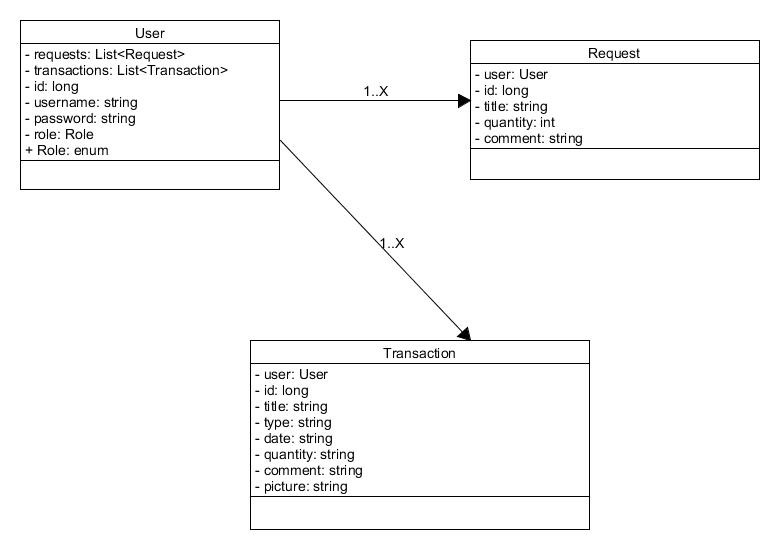
# FamilyBudge – Dokumentáció

## A fejlesztői környezet és eszközök bemutatása:

A project a NetBeans nevű fejlesztői környezetben készült a Maven Project, mely egy olyan fejlesztői eszköz, amivel könyen lehet elhasználni más modulokat/plug-in-eket és leszedi ehez a szükséges dependencyket(függőségeket). Valamint a Spring Boot felhasználásával, mely egy olyan eszköz melyel könnyen és gyorsan lehet prototipusokat létrehozni. Továbbá felhasználtuk a h2 adatbázis motort is a projectben, mely Java SQL adatbázist vezényel, valamint a Lombok nevű plug-int aminek köszönhetőleg nem kell gettereket írni, valamint összehasonlító(equals) metódusokat.

## Adatbázis terv:



## Alkalmazott könyvtárstruktúra bemutatása:

A forrás fájlok a familybudge nevű főkönyvtárban található, mely több alkönyvtárra ágazik szét:

* annotation
* config
* controller
* entity
* repository
* service

A fő könyvtárban található a FamilyBudge.java fájl mely a program belépésí pontja.

Az annotation könyvtáron belül található a Role.java

A config könyvtáron belül található az AuthorizationInterceptor.java fájl.

A controller könyvtáron belül található a RequestController.java, TransactionController.java, UserController.java fileok.

Az entity könyvtárban találhatóak Request.java, Transaction.java, User.java fileok.

A repository könyvtárban található a RequestRepository.java, TransactionRepository.java, UserRepository.java fileok.

A service könyvtárban található a RequestService.java, TransactionService.java, UserService.java , UserNotValidException.java fileok.

## Végpont-tervek és leírások:

A controllerek valósítják meg a végpontokat.

A UserController a "/api/user" végpontra dolgozik. A metódusai meg ezen belül osztódnak el 3 végső végpontra.

A register metódus a "/register" végponton fogadja a bejövő adatokat.

A login metódus a "/login" végponton fogadja a bejövő adatokat.

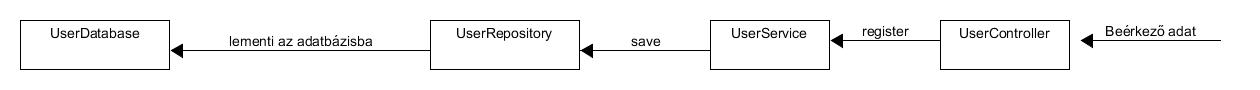
A logout metódus a "/logout" végponton fogadja a bejövő adatokat.

A TransactionController a "/api/transaction" végpontra dolgozik. Itt igazából mindig az lesz a végpont ami az adott transaction id-je(azonosítója). Ezek a metódusok dolgoznak a végponra: create, update, read, delete.

A RequestController a "/api/request" végpontra dolgozik. Itt igazából mindig az lesz a végpont ami az adott request id-je(azonosítója). Ezek a metódusok dolgoznak a végponra: create, read, update, delete.

## 1db végpont működésének leírása:

Vegyük pl a UserController osztályt. A UserController osztály a "/api/user" végpontra dolgzik. Ezen belül vegyük pl a register metódusát ami a "/register" dolgozik az elözőn belül, tehát a "/api/user/register" végpontra fog érkezni a kérés. Ezt a Controller elkapja, majd mivel csak a register metódus van ezen a végponton a register metódus tovább küldi a UserService register nevű metódusának, mely aztán a létrehozott usernek beállít még egy rolet(User vagy Admin) majd meghívja a UserRepository save metódusát, mely elmenti az adatbázisban a beadott adatokat. És itt véget ér a folyamat.



## Fontosabb Specifikumok:

Nincsenek különleges specifikációk.