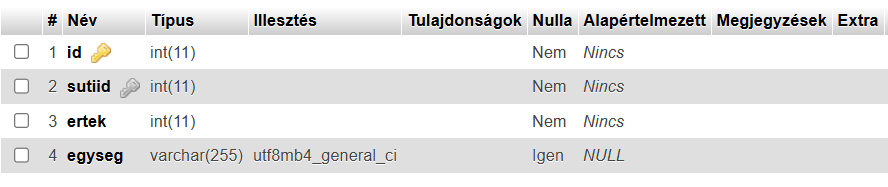
***Java Alkalmazások Gyakorlat Beadandó Dokumentáció***

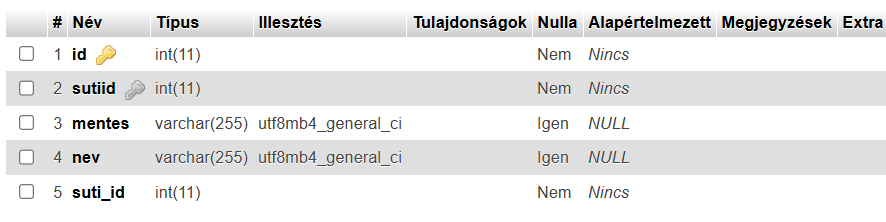
Github link: <https://github.com/HamNorb/Cukraszda2>

***Adatbázis leírása***

A kiindulási adatbázisunk 3 táblát tartalmazott, amelyek a következők voltak:



**1. ábra**

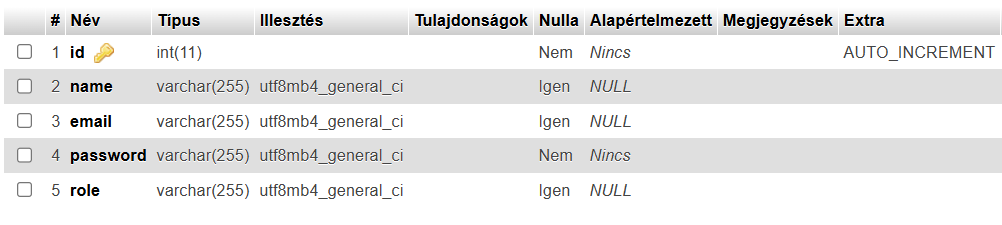


**2. ábra**

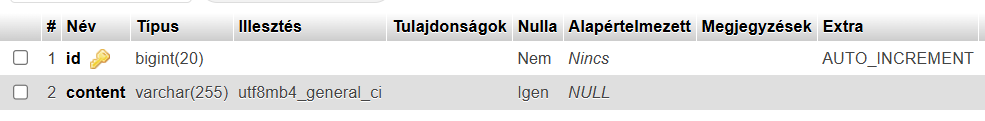


**3. ábra**

Ez lett kiegészítve egy users táblával, hogy megvalósítsuk a felhasználók kezelését, illetve egy uzenet táblával, hogy a kapcsolatok oldalt is megtudjuk valósítani.

