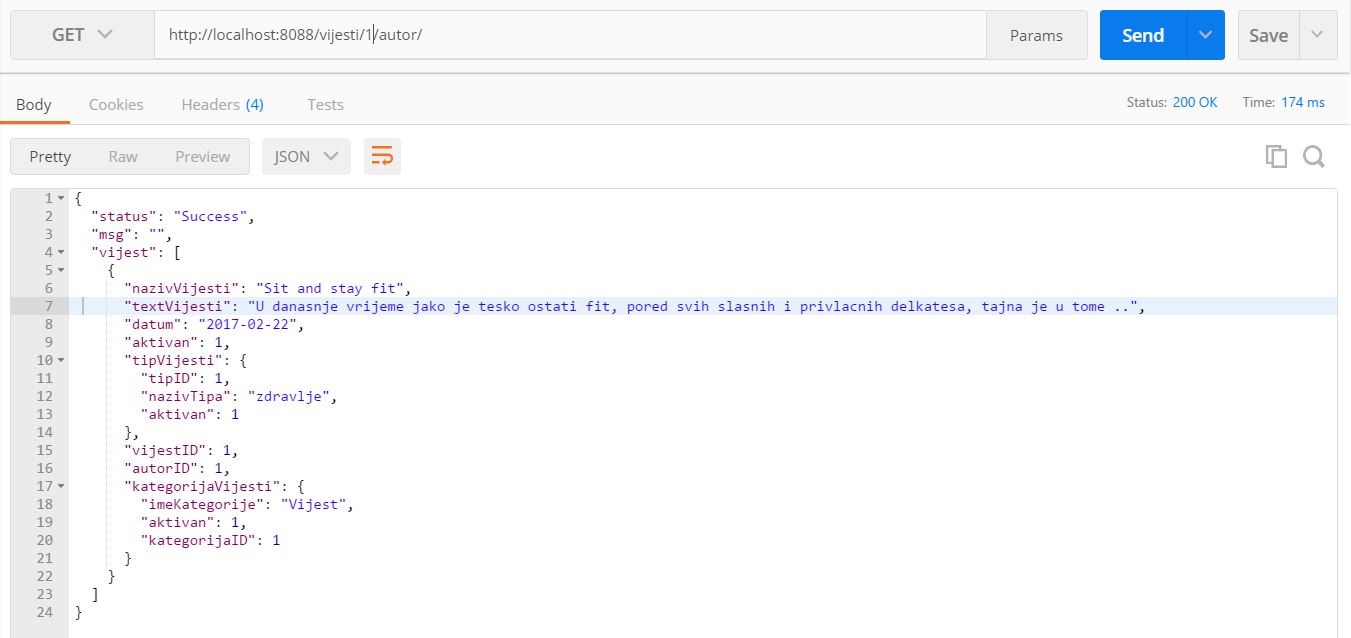
Slika 3.2 discovery servis

Nakon povezivanja bilo je moguce implementirati odredjeni servis potreban za odredjene servise. Takodjer povezivanje je omogucilo potpunu funkcionalnost servisa, jer smo sada u mogucnosti i provjeriti da li korisnik sa odredjenim id-jem postoji, te shodno tome, mozemo reci da smo i implementirali odredjene verifikacije koje pomazu izradi kvalitetnijem back-end-a.

Na slici 3.3 prikazana je klasa koja je poveznica sa mikroservisom korisnici, a na slici 3.4 prikazan je response http get requesta, koji komunicira sa mikroservisom korisnici



Slika 3.3 Komunikacija sa korisnicima



Slika 3.4 response na http get request na path „http:localhost:8088/vijesti/1/autor“, koji ce nam vratiti vijesti odredjenog korisnika.