

RECIPEBOOK

Projektmunka

Debreceni SZC Baross Gábor Technikum, Szakképző Iskola és Kollégium

Készítette:
Böjti István
Kiss-Ruprecht Dóra
Zavadovics Ákos

Tartalomjegyzék

| | |
|--|----|
| 1. Bevezetés | 2 |
| 1.1. Téma ismertetése | 2 |
| 1.2. Témaválasztás indoklása | 2 |
| 2. Fejlesztői dokumentáció | 3 |
| 2.1. Technológiák bemutatása | 3 |
| 2.1.1. Kliensoldal | 3 |
| 2.1.2. Szerveroldal | 4 |
| 2.2. Adatszerkezet, típusok, változók | 4 |
| 2.3. Program algoritmusai | 5 |
| 2.4. Tesztdokumentáció | 6 |
| 2.5. Fejlesztési lehetőségek | 11 |
| 3. Felhasználói dokumentáció | 13 |
| 3.1. Program célja | 13 |
| 3.2. Hardver és szoftverigény | 13 |
| 3.3. Telepítés és indítás | 13 |
| 3.4. Részletes bemutatás | 13 |
| 3.5. Hibajelzések | 18 |
| 3.6. A program készítőjének elérhetősége | 18 |
| 4. Összefoglalás | 19 |
| 5. Irodalomjegyzék | 20 |

1. Bevezetés

1.1. Téma ismertetése

Az dolgozat témája egy recept alkalmazás, amivel recepteket lehet létrehozni, rögzíteni az azokhoz szükséges hozzávalókat és az elkészítés módját.

1.2. Témaválasztás indoklása

A téma a csapattagok érdeklődése alapján került kiválasztásra. Mindenképpen olyan témát kerestünk, amely mindenki számára érdekes lehet. Közös pont, hogy mindhárman érdeklődünk a főzés iránt, így egyértelműen ebben a témakörben szerettünk volna egy olyan alkalmazást fejleszteni, amely mindenkinek hasznos lehet, akik szoktak főzni, szeretnék másokkal is megosztani a receptjeiket, illetve szívesen próbálnak ki mások receptjeit.. Így esett a választásunk egy recept alkalmazásra, amely segítséget nyújthat a felhasználónak az ételkészítés során. A korábban kipróbált és már bevált recepteket el tudják menteni, majd szükség esetén elővenni.

Továbbá az alkalmazás egy olyan funkcionális területet céloz meg, amely széles körben elterjedt az interneten. Egy ilyen alkalmazás bemutatása a választott technológia sokoldalúságára és alkalmazhatóságára is rávilágíthat. Így tehát, azért választottuk ezt a témát és készítettük el az alkalmazást, mert a választott technológia ideális eszközöket és lehetőségeket kínál az adott probléma megoldására.

2. Fejlesztői dokumentáció

2.1. Technológiák bemutatása

2.1.1. Kliensoldal

Kliensoldalon használt technológiák esetén a Google által fejlesztett Angular keretrendszert választottuk. Az Angular az egyoldalas webalkalmazás működési mechanizmusát használja, így az alkalmazásunk nem új oldalakat tölt be, hanem a szervertől lekért adatok alapján dinamikusan, komponensek segítségével változtatja a felhasználói felületet. Ezzel a megoldással jelentősen gördülékenyebbé lehet tenni a felhasználói interakciókat, aminek következtében a felhasználói élmény és elégedettség is növekszik. Kiegészíti a HTML szintaxisát, lehetővé téve többek között a dinamikus tartalom megjelenítést. Automatikusan frissíti a kigenerált DOM-ot, amikor egy komponens állapota megváltozik. A legegyszerűbb példa erre a tulajdonságra a dinamikus szöveg behelyettesítés. Ezenfelül támogatja a tulajdonságkötést (property binding), eseménykötést (event binding), illetve a kétirányú adatkötést (two-way data binding), amiknek köszönhetően kommunikálhatunk a sablonunk (HTML) és a logikát tartalmazó TypeScript osztály között.

Az Angular jól integrálható számos külső könyvtárakkal is. A leggyakrabban használt ezek közül az RxJs Javascript könyvtár amivel az aszinkron módon érkező adatokat tudjuk kezelni, illetve az NgRx amivel az állapotkezelést tudjuk magasabb szintekre emelni.

Összességében elmondható, hogy az Angular fő előnye a nagyobb skálázható projektek tervezésében és megvalósításában rejlik, de kisebbek esetén is effektíven felhasználható és hasznosítható.

A projekt frontend oldali fejlesztéséhez a Microsoft által ingyenesen elérhető Visual Studio Code forráskód szerkesztő programot használtuk. Elérhető Windows, macOS és Linux operációs rendszereken. Beépített támogatást biztosít JavaScript, TypeScript és Node.js-ben írt alkalmazások számára, gazdag bővítmény készlete által más nyelveket (C++, C#, Java, Python) is könnyen integrálhatóvá tesz. Funkciói közé sorolható, a szintaxis kiemelés, automatikus behúzás, kódkiegészítés, kódrészletek beillesztése, kódok közötti navigáció és hibakeresés.