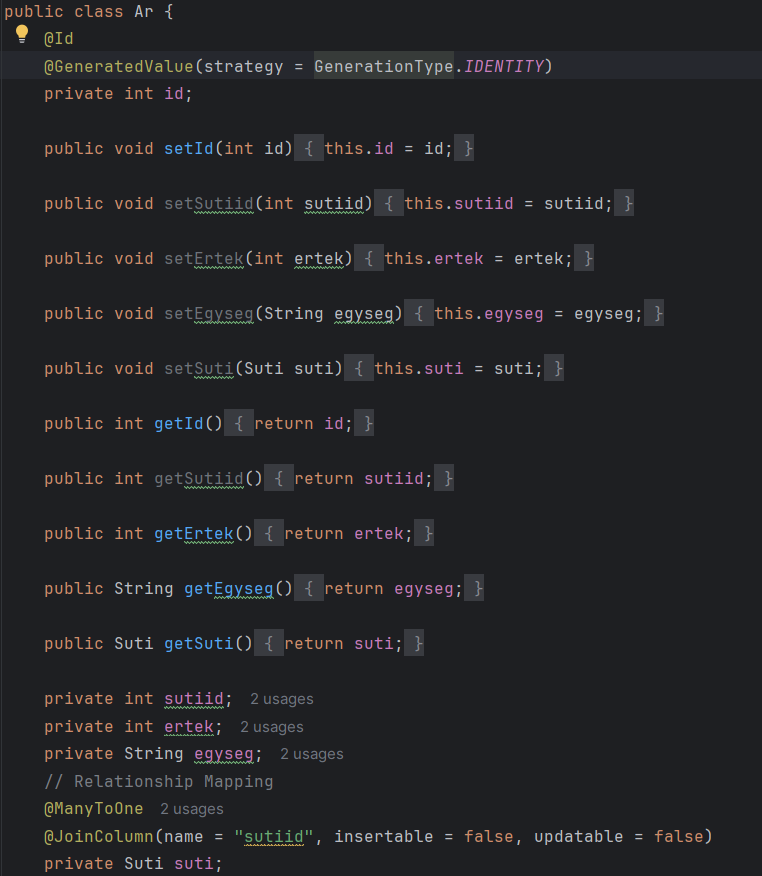


**4. ábra**



**5. ábra**

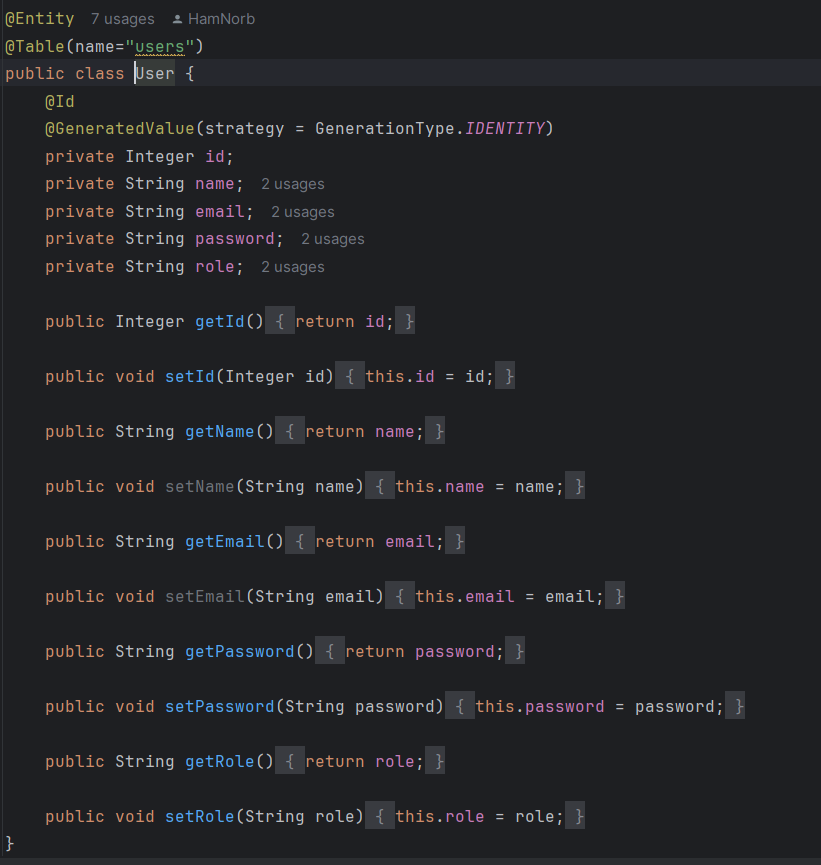
Az adatbázisok tábláit az alábbi modellekkel és hozzájuk tartozó repositorykkal (ezekről csak egy képet mellékelünk az egyszerűség kedvéért) reprezentáltuk a forráskódban:



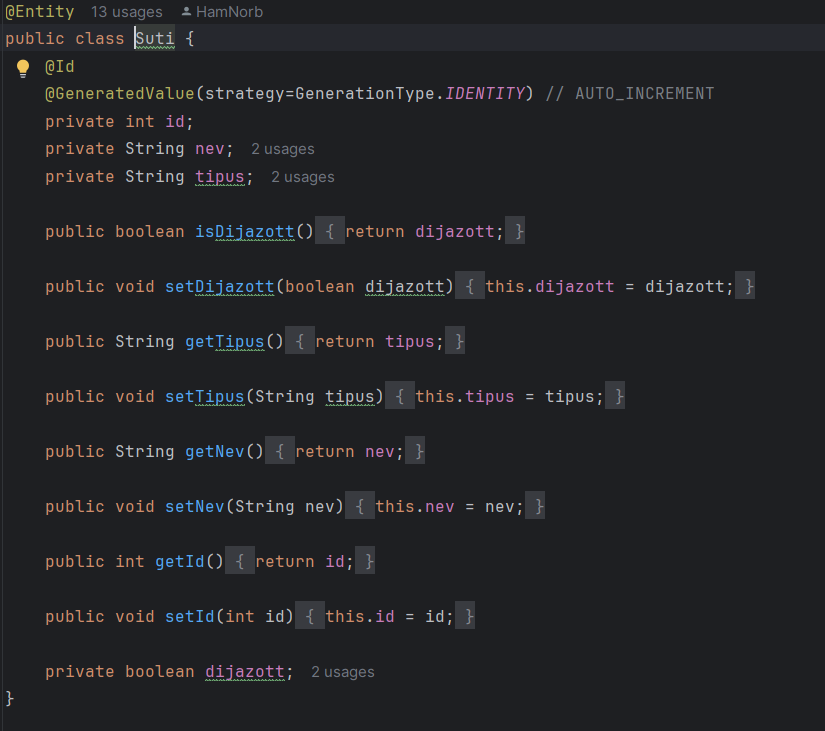
**6. ábra**



**7. ábra**



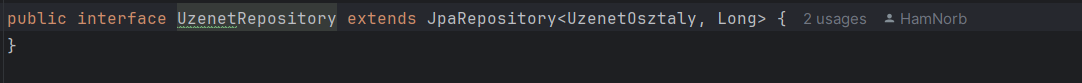
**8. ábra**



**9. ábra**

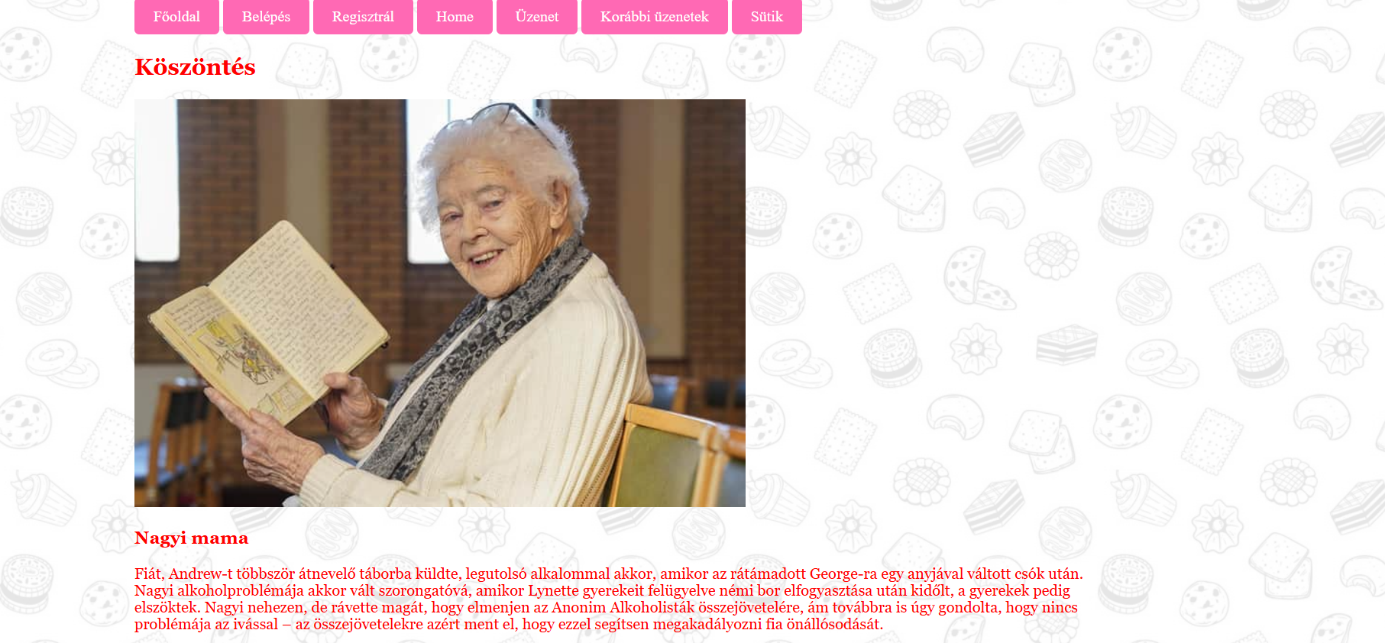


**10. ábra**

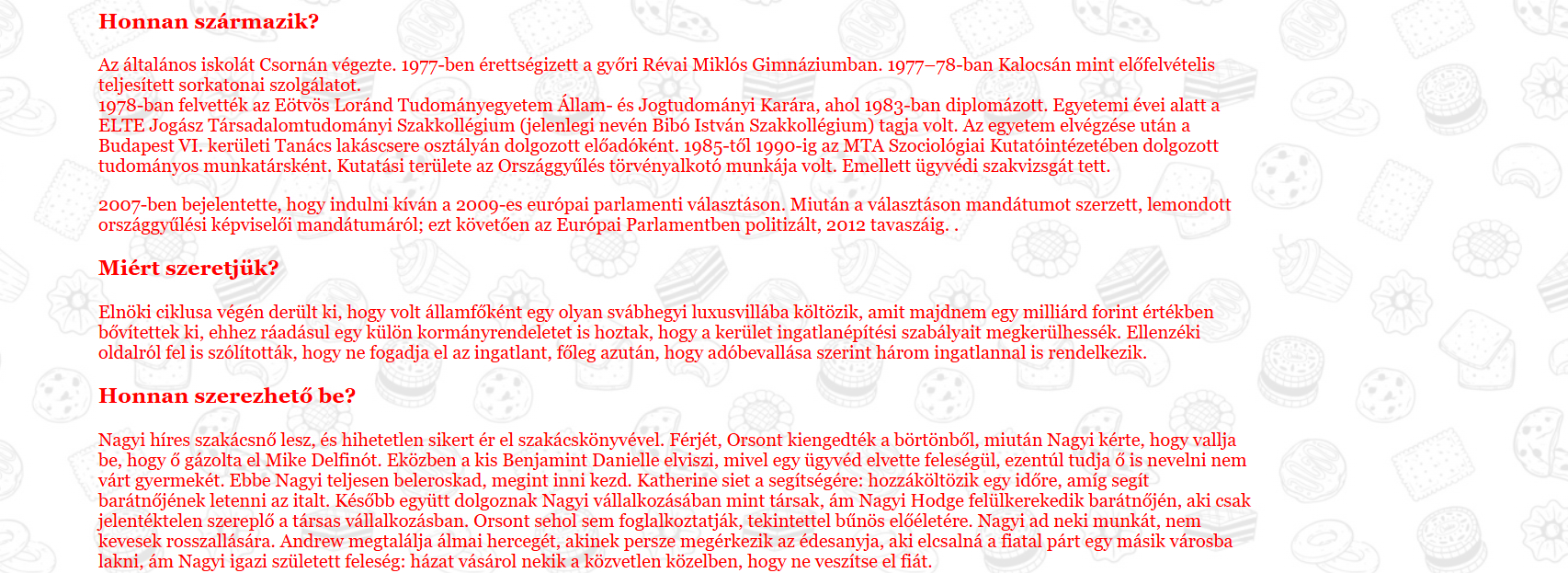
