

Java alapú webes keretrendszerek
Házi feladat Dokumentáció:

InvoiceManager
(Számlakezelő app)

Biró Bence
SM5P0J
Dátum: 2023.05.03

Feladat:

A webalkalmazás feladata, hogy felhasználók jogaiktól függően képesek legyenek a számláikat megnézni, módosítani, törölni és létrehozni. Valamint a felhasználókat is lehet benne törölni és jogaikat módosítani, valamint lehet regisztrálni a rendszerbe.

Környezet:

IntelliJ IDE fejlesztői környezetben, Windows 10 operációs rendszeren, Java 17-es verzióval, Maven 4.0.0 és Spring Boot 3.0.2 használva.

Forráskód struktúrája:

A Maven könyvtárstruktúráját használom. Az alkalmazás 3 rétegre van bontva. + réteg a következő : Repository, Service, Controller. Ezek mindegyike, csak az alatta lévőre hivatkozik, tehát Controller csak a service-re(megeshet, hogy a Repository-ra), Service csak a Repository-ra, Repository pedig az adatbázissal kommunikál.

- /pom.xml – projekt build fájl
- /src/main/java – java forráskódok
 - Fő package név : com.exmaple.invoicemanager
 - Alkönyvtárak:
 - DTO – Egyes entitásokhoz tartozó Data Transfer Object-ek
 - /libs.Error – Egyes Exception-ök találhatóak itt
 - Model – Entitások definíciói
 - Repository – Entitásokhoz tartozó JpaRepository-k
 - Security – Authentikációhoz és Authorizációhoz tartozó konfigurációk és modellek
 - Service – Entitásokhoz tartozó üzleti logikák
 - Thymeleaf – Thymeleaf-hez tartozó konfiguráció
 - Web – Oldalak és jogok alapján meghatározott Kontrollerek és handler-ek
- /src/main/resources – nem fordítandó de csomagolandó fájlok
 - /static/js – templétekhez tartozó .js fájlok
 - /static/styles – temlétekhez tartozó .css fájlok
 - /templates/layout – oldalakhoz tartozó közös sablon
 - /templates – oldalak html fájlja
 - application.yml – egyes statikus értékek beállítása
 - data.sql – sql kód az adatbázis default feltöltéséhez

Maven függőségek:

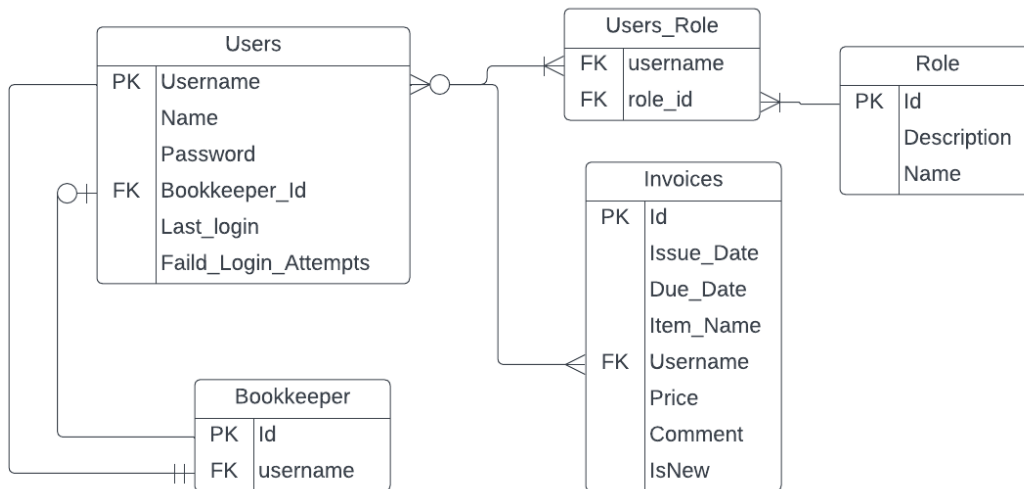
A forráskódban a pom.xml fájlban találhatóak a függőségek, a „dependencies” tag alatt ebben a sorrendben. Ezek további implicit függőségeket hozhatnak magukkal. Fejlesztés kezdetén legfrissebb verziókat használtam.

Spring Boot-hoz kapcsolódó függőségek verziói megegyeznek a Spring Boot verziójával(3.0.2).

- Spring Boot JPA – 3.0.2
- PostgreSQL – 42.5.1
- Spring Boot Web – 3.0.2
- Lombok – 1.18.24
- Spring Boot Security – 3.0.2
- Spring Boot Thymeleaf – 3.0.2
- Thymeleaf SpringSecurity6 – 3.1.1.RELEASE
- Thymeleaf Layout Dialect – 2.4.1
- Spring Boot Test – 3.0.2
- Spring Security Test – 6.0.1

Adatbázis felépítése:

PostgreSQL adatbázist használtam. Adatbázis neve „invoice”. Az URL-je a application.yml-ben található.



Tanulság:

- Primary Key csak nagyon kivételes esetben ne Id legyen, kód olvashatóság és egyszerűség szempontjából
- Figyelni bidirectional kapcsolatokra az adatbázisban, ne legyen redundáns megvalósítás az adatbázisban

Entitások

User:

- username azonosítja
- eltárolja a felhasználó adatait
- rámutat a könyvelőre, aki kezeli a felhasználó számláit
- szerepköröket egy set-ben tárolja
- eltárolja mikor jelentkezett be utoljára
- eltárolja, hány sikertelen bejelentkezési próbálkozása volt a felhasználónak
 - nem működik
 - plus funkcióhoz segítene(captcha)

Role:

- az egyes szerepkörök is adatbázisban vannak tárolva
- 3 szerep van: USER, BOOK, ADMIN
- szerepek képességei:
 - USER
 - megtudja nézni a számláit listában és részletesen is és tudja törölni is őket
 - BOOK
 - saját és klienseinek a számláit megtudja nézni, módosítani és törölni, valamint új számlát is tud létrehozni
 - ADMIN
 - megtudja nézni a saját számláit
 - felhasználókat tud törölni
 - módosíthatja a felhasználók szerepeit

Invoice:

- eltárolja a számlák adatait
- rámutat a felhasználóra, akié a számla
- a státusz is elmenti, hogy volt-e már részletesen megnyitva

Bookkeeper:

- BOOK szereppel rendelkező felhasználók vannak benne, saját könyvelő Id-val
- valamit elérhetőek a könyvelőhöz tartozó klienseket

Repository-k:

Minden entitáshoz tartozik egy Repository, amely megvalósítja a JpaRepository és ezen keresztül tudunk kommunikálni az adatbázissal.