2.1.2. Szerveroldal

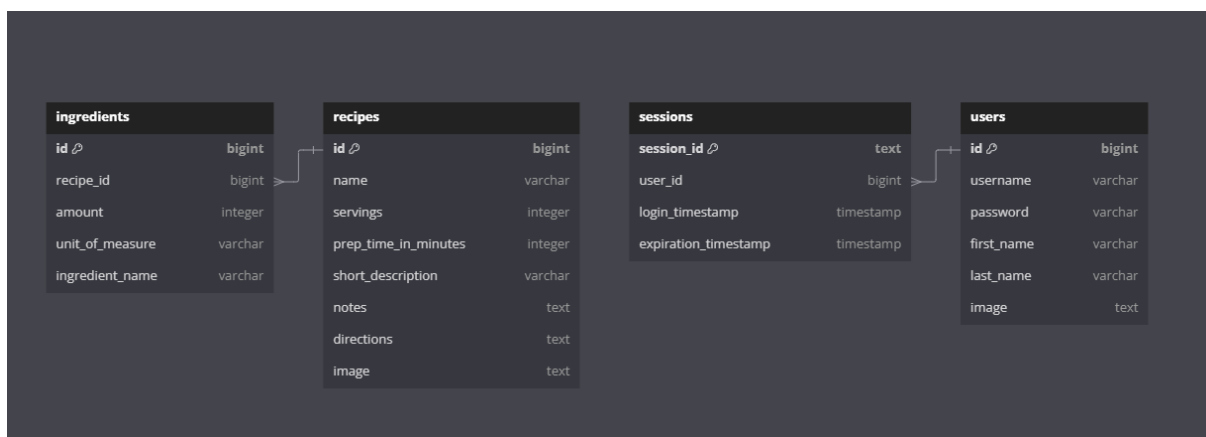
A szerveroldal fejlesztéséhez Java 17 lett választva, hiszen ez a verzió hosszú támogatást élvez még évekig, így ideális élő rendszerek fejlesztéséhez. A Java-n belül a Spring Boot keretrendszer lett kiválasztva, mivel a Spring Boot gyors és könnyű alkalmazásfejlesztést tesz lehetővé a konvenciók és beállítások által. Ezen túlmenően a Java platformfüggetlen, így az alkalmazásokat bármely környezetben könnyen futtathatjuk.

Az alkalmazás egy PostgreSQL 14.3 adatbázishoz lett fejlesztve, de az adatok és a táblák annyira generikusak, hogy bármilyen SQL adatbázist lehetne hozzá használni.

Fejlesztéshez használt felhásználói környezet az Eclipse és pgAdmin 4.

2.2. Adatszerkezet, típusok, változók

Az alkalmazás csak alapvető típusokat használ a változókhoz. Általánosságban az ID oszlopok Long-ot használnak, egyéb numerikus változók Integert, időbélyegek LocalDateTime-ot, ami adatbázisban timestampnek felel meg, illetve még String-et használ az alkalmazás, ami az adatbázis oldalon kétféleképpen jelenhet meg. Az egyik a varchar, a másik a text. A text mezőt általában hosszabb szövegekre használjuk, vagy jelen esetben a képek ebben vannak letárolva Base64 kódolással.



Az fenti ábra az egyszerű adatbázis modellt mutatja. Létezik egy Users tábla, ami a bejelentkezési adatokat tartalmazza, a jelszót MD5-ben tárolva, vezetéknév és keresztnév, illetve egy profilképet Base64-ben kódolva.

Egy Session tábla, amelybe bejelentkezés után keletkezik egy sor, aminek a session_id oszlopa egy GUID, vagyis egy Globálisan Egyedi Azonosító. Ez azért hasznos session azonosítónak, mert soha nem fog keletkezni kettő ugyan olyan session. Illetve ez a tábla még tartalmazza a bejelentkezés idejét, és a lejárat idejét, ami a bejelentkezéstől számítva 8 órán keresztül érvényes.

A Recipes táblába található maga a receptek, a nevével, leírásával, képével, elkészítése stb. mezővel ellátva.

Végül, de nem utolsó sorban az Ingredients tábla, ami a receptnek a hozzávalóit tartalmazza. Tartalmaz egy recept azonosítót, mennyiséget, mértékegységet, és a hozzávaló nevét.

2.3. Program algoritmusai

2.3.1. A regisztráció algoritmusai

A regisztráció során felhasznált algoritmusok közül az első megnézi, hogy a megadott username üres-e, amennyiben igen, akkor visszaadunk egy hibát a programból, hiszen üres felhasználónévvel nem lehet regisztrálni. Az alábbi kép ezt mutatja:

```
if(saveUserDto.getUsername() == null || saveUserDto.getUsername().isBlank())  
    throw new NullPointerException("Username is empty!");
```

A jelszónál ugyan ezt az ellenőrzést megcsináljuk.

A következő ellenőrzésnél megnézzük, hogy a felhasználónév létezik-e már.

```
if(userExists(saveUserDto.getUsername()))  
    return ResponseEntity.ok().build();
```

Ha létezik akkor csak okéval visszatérünk, biztonsági okokból nem jelezzük hibaként hogy már létezik.

2.3.2. Bejelentkezés algoritmusai

A bejelentkezés során kevesebb ellenőrzésről beszélünk, hiszen, ha regisztrációnál nem engedjük az üres felhasználónevet és jelszót, akkor bejelentkezésnél sem fog ilyen User-t találni.

Ettől függetlenül megtalálható benne a jelszó MD5 hashelése, amit korábban már említettünk, illetve a Sessionnek egy GUID vagy UUID generálás. Ennek a generálását szerencsére nem magunknak kell implementálni, hanem a Javának vannak belső függvényei/metódusai ehhez.

```
String sessionId = UUID.randomUUID().toString();  
String md5Password = DigestUtils.md5Hex(loginDto.getPassword()).toUpperCase();
```

Ezután az elküldött jelszót MD5-ben hasheljük az alábbi módon:

```
String md5Password = DigestUtils.md5Hex(saveUserDto.getPassword()).toUpperCase();
```

Szerencsére, az egész algoritmust nem magunknak kell megírni, a Javához rengeteg előre megírt hasznos dependencyk léteznek, ilyen például az Apache Commons Codec. Ezt használtuk MD5 hashelésre. Valójában, az MD5-t nem érdemes használni, nem elég biztonságos, de a program egyszerű bemutatása érdekében most megfelelőnek találtam.

2.3.3. Profilkép és recept kép algoritmus

A képeket már fentebb említett Base64-ben kódoljuk és mentjük adatbázisba, ez egyszerűbben Stringként kezelhető az alkalmazás minden rétegében.

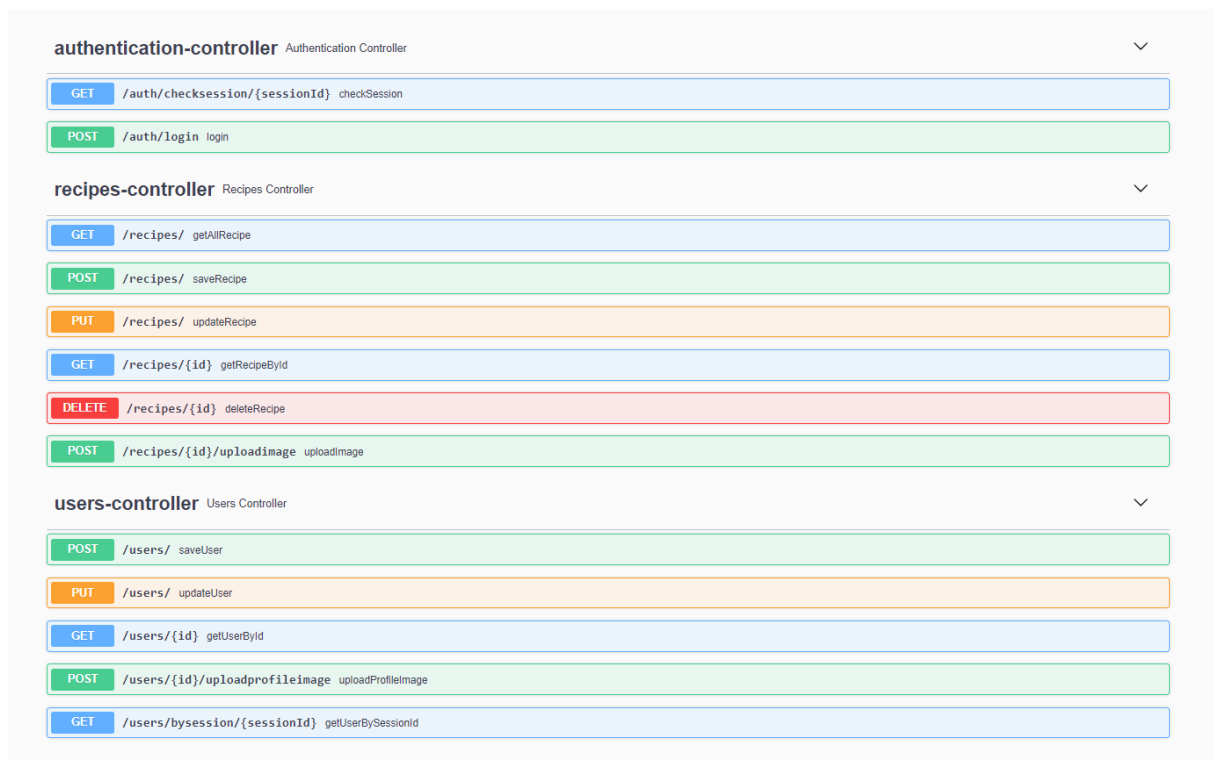
```
String imageEncoded = Base64.getEncoder().encodeToString(image);
```