******

**11. ábra**

***1. Feladat***

******

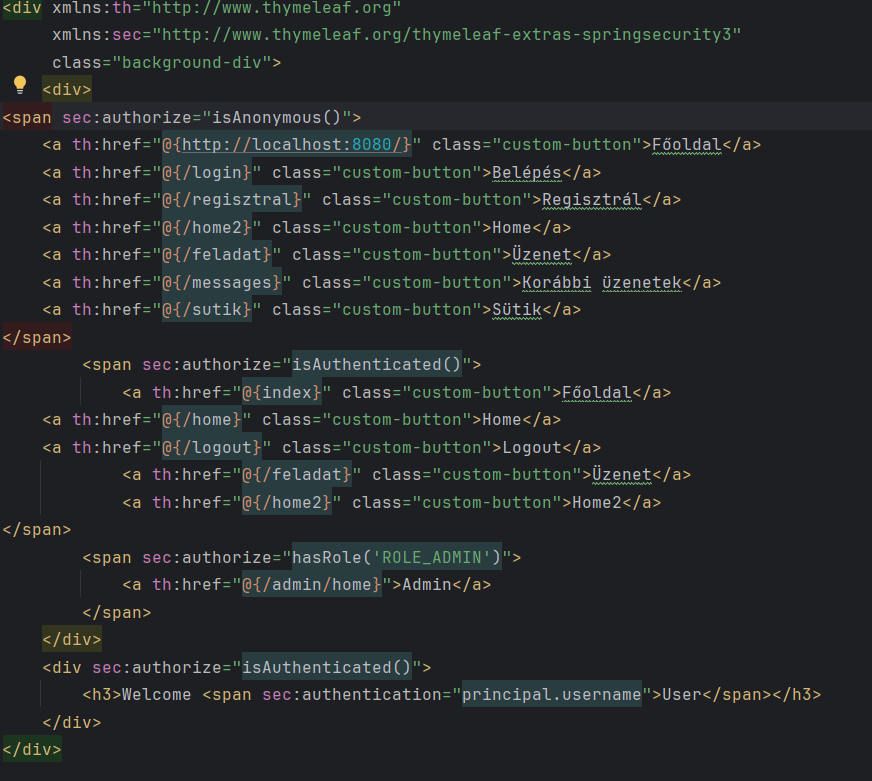
**12. ábra**

******

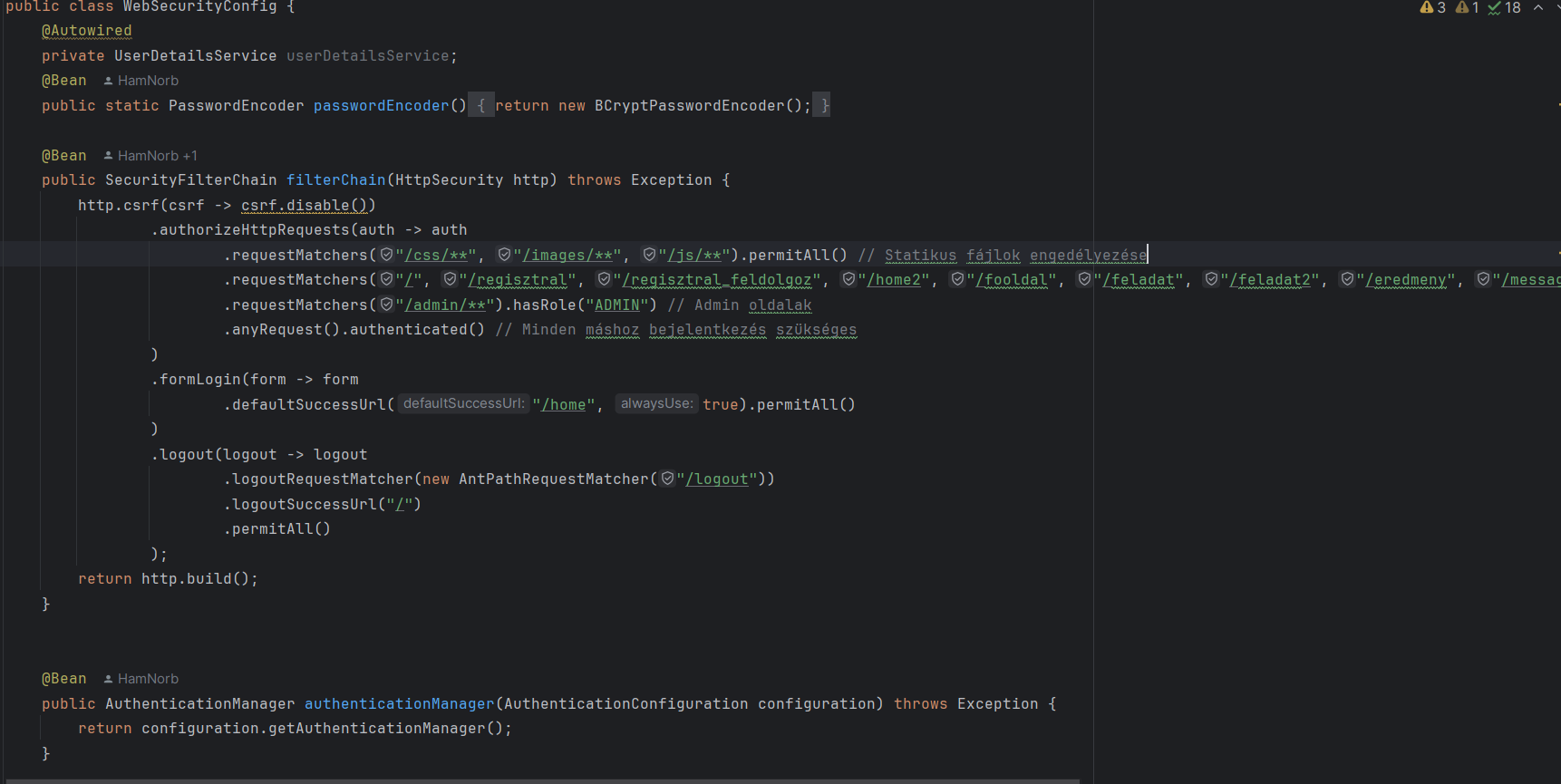
**13. ábra**

***2. és 3. Feladat***

A szerepek elkülönítését spring security és a user adatbázis segítségével valósítottuk meg. A spring security segített a szerepek elkülönítésében, aminek megvalósítása az 14. és 15. ábrán található. A user táblával pedig a userek tárolását, illetve nem tárolását oldottuk meg. Előbbi esetben a vendég szerep van kiosztva.



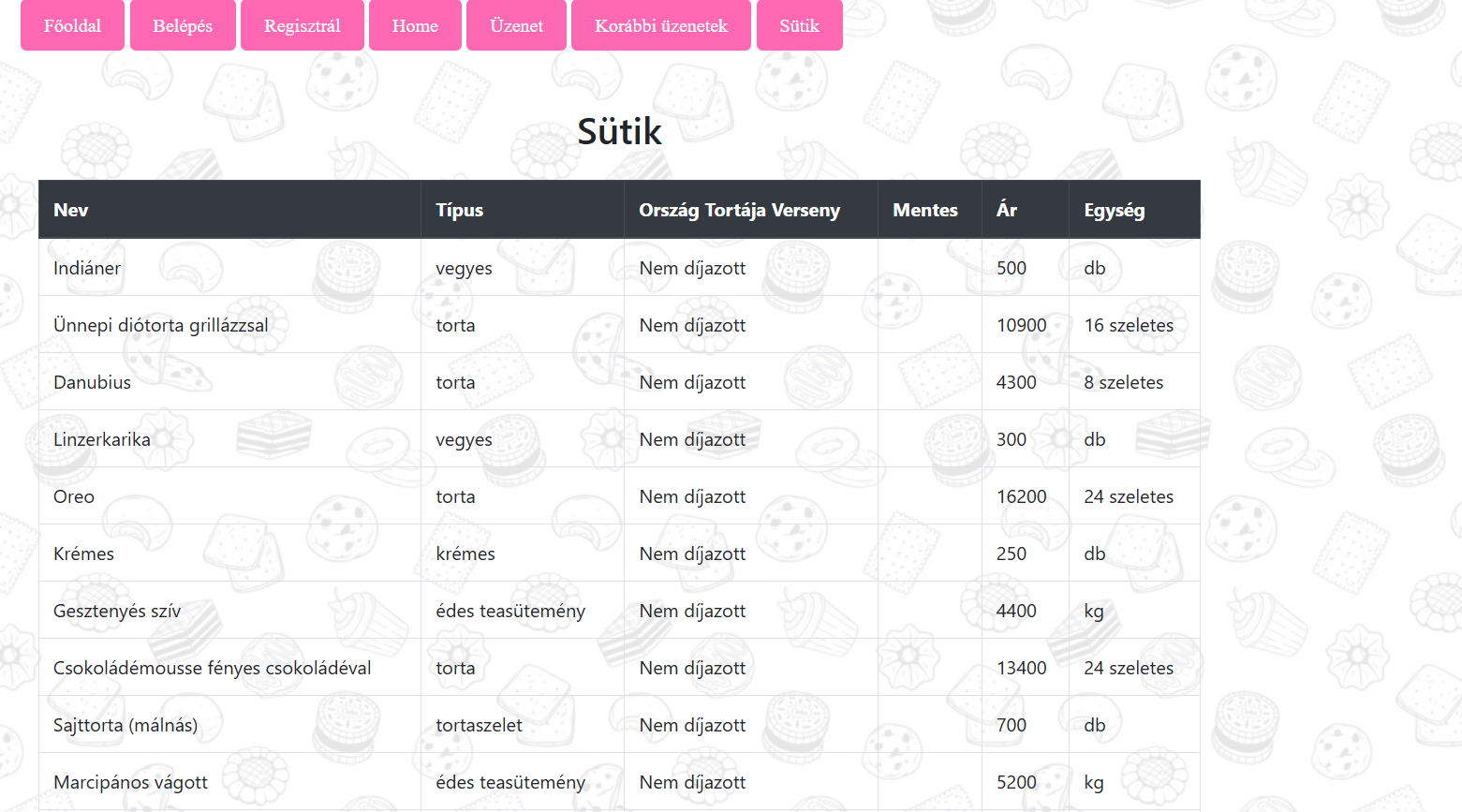
**14. ábra**



**15. ábra**

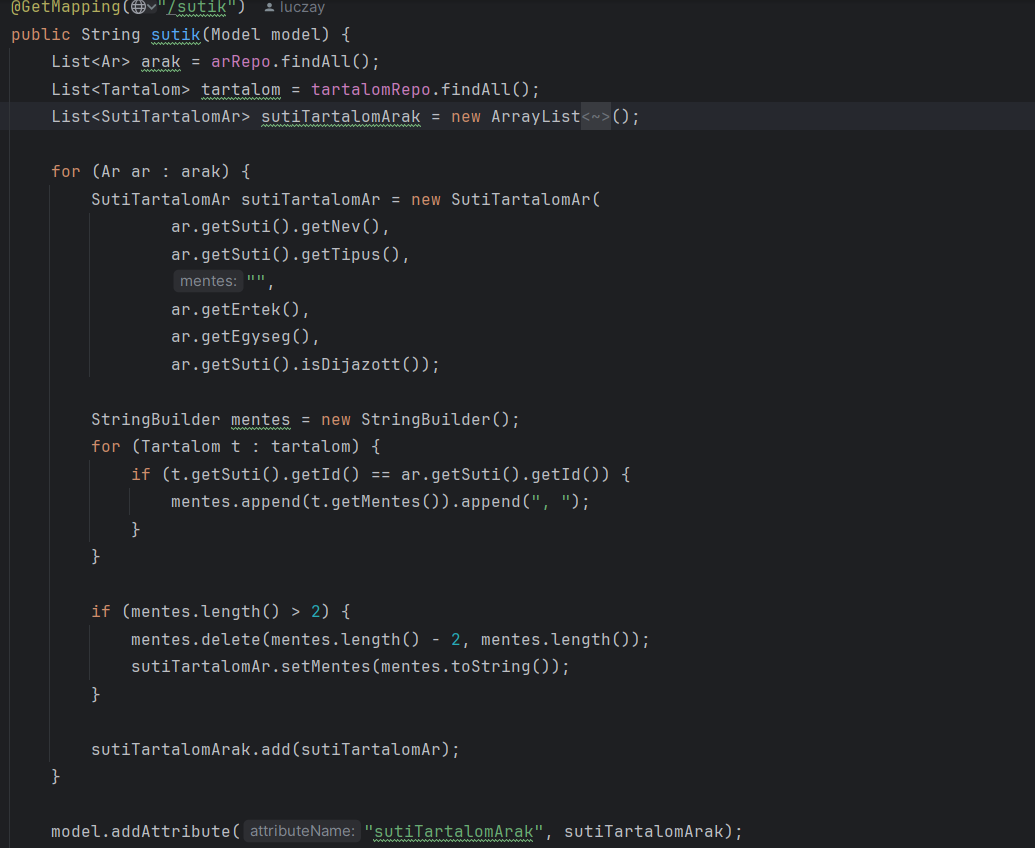
***4. Feladat***

Ezt a feladatot egy sütik nevű oldalon valósítottuk meg, ide írtuk ki a 3 tábla adatait összefűzve, ami az 16. ábrán tekinthető meg.

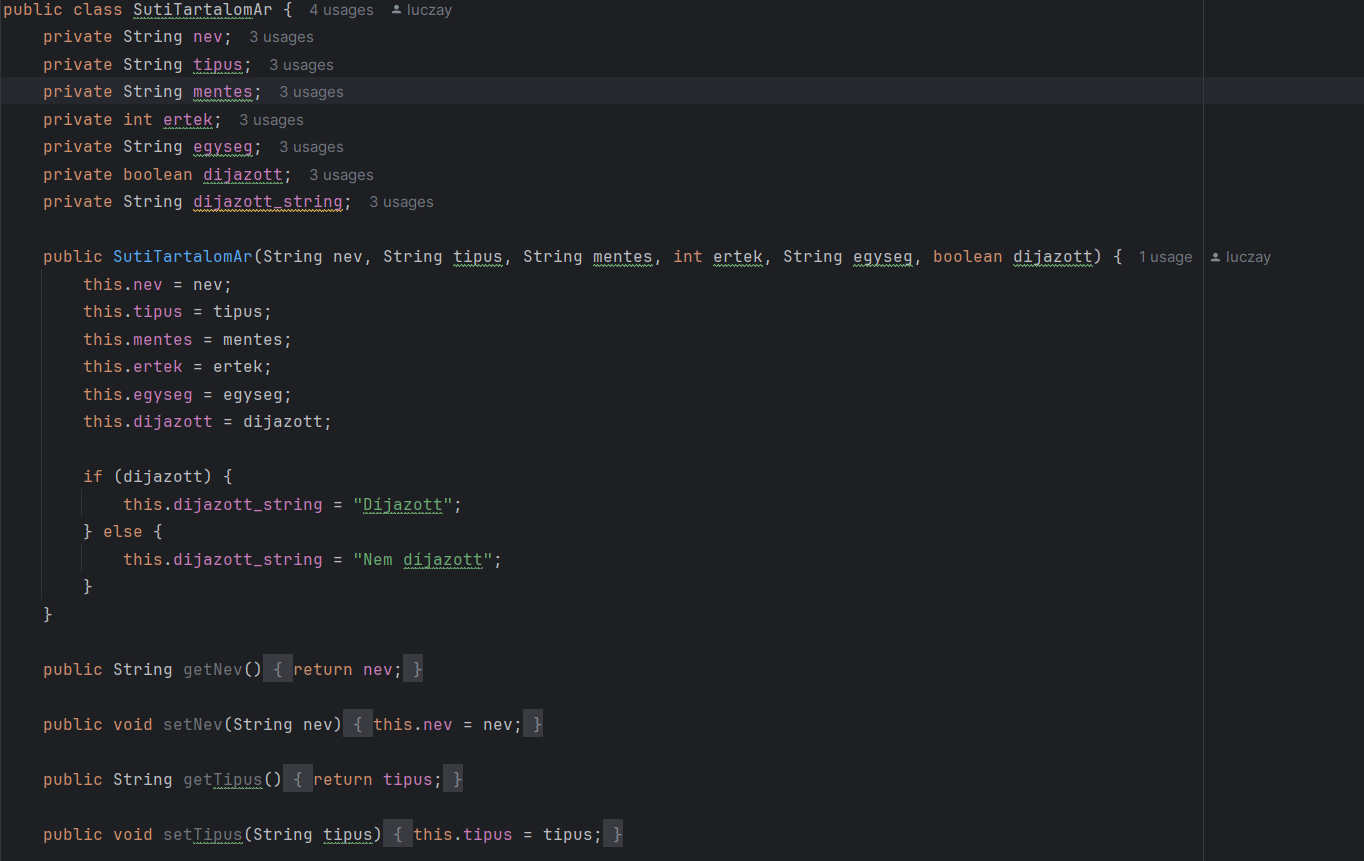


**16. ábra**

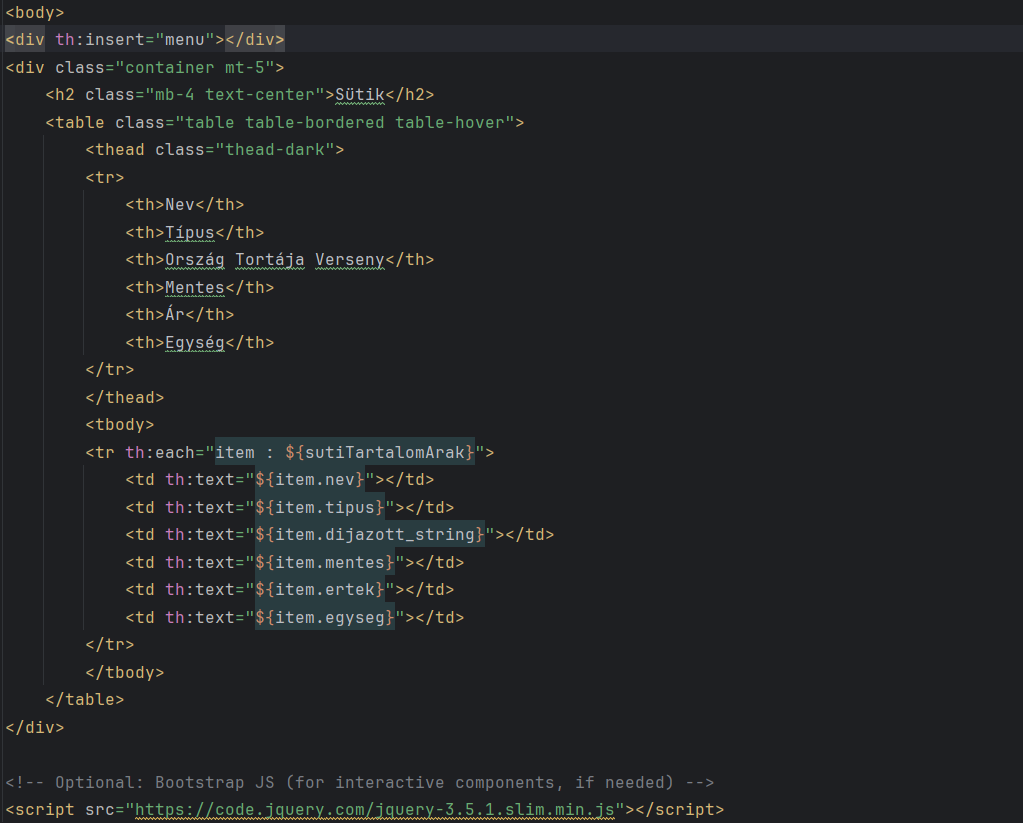
Szerver oldalon ezt nem egy joint használó sql lekérdezéssel valósítottuk meg, hanem felhasználtuk a java JPA, meg maga a java nyelv nyújtotta előnyöket. Elsőnek lekértük az ár és tartalom adatokat az adatbázisból, majd az árakon végig iteráltunk. Minden egyes iterációnál csináltunk egy SutiTartalomAr objektumot aminek a propety értékeit az Ar objektum segítségével töltöttük fel, kivéve a Mentes propertyt. Annak az értékét a lekért tartalom adatokból szedtük ki. Végül a SutiTartalomAr objektumot hozzáadtuk a sutiTartalomArak listához. Ezt a listát pedig tovább adtuk a thymeleaf html oldalunknak a Model objektum segítségével, amin végig iteráltunk és megjelenítettük az eredményt egy táblában. Ennek a megvalósítása az 17., 18., 19. és 20. ábrákon látható.

******

**17. ábra**

******

**18. ábra**

******

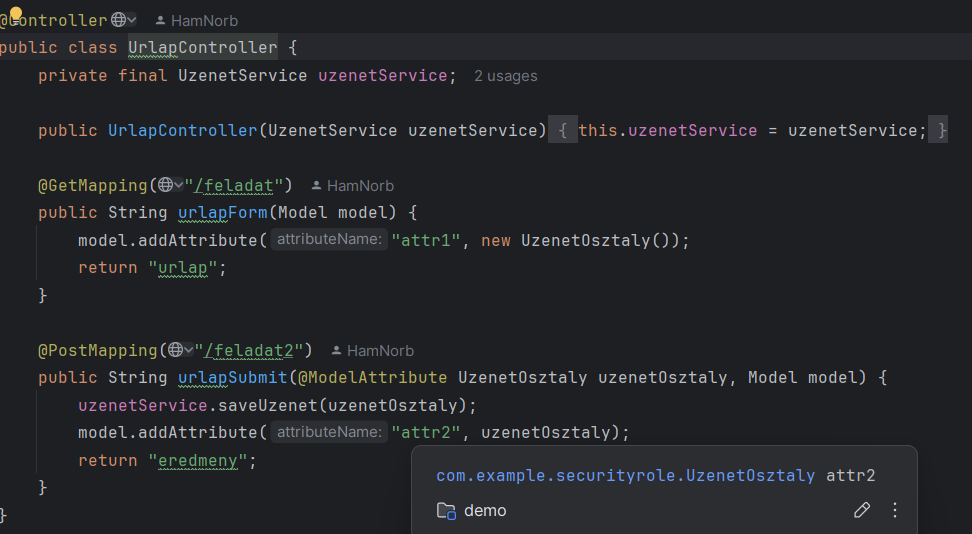
**19. ábra**

***5. Feladat***

A kapcsolat űrlapot az Üzenet nevű oldalon valósítottuk meg, ez látható a . ábrán. Az űrlapban Id és Message mezők vannak, amiket elküldve a szerver az UrlapController osztály urlapSubmit metódusával kezel le, ami a 20. ábrán látható.

******

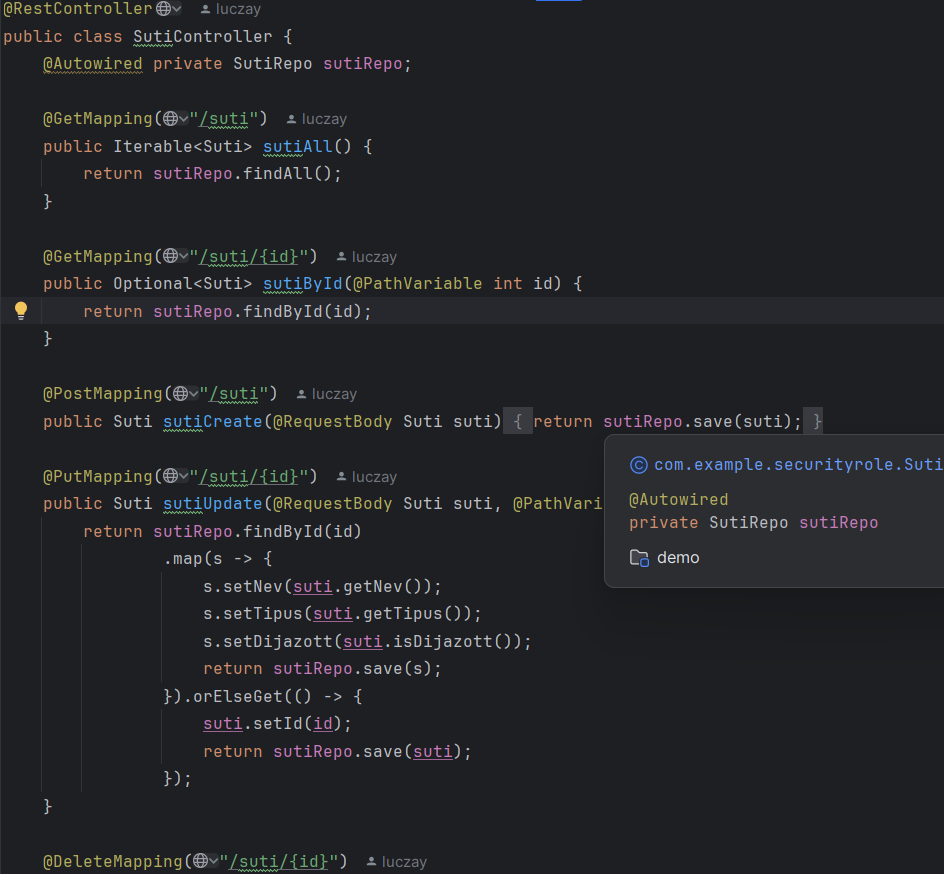
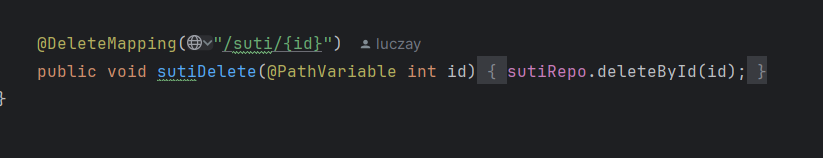
**20. ábra**

******

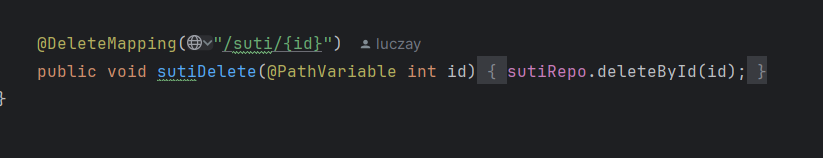
**21. ábra**

***7. Feladat***

A restful apit a suti táblára valósítottuk meg. A megfelelő mappingeket felhasználva a 22. és 23. ábrán látható, hogy a /suti útvonalon érhető el a REST API, és a java JPA könyvtárat használtuk ki a Spring Boot @RestController segítségével a restful tevékenység megvalósításához. Továbbá ehhez a feladathoz a törlés miatt a táblák sql definícióit ki kellett egészíteni ON DELETE CASCADE kóddal a foreign keyeknél, hogy süti törlése esetén törlődjenek az ár és tartalom táblák.



**22. ábra**



**23. ábra**