Tema:

FirmInternalApp

Opis teme:

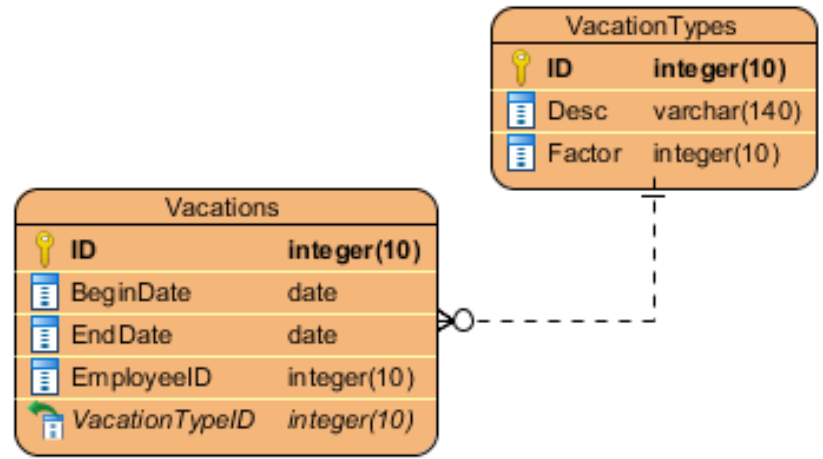
Svaka ozbiljnija IT firma posjeduje aplikaciju koja se koristi interno za operacije poput upravljanja korisnicima, dokumentima i zahtjevima. FirmInternalApp je upravo primjer takve aplikacije, koja olakšava svakodnevni rad u firmi, omogućava pregled relevantnih dešavanja i novosti.

Aplikacija će raspolagati sa 4 modula, te će omogućiti sljedeće funkcionalne zahtjeve: upravljanje obavijestima, upravljanje godišnjim odmorima, upravljanje dokumentima, kao i dvije vrste korisnika i privilegije shodno dodijeljenom tipu. Ukoliko se radi o administratoru, dodatne privilegije bi omogućile brisanje i dodavanje dokumenata. Uposlenik HR tima će se razlikovati po mogućnostima odobravanja ili odbijanja zahtjeva za godišnji odmor. Svi ostali korisnici će imati pristup predaji zahtjeva, pregledu uposlenika po timovima, dodavanju novih timova i sl.

### Modul za godišnje odmore

Modul zadužen za kreiranje, editovanje, brisanje i pregled godišnjih odmora. Ove funkcionalnosti ovise od tipa korisnika.

ERD dijagram za ovaj modul izgleda ovako:

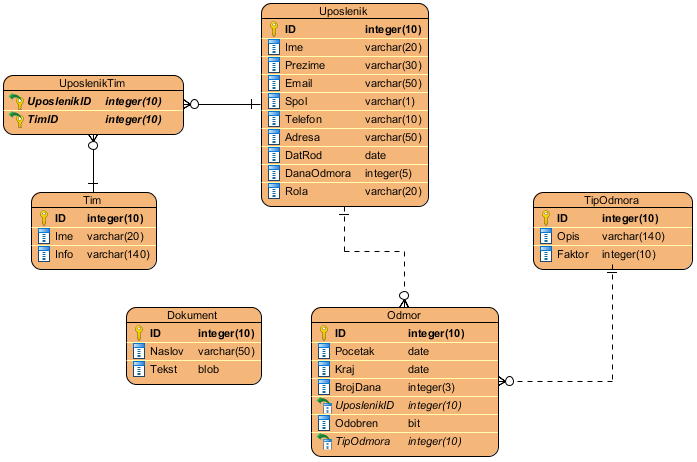


Ilustracija Vacations ERD

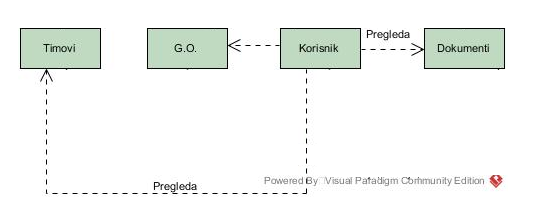
Veze sa drugim modulima:

* **Korisnici**: svaki korisnik moze slati zahtjev za godišnjim odmorom

ERD:

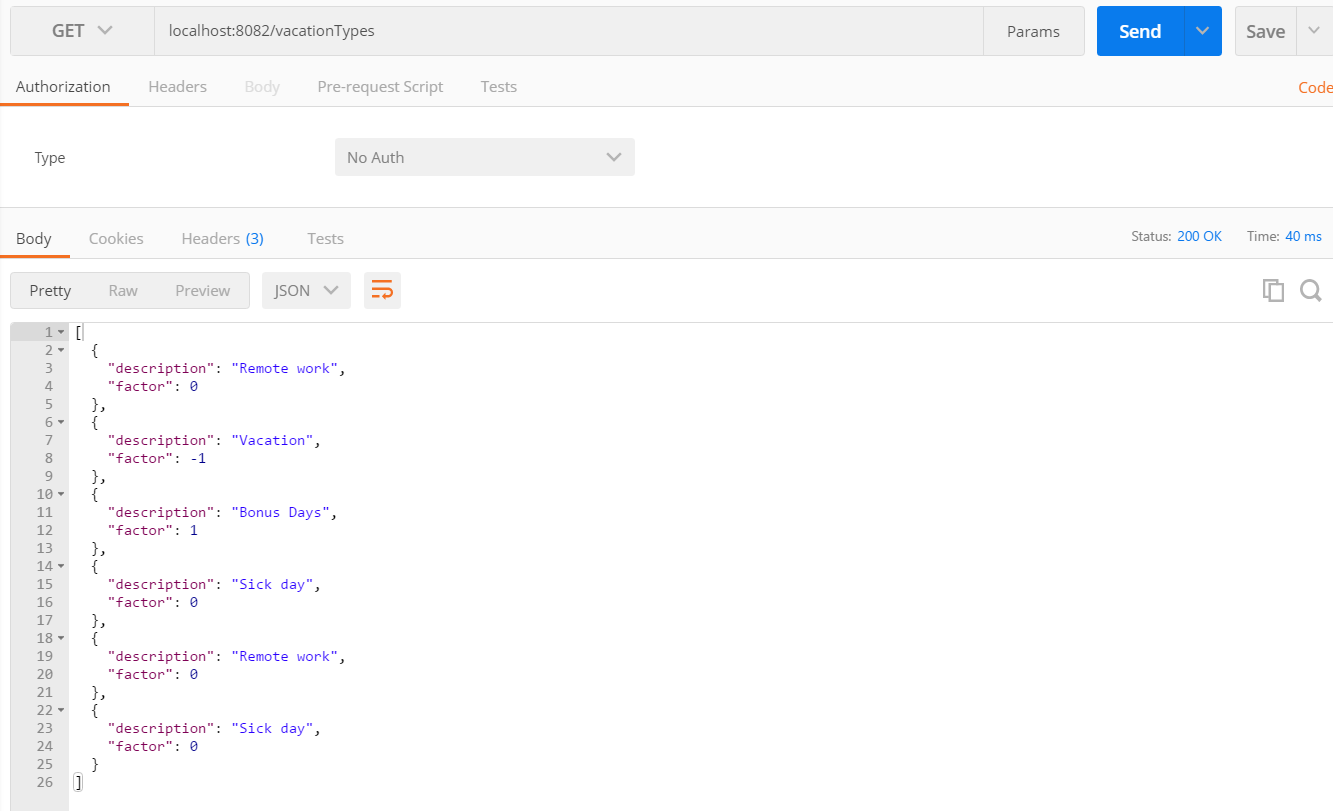


Veze između modula:

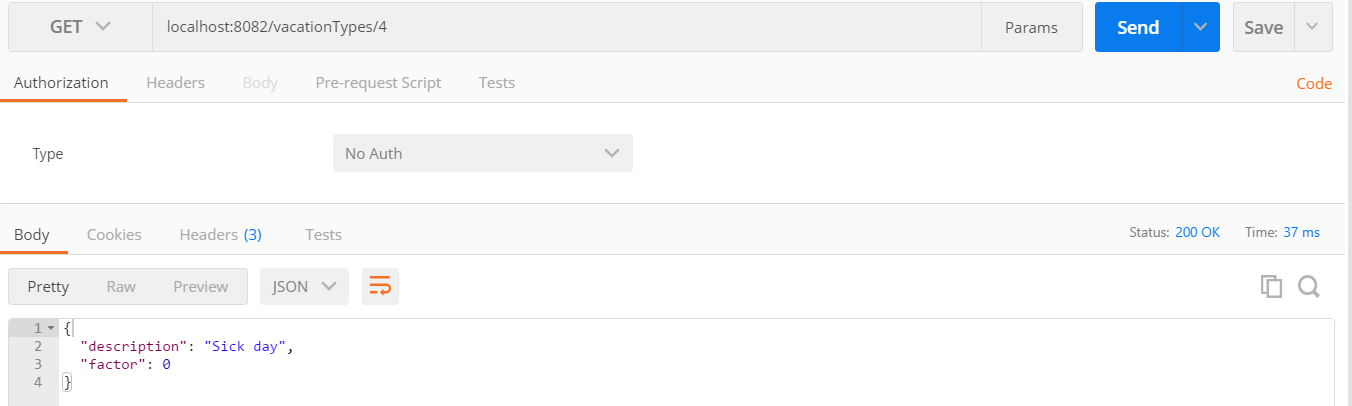


### CRUD za Vacations modul:

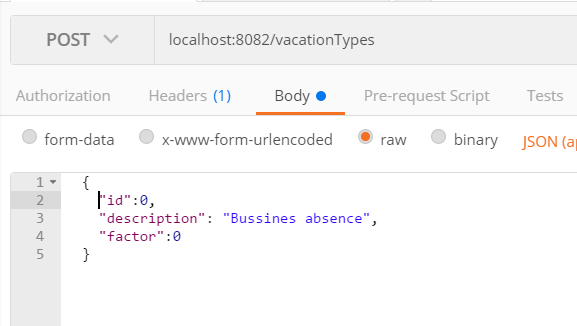
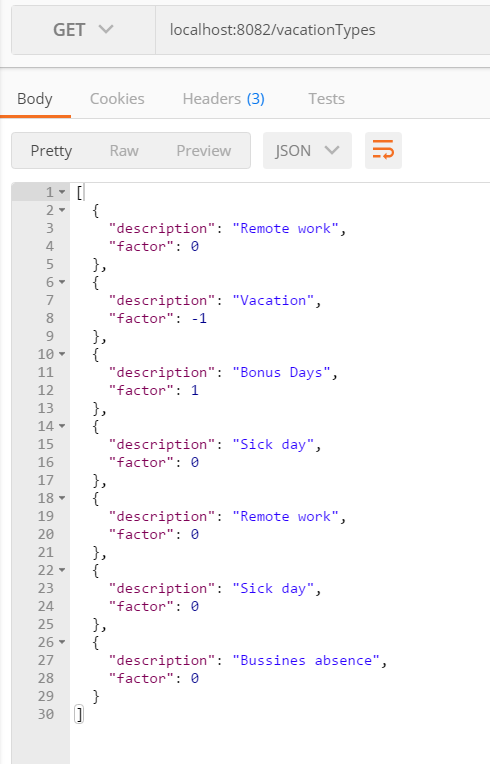
#### VacationTypes: GET ALL



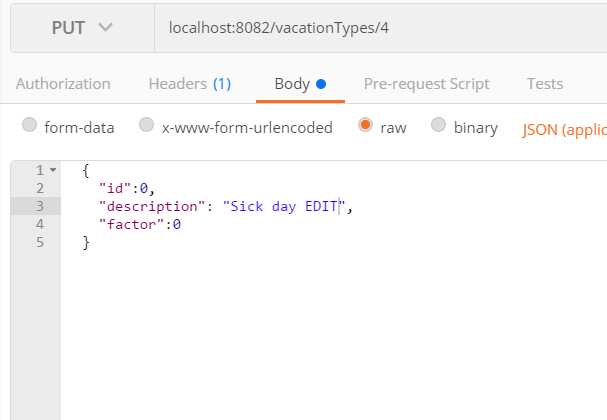
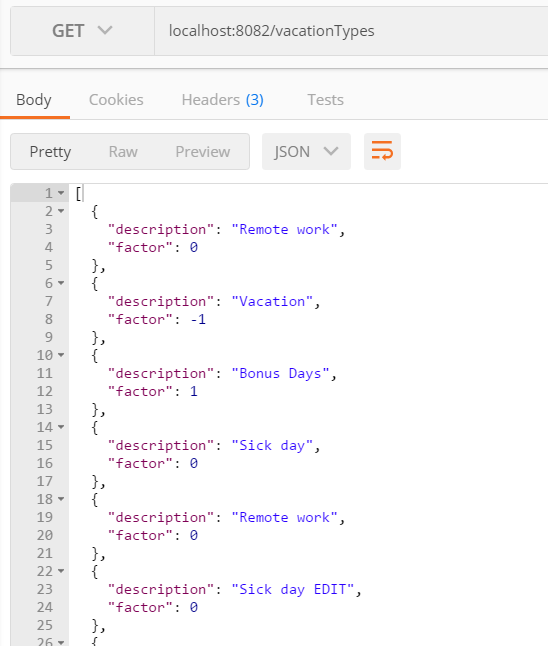
#### VacationTypes: GET BY ID



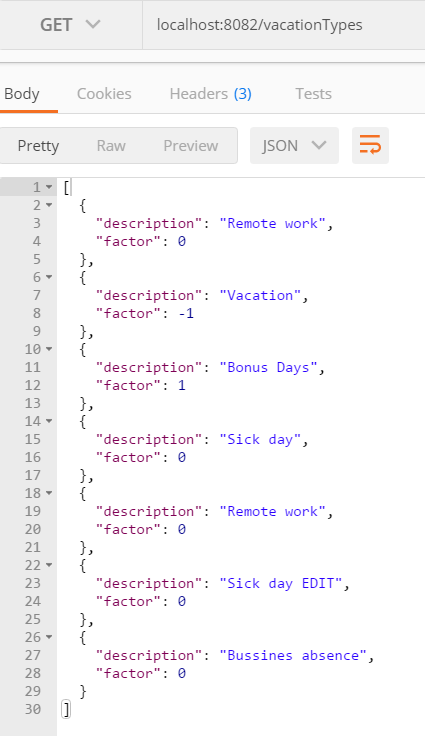
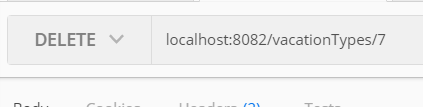
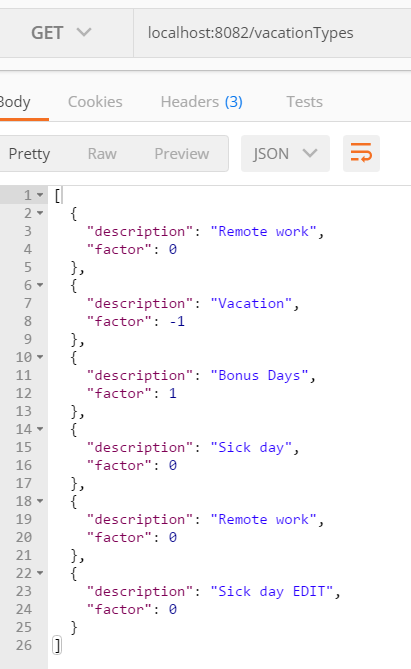
#### VacationTypes: POST

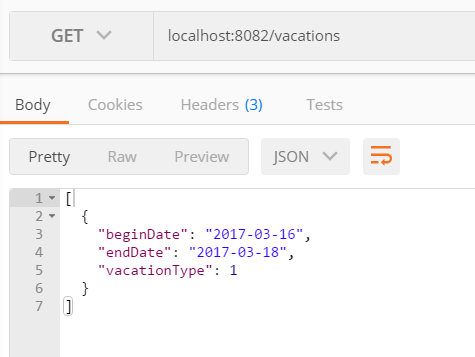
#### VacationTypes: PUT

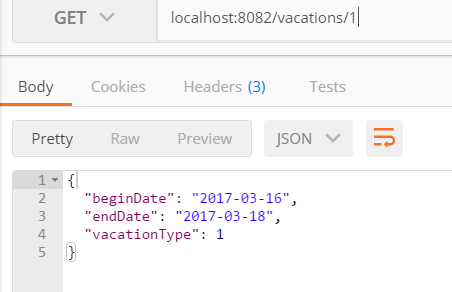
#### VacationsModule: Delete

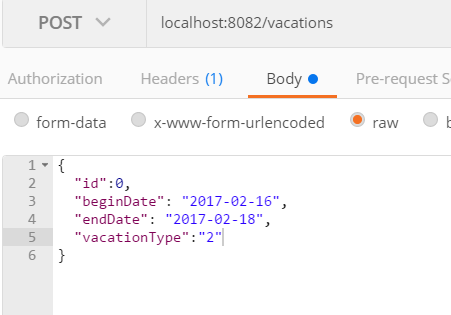
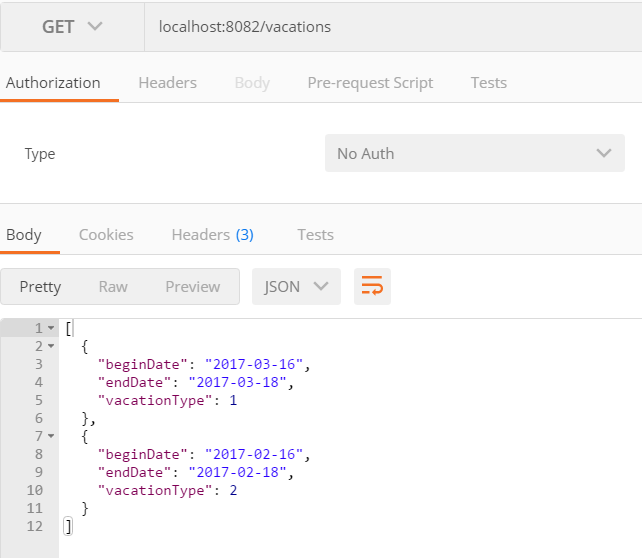
#### Vacations: GET ALL



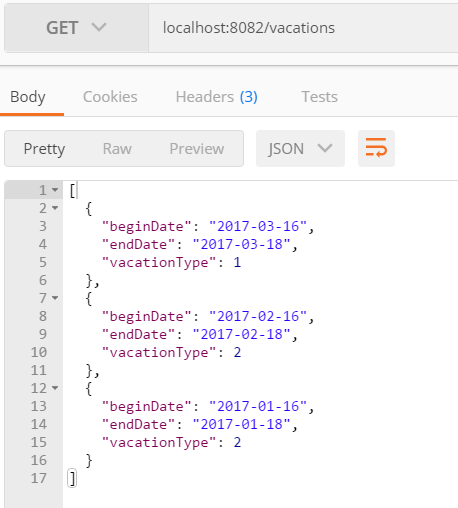
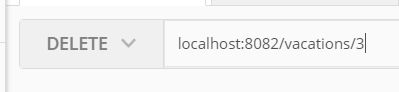
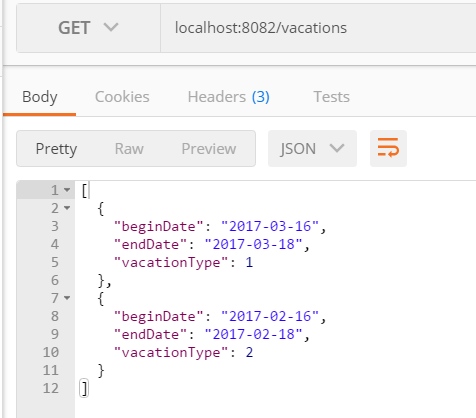
#### Vacations: GET BY ID



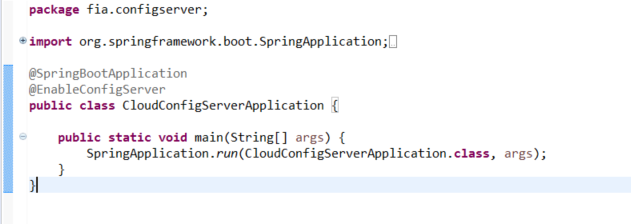
#### Vacations: POST

#### Vacations: DELETE

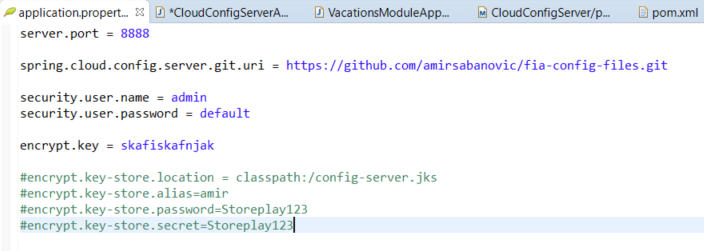
  

Prateci uputstva za eureka server sa c2, prvi korak je bio napraviti novi spring boot app projekat. Dodale smo dependency server eureka i anotaciju @EnableConfigServer kako bi naglasile da se radi o eureka serveru.

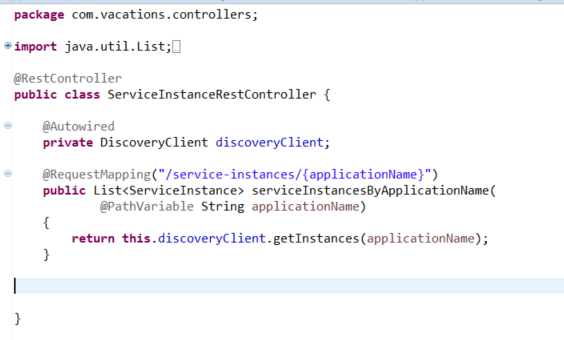
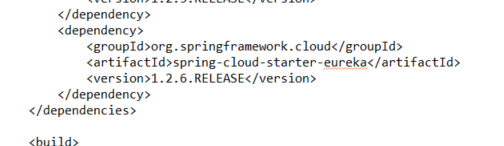


Svako je za svoj modul na poseban repository postavio properties za svoj modul sa podacima potrebnim za taj modul (koji je zapravo config server)

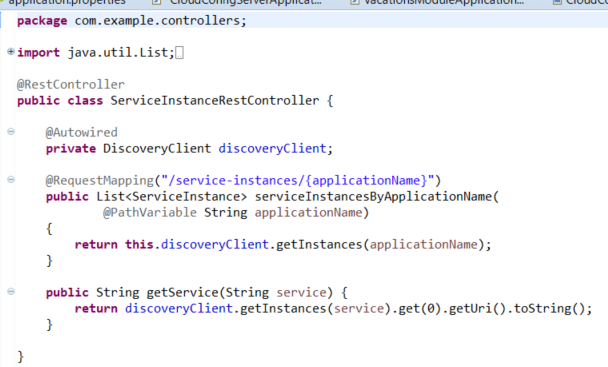
U application.properties servera smo postavili server port, username i password, te link od gore navedenog repozitorija, s kojeg se kupe podaci.



Nakon toga sam za svoj modul kreirala klasu ServiceInstanceRestController. U klasu koja pokrece modul sam dodala @EnableDiscoveryClient (i importovala package vezan za njega), i to je omogućilo komunikaciju izmedju modula i Eureka servera. Također dodala sam dependency u pom.xml za eureka client.

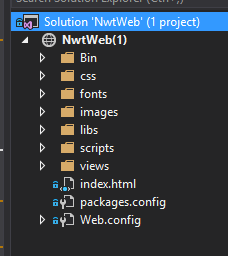
 

Obzirom da je za module Users i Teams potrebna informacija o drugim modulima (jer šalju update), dodale smo i funkciju getService koja vraća link na modul koji je potreban. (ukoliko se posalje teams-client vratit ce link na teams modul)

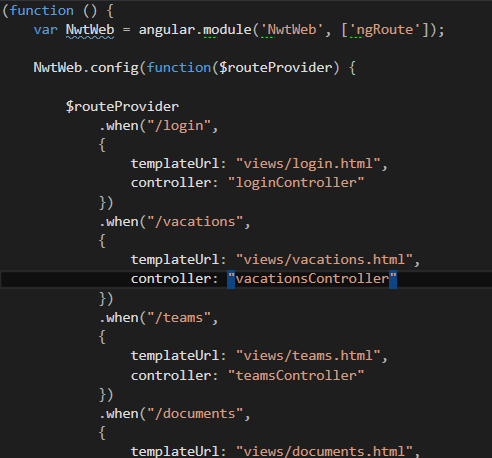


Angular

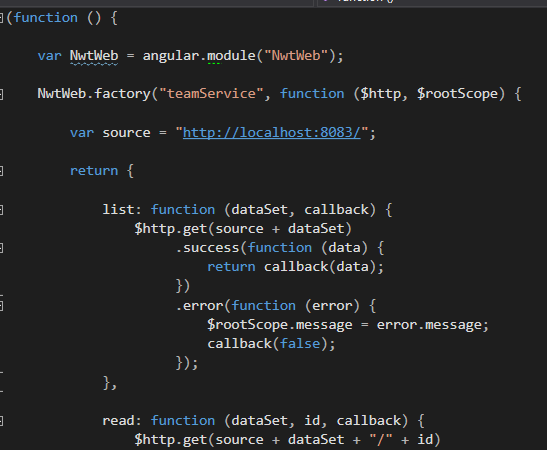
Prije svega smo kreirali novi Web site (File – New – Website). Solution smo organizovali na sljedeci nacin:



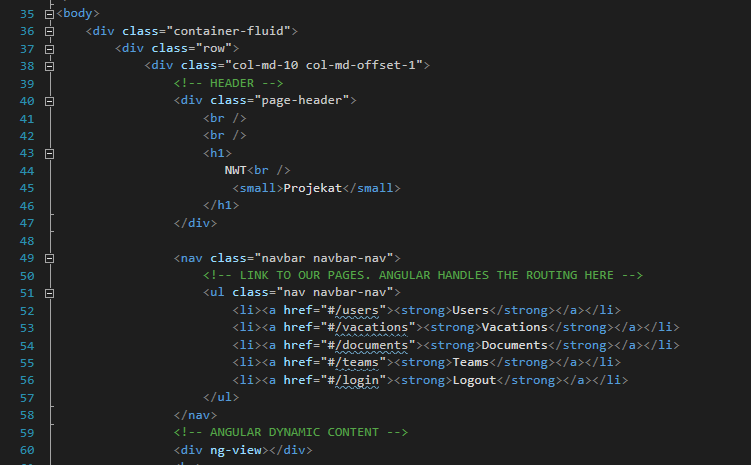
Dodali smo app.js file koji sadrzi podatke o tome koja ruta treba prikazivati koji html, i koji kontroler je vezan za taj view.



Zahtjevima (get, post, put, delete) ce upravljati servisi. Kreirala sam teamService.js kao servis vezan za timove, i postavila da podatke cita sa porta 8083.

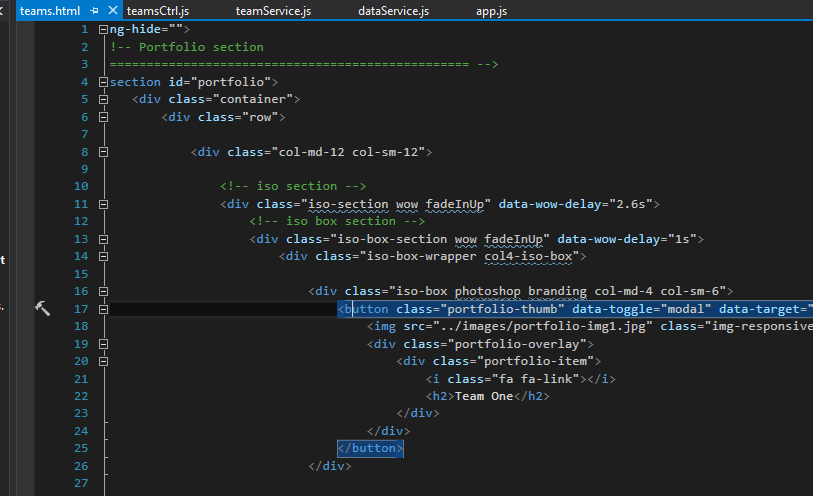


Svi file-ovi su referencirani u index.js. Ovo je single page aplikacija, i nas meni se nalazi u index.html, dok se ostatak stranice renda u <div ng-view> dijelu.

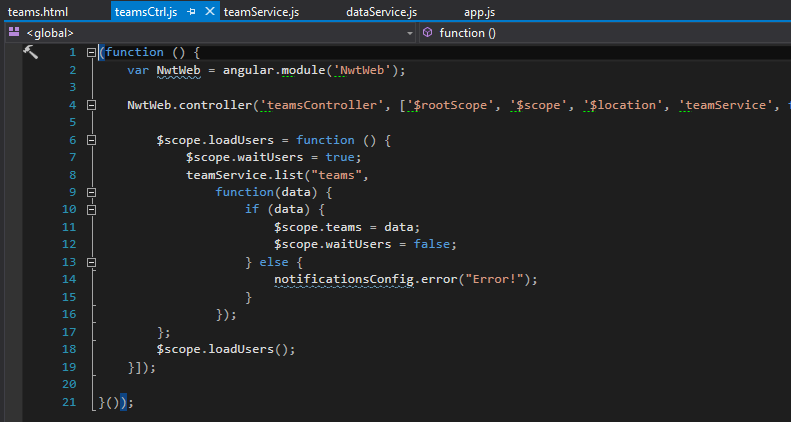


Za ovaj put, uzela sam da radim na teams-u. Trenutno su mockani podaci.

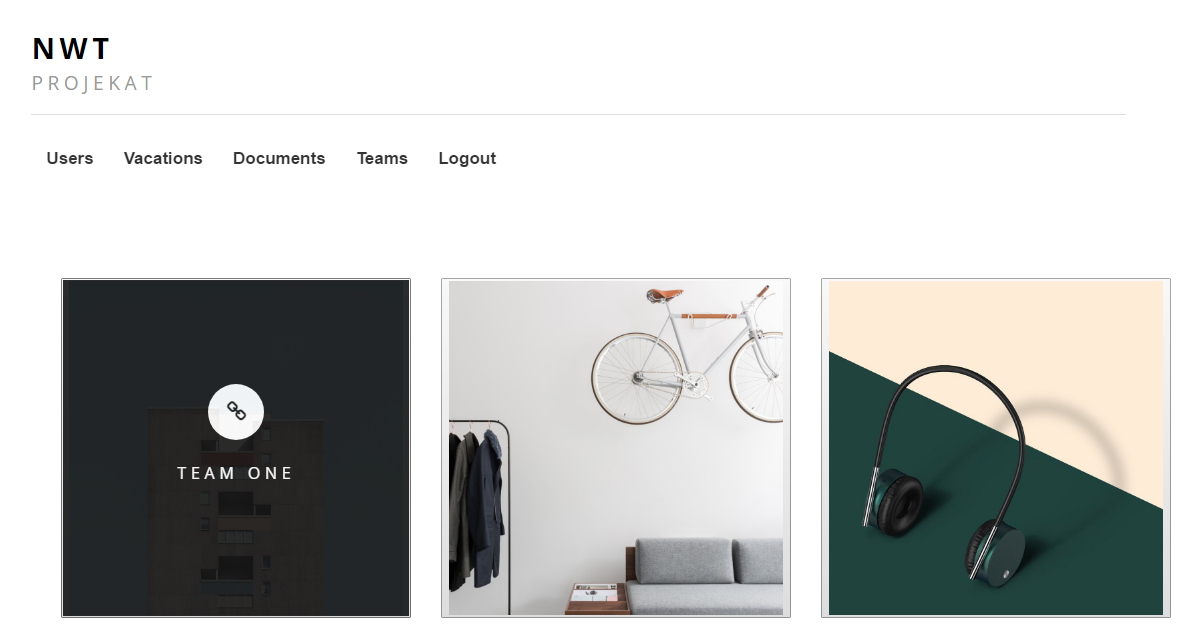
HTML:



JS:



Sa mock-anim podacima, teams view izgleda ovako:



Na hover se prikazuje ime tima, a na klik se otvaraju podaci o clanovima tima.