2.4. Tesztdokumentáció

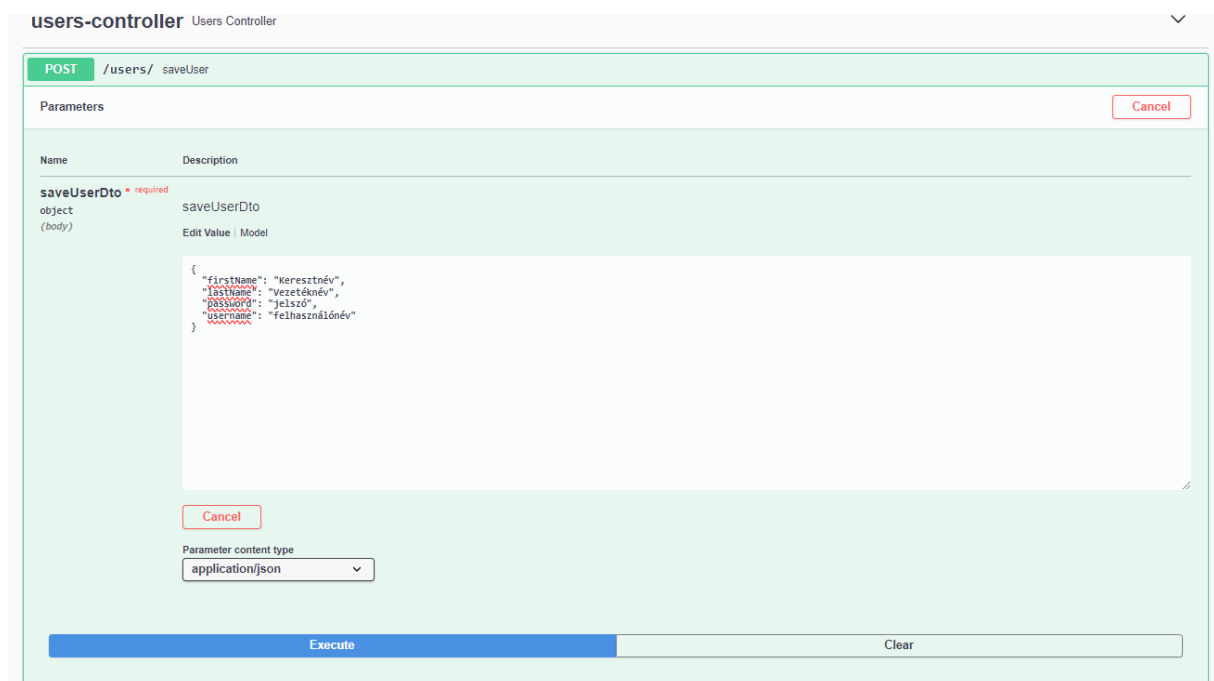
Swagger használata

Egyik szerveroldali tesztelésre a lehetőség a Swagger használata. A Swagger lehetővé teszi a fejlesztők és az API-felhasználók számára, hogy könnyen megértsék az API által kínált funkcionalitást, ezáltal a Swagger segítségével könnyedén ellenőrizhetjük, hogy az API megfelel-e a specifikációnak, és helyesen működik-e az elvárt módon. Lokálisan indítva az alábbi elérhetőségen található a swagger felülete: <http://localhost:8080/swagger-ui/index.html>

A swagger az alábbi módon van felépítve:



Az alábbi képen látható egy példa hívás a regisztrációra:



A szerver válasza a regisztrációra:

| Server response | |
|-----------------|--|
| Code | Details |
| 200 | <div>Response headers</div> <div>connection: keep-alive content-length: 0 date: Tue30 Apr 2024 13:07:57 GMT keep-alive: timeout=60 vary: OriginAccess-Control-Request-MethodAccess-Control-Request-Headers</div> |
| Responses | |
| Code | Description |
| 200 | OK |
| 201 | Created |
| 401 | Unauthorized |
| 403 | Forbidden |

A szerver 200-as kódot adott vissza, ami OK-ot jelent, vagyis a mentés sikeres volt. Kipróbálhatjuk, hogy mi fog történni, amennyiben nem adunk meg jelszót.

object
(body)

saveUserDto

Edit Value | Model

```
{  "firstName": "Keresztnev",  "lastName": "Vezeteknev",  "password": "",  "username": "felhasználónev?"}
```

Cancel

Parameter content type
application/json

Execute

Clear

Responses

Response content type */*

Curl

```
curl -X POST "http://localhost:8080/users/" -H "accept: */*" -H "Content-Type: application/json" -d "{ \"firstName\": \"Keresztnev\", \"lastName\": \"Vezeteknev\", \"password\": \"\", \"username\": \"felhasználónev?\"}"
```

Request URL

http://localhost:8080/users/

Server response

| Code | Details |
|------|--|
| 500 | <div>Error:</div> <div>Response headers</div> <div>connection: close content-length: 0 date: Tue30 Apr 2024 13:10:26 GMT vary: OriginAccess-Control-Request-MethodAccess-Control-Request-Headers</div> |

Responses

| Code | Description |
|------|-------------|
|------|-------------|

Láthatjuk ebből, hogy a szerver válasza 500, vagyis hibára futottunk.

8

Swagger használata mellett lehetőség van más típusú tesztek, például mock tesztek bevezetésére is az alkalmazásfejlesztés folyamatában. A mock tesztek lényege, hogy szimulálják a valós környezetet, és segítségükkel izoláltan lehet tesztelni az alkalmazás egyes részeit anélkül, hogy valós függőségekkel vagy külső rendszerekkel kellene interakcióba lépni. Nézzünk egy példát.

```
@Test
public void testLogin() throws Exception {
    LoginDto loginDto = new LoginDto();
    loginDto.setUsername("testUser");
    loginDto.setPassword("testPassword");

    Sessions session = new Sessions();
    session.setSessionId("sessionId");
    session.setUserId(1L);
    session.setLoginTimestamp(LocalDate.now());
    session.setExpirationTimestamp(LocalDate.now().plusHours(8));

    when(authenticationService.login(Mockito.any(LoginDto.class)))
        .thenReturn(new ResponseEntity<>(session, HttpStatus.OK));

    mvc.perform(post("/auth/login")
        .contentType(MediaType.APPLICATION_JSON)
        .content(objectMapper.writeValueAsString(loginDto)))
        .andExpect(status().isOk())
        .andExpect(jsonPath("$.sessionId").value("sessionId"));
}
```

Ez a teszt egy "login" végpontot tesztel az alkalmazásban. A teszt lépései a következők:

- Létrehoz egy LoginDto objektumot, ami egy felhasználó bejelentkezési adatokat tartalmaz.
- Létrehoz egy Sessions objektumot, ami egy bejelentkezési munkamenetet reprezentál, például egy felhasználóhoz tartozó munkamenetet.
- Beállítja a munkamenet ID-ját, a felhasználó azonosítóját, bejelentkezési időpontját és lejáratási időpontját.
- Beállítja a Mockito segítségével, hogy a authenticationService fiktív választ adjon vissza, amikor a login metódust hívják.
- Elküld egy HTTP POST kérést a /auth/login végpontra.
- Várakozik egy válasza, aminek státusza 200 OK, és az sessionId értékének a helyességét ellenőrzi a válasz JSON tartalmában.

Összességében ez a teszt ellenőrzi, hogy a bejelentkezési végpont megfelelően működik-e a várt munkamenet objektumot visszaadva.

2.5. Fejlesztési lehetőségek

Az alkalmazást fejlesztésére számos lehetőség áll rendelkezésre, többek között az alábbiak:

- Kategóriák és címkék: Hozzáadható lenne egy kategóriák és címkék rendszer, amely segítségével a felhasználók csoportosíthatják és könnyebben megtalálhatják a recepteket. Például lehetne kategóriákban rendezni a recepteket (pl. reggeli, ebéd, vacsora) és címkéket rendelni hozzájuk (pl. vegetáriánus, gyorsan elkészíthető stb.).
- Keresés és szűrés: Implementálható lenne egy keresési funkció, amely lehetővé teszi a felhasználók számára a receptek közötti keresést kulcsszavak alapján. Emellett szűrési lehetőségeket is hozzá lehetne adni, például alapanyagok vagy elkészítési idő alapján.
- Népszerűség és értékelés: Hozzáadható lenne egy népszerűségi és értékelési rendszer, amely lehetővé teszi a felhasználók számára, hogy értékeljék és megoszthassák a recepteket, valamint böngészhetnek a legnépszerűbb receptek között.
- Social sharing: Lehetőség biztosítása a receptek megosztására közösségi média platformokon vagy e-mailen keresztül, ami növelné az alkalmazás népszerűségét és a felhasználók közti interakciót.
- Hozzászólások és vélemények: Funkció hozzáadása, amely lehetővé teszi a felhasználók számára, hogy hozzászóljanak a receptekhez és megosszák a saját tapasztalataikat vagy változtatásokat javasoljanak.
- Nyelvek és lokalizáció: Támogatás hozzáadása több nyelvhez és lokalizációhoz, ami lehetővé teszi az alkalmazás szélesebb körű elterjesztését.
- Étrend-specifikus funkciók: Funkciók hozzáadása olyan speciális étrendek támogatásához, mint például vegetáriánus, vegán, gluténmentes, paleo stb.
- Étrend- és kalória-nyomon követés: Funkció hozzáadása, amely lehetővé teszi a felhasználók számára, hogy nyomon kövessék az elfogyasztott ételek kalória- és tápanyagtartalmát, és figyelemmel kísérjék étrendjüket és egészségügyi célokat.
- Videók: Lehetőség biztosítása a felhasználóknak, hogy video recepteket nézzenek, amelyek bemutatják az étel elkészítését lépésről lépésre, segítve ezzel a vizuális tanulást és megértést.

-
- Allergén és táplálkozási információk: Funkció hozzáadása, amely lehetővé teszi a felhasználók számára, hogy az ételreceptekben azonosítsák és címkézzék az allergéneket és egyéb fontos táplálkozási információkat, segítve ezzel az ételintoleranciák és diéták figyelembevételét.
 - Vásárlási ajánlások és linkek: Funkció hozzáadása, amely lehetővé teszi a felhasználók számára, hogy a receptek mellé kapjanak vásárlási ajánlásokat és linkeket az alapanyagok beszerzéséhez, például online áruházakhoz vagy helyi üzletekhez.
 - Étkezési tervek és heti menük: Lehetőség biztosítása a felhasználóknak, hogy előre megtervezzék a heti étkezéseiket és létrehozzanak étkezési terveket, amelyek segítik őket az egészséges és változatos étrend összeállításában.

3. Felhasználói dokumentáció

3.1. Program célja

A felhasználó a megadott felületen képes legyen regisztrálni, majd ezt követően bejelentkezni. A bejelentkezést követően a főoldalra történik a navigáció, ahol láthatja az eddig felvett receptjeit, amik között keresni tud, illetve újakat tud felvenni a megadott felugró ablak segítségével. Az egy receptet tartalmazó részekben funkció gombok segítségével megtekinthetjük azok részleteit, vagy törölni tudjuk őket. Egy recept részletesebb leírása egy új oldalon jelenik meg, itt már lehetőséget adva annak módosítására is. Az egész alkalmazás során látható egy oldalmenü ami tartalmazza a felhasználó nevét és profilképét illetve a receptek menüpontot és a kijelentkezés lehetőségét. A profilképén látható fogaskerék ikon megnyomása után megjelenő felugró ablakban módosíthatja a nevét, jelszavát, illetve profilképét.

3.2. Hardver és szoftverigény

A webalkalmazás elérhető vezetékes vagy vezeték nélküli internetkapcsolattal rendelkező számítógép vagy okos eszköz (telefon, táblagép) által. Az alkalmazás automatikusan alkalmazkodik a felhasználó eszközének képernyő méretéhez, így az egyéb beállításokat nem igényel.

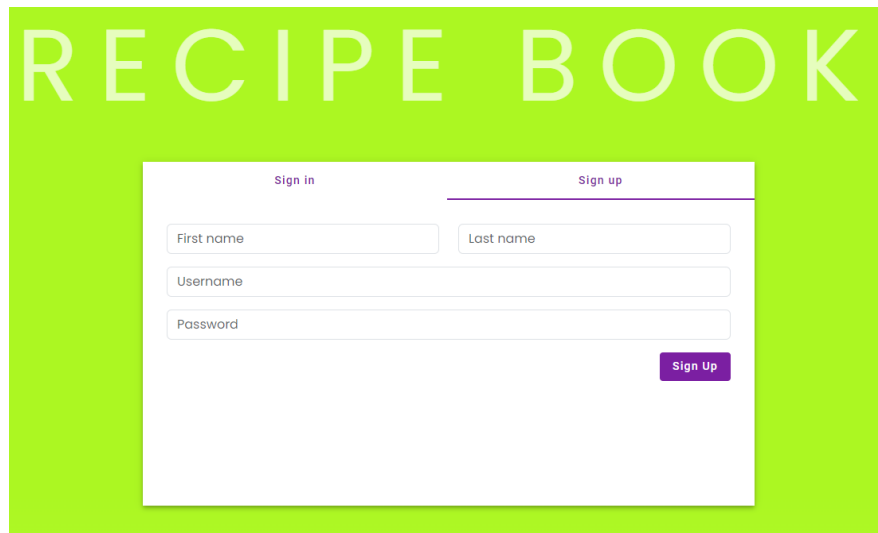
3.3. Telepítés és indítás

Mivel egy webalkalmazásról van szó, így nincs szükség telepítésre, egyszerűen a weboldal URL címén lehet elérni az alkalmazást.

