Model perzistencije

FoodPin

Članovi tima:

Tea Mitić 17274
Dimitrije Mitić 17269

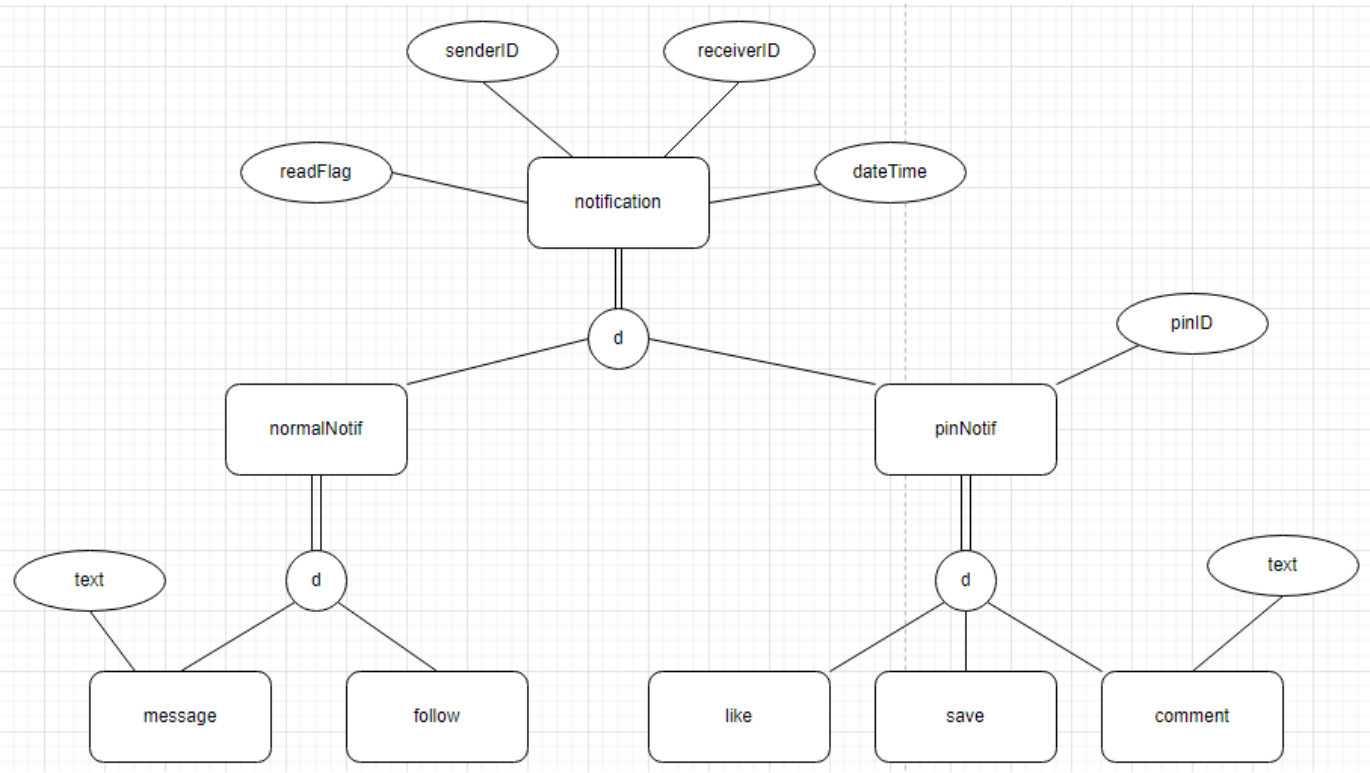
Sadržaj

Modeli podataka

FoodPin projekat sadrži dve baze podataka - Neo4j, kao primarnu bazu za skladištenje podataka, i MySQL kao log bazu za perzistenciju notifikacija i poruka između klijenata.

Modeli podataka projekta FoodPin prikazani su na slikama ispod:

MySQL EER model



Model je zamišljen tako da postoji osnovni tip notifikacije koji ima atribute:

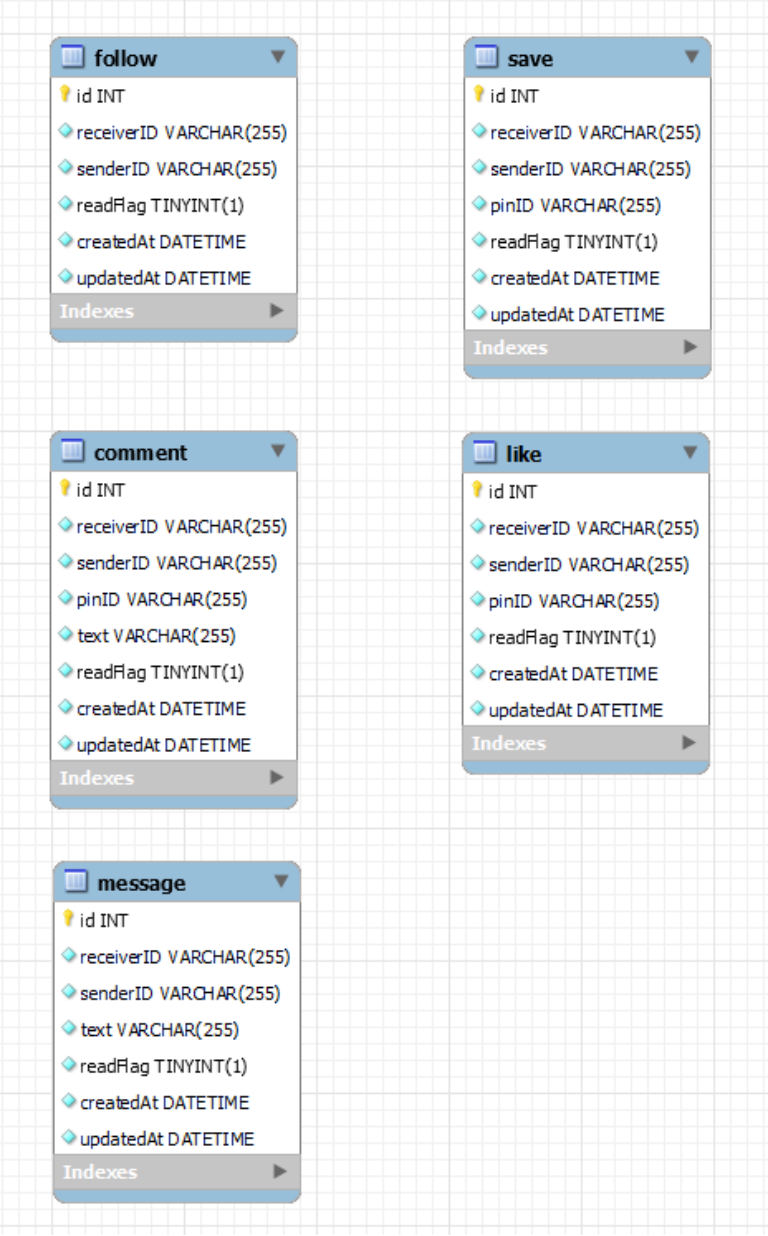
- id osobe čija akcija je kreirala notifikaciju,
- id osobe kojoj treba da stigne notifikaija,
- dateTime atribut koji se sastoji od dve kolone - datum i vreme kreiranja notifikacije i datum i vreme ažuriranja notifiakcije (ovo je automatski dodato od strane ORM mapera Sequelize za node.js),
- flag za pročitanu notifikaciju kako bi se na frontu znalo da li treba prikazati nepročitanu notifikaciju.

Iz osnovnog modela se izvode druga dva modela notifikacije. Prvi model je vezan za pin (objavu u aplikaciji) i sadrži id te objave. Notifikacije za pin su stvorene interakcijom user-a sa samim pinom - pinNotif. Drugi model se odnosi na notifikacije stvorene interakcijom između user-a – normalNofit.

Iz gore navedenih notifikacija se nasleđuju po dva konkretna tipa notifikacija:

- *MessageNotification* - za logovanje poruka između klijenata, sadrži atribut tekst poruke,
- *FollowNotification* - za logovanje notifikacija o zapraćivanju drugih korisnika aplikacije,
- *LikeNotification* - za logovanje notifikacija o lajkovanoj objavi,
- *CommentNotification* - za logovanje notifikacija o postavljenom komentaru na objavi,
- *SaveNotification* – za logovanje notifikacija o sačuvanoj objavi od strane user-a.

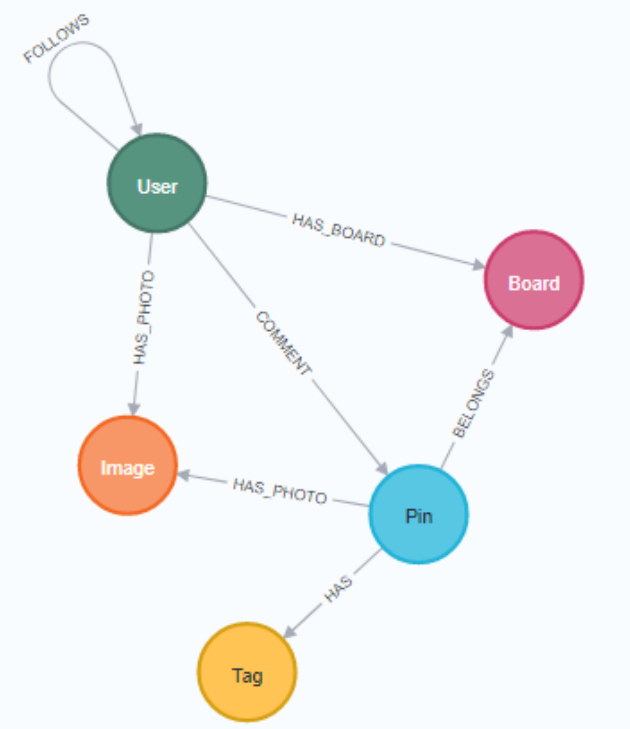
MySql model mapirana baza podataka na relacioni model prikazana je na sledećoj slici:



EER model se razlikuje od implementiranog relacionog modela zbog načina mapiranja EER modela. Izabran je ovaj tip mapiranja zbog preglednosti log podataka u bazi.

Neo4j model

Neo4j model podataka je prikazan na sledećoj slici:

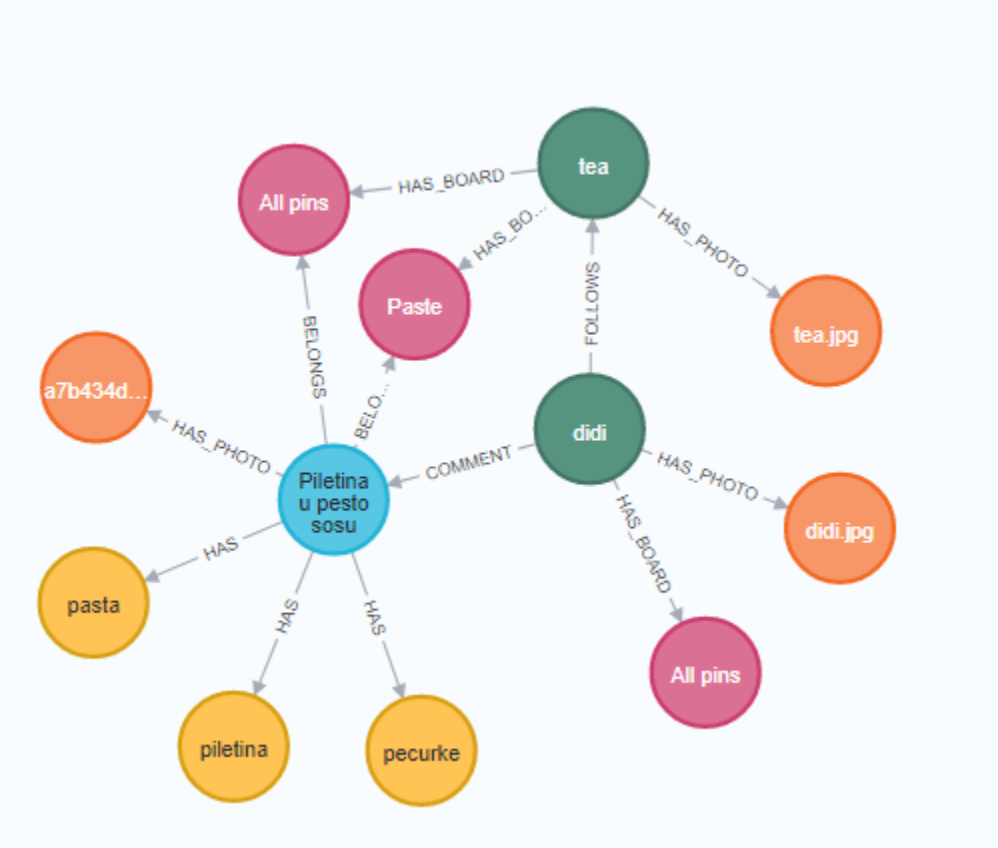


Kao što je već navedeno Neo4j graf baza podataka je implementirana da bude primarno skladište podatka. Na slici se vide tipovi čvorova u bazi i njihove međusobne veze.

Kratak opis Neo4j modela:

- *User čvor* - čuva konkretne podatke o korisniku, kao što su username, password, ime, prezime i slični atributi.
- *Board čvor* - sadrži objave (Pinove) u sebi i predstavlja medijum između pin-ova i korisnika. Prednost korišćenja board-ova je organizacija pin-ova radi lakšeg korišćenja aplikacije.
- *Pin čvor* - sadrži sve neophodne podatke vezane za jednu objavu na društvenoj mreži (naziv, opis, sliku itd). Jedan pin pripada samo jednom user-u.
- *Tag čvor* - služi za dodatno opisivanje pin-ova kako bi se omogućio sistem preporuke sličnih pin-ova.
- *Image čvor* – sadrži sliku objave ili user-a.
- Veza *Follows* između user-a označava da jedan user prati drugog user-a.
- Veza *Has-Board* između user-a i board-a označava da user ima taj board na svom profilu i može u njemu da skladišti svoje pin-ove, i čuva tuđe.
- Veza *Belongs* između pin-a i board-a označava da pin mora pripadati nekom board-u.
- Veza *Has* između pin-a i tag-a označava da pin mora da ima tagove zbog sistema preporuke.
- Veza *Comment* između user-a I pin-a označava da je user postavio komentar na objavi/pin-u.
- Veza *Has-Image* između image I user-a I image I pin-a označava da user/pin ima sliku.

Primer podataka u Neo4j bazi:



Mehanizam mapiranja

Za Neo4j bazu podataka je iskorišćena biblioteka Neode koja omogućava lakše kreiranje šeme graf baze I olakšava pisanje CYPHER upita. Korišćenjem Neode biblioteke implemenitran je Repository pattern kako bi se kreirao sloj perzistencije u projektu i omogućio jedinstven interfejs sloju biznis logike.
Repository pattern je implementiran instanciranjem jedne instance Neo4j servera u projektu u fajlu Persistance/neo4j/config.js odakle se includuje ta instanca u DataProvider klasama gde se izvršavaju upiti ka Neo4j bazi.

Za MySQL bazu iskorišćena je biblioteka Sequelize. To je ORM za node.js koji omogućava rad sa mysql i drugim relacionim bazama podataka. Iskorišćena je ista logika implementacije Repository pattern-a. Sequelize instanca povezana sa mysql bazom se nalazi u config fajlu na putanji Persistance/mySql/config/mySql-config.js. I ovde su implementirane DataProvider klase za izvršavanje upita.