3.4. Részletes bemutatás

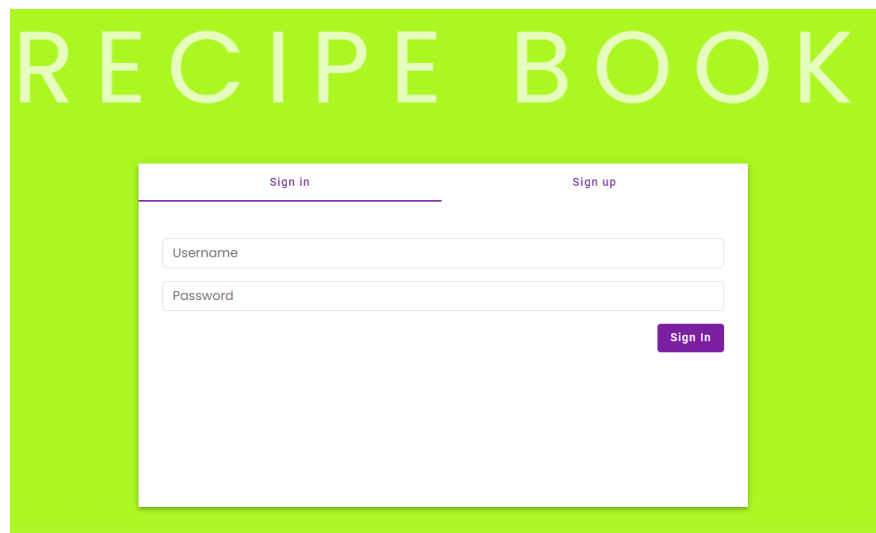
Induláskor, az URL címet beírva az adott böngésző címsorjába, az alkalmazás az autentikációs felületre navigál, ahol a felhasználó választhat a bejelentkezés és regisztráció között. A

regisztráció néhány alapadat megadásával történik, mint keresztnév, vezetéknév, felhasználónév és jelszó.



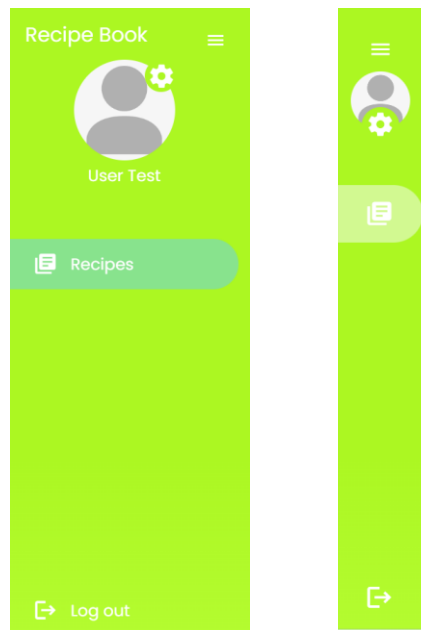
The image shows a web form titled "RECIPE BOOK" in large, light green letters on a dark green background. The form itself is white and contains two tabs: "Sign in" and "Sign up". The "Sign up" tab is selected, indicated by a purple underline. Below the tabs, there are four input fields: "First name", "Last name", "Username", and "Password". A purple "Sign Up" button is located at the bottom right of the form.

Ezt követően visszatérhetünk a bejelentkezés fülhöz ahol felhasználónév és jelszó páros segítségével bejelentkezhetünk, ilyenkor a főoldalra leszünk átirányítva.



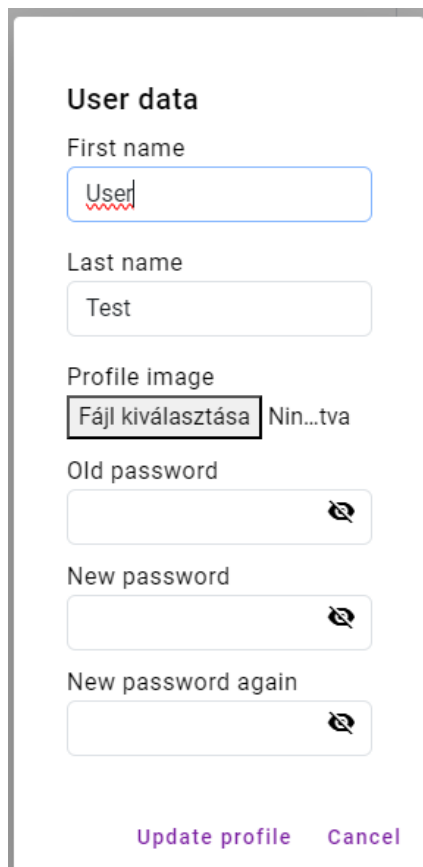
The image shows the same "RECIPE BOOK" web form, but now the "Sign in" tab is selected, indicated by a purple underline. The "Sign up" tab is now greyed out. The form contains two input fields: "Username" and "Password". A purple "Sign In" button is located at the bottom right of the form.

A bejelentkezett felhasználó számára megjelenik egy oldal sáv amin látható a felhasználó profilképe, neve, receptek menüpont és a kijelentkezés lehetősége. Az oldalsáv a fent található hamburger ikon segítségével összecsuksukható, aminek használata kisebb képernyő méretek esetén erősen ajánlott. Amíg a felhasználó nem állít be profilképet addig egy semleges felhasználói kép látható. A kijelentkezés lehetőséget választva az autentikációs felületre



leszünk vissza navigálva.

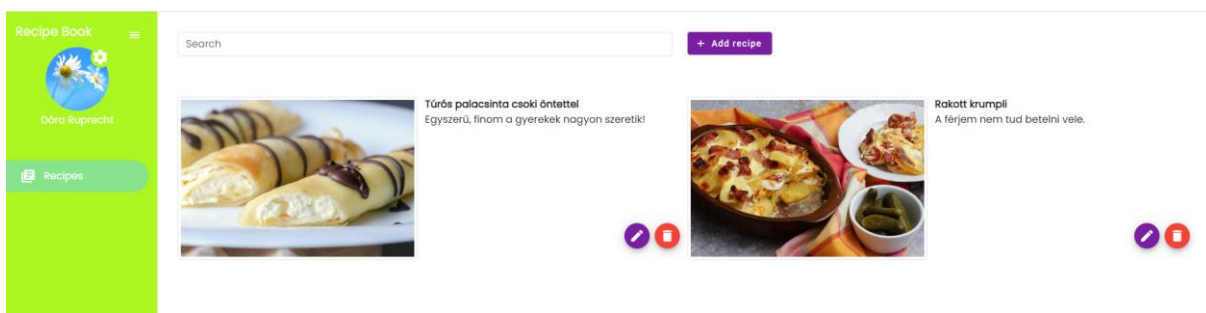
A felhasználói adatok módosítására a profilképen jobb felső sarkában található fogaskerék ikon szolgál. Ezt megnyomva megjelenik egy felugró ablak. Itt nem szükséges az összes mezőt kitölteni, amennyiben nem kívánjuk azokat megváltoztatni.



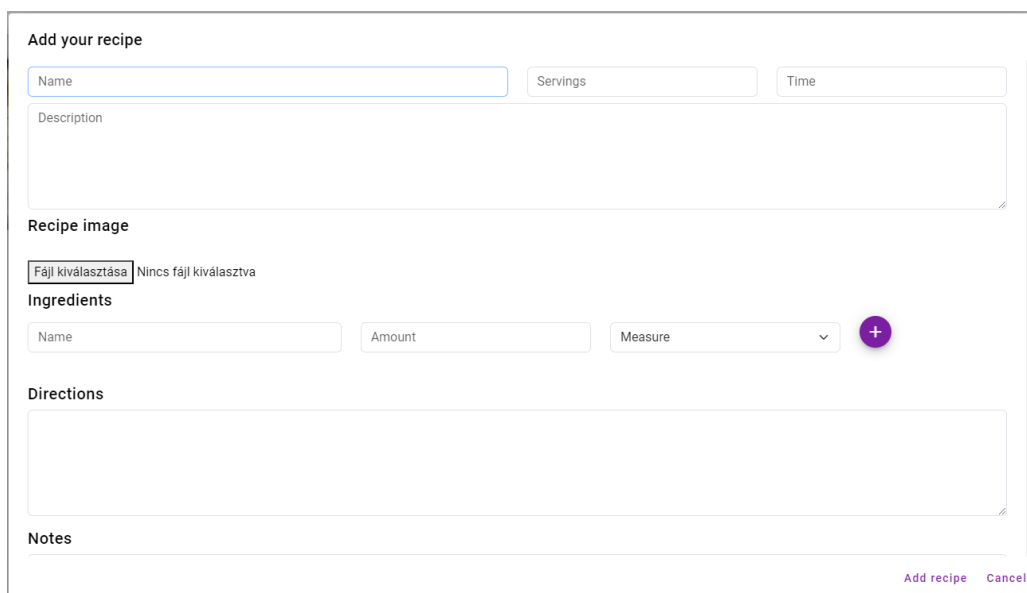
The image shows a modal form titled "User data" for editing user information. It contains the following fields and controls:

- First name:** A text input field containing "User".
- Last name:** A text input field containing "Test".
- Profile image:** A button labeled "Fájl kiválasztása" (Select file) next to the text "Nin...tva".
- Old password:** A password input field with a toggle icon on the right.
- New password:** A password input field with a toggle icon on the right.
- New password again:** A password input field with a toggle icon on the right.
- Buttons:** At the bottom, there are two buttons: "Update profile" and "Cancel".

A főoldalon jelennek meg receptjeink. Az egyes szegmensek a recept nevét képét és rövid leírását tartalmazzák, emellett a ceruza ikonnal jelzett gombbal részletesebb megtekinthetjük, a mellette lévő kuka ikonnal ellátott gomb pedig a törlésre szolgál. A keresősáv segítségével a nevekre tudunk keresni (például: megadott szó: túró, kihozza az összes olyan receptet aminek nevében túró szerepel).



Receptet a keresősáv melletti gombbal vehetünk fel, ilyenkor egy felugró ablakban megnyíló űrlapot kitöltve adhatjuk meg a leírást.



Az űrlapban található mezők kitöltését követően a jobb alsó sarokban tudjuk elmenteni a receptünket.

A részletesebb megjelenés egy új oldalra vezet, ahol a már eddig látott információkat kiegészítve láthatjuk a recept elkészítési idejét, adagszám, hozzávalók, utasítások, utasításokhoz tartozó tippek, tanácsokat.



Túrós palacsinta csoki öntettel

35 min 16 servings

INGREDIENTS

- 20 dkg liszt
- 2 Tojás
- 3 dl Tej
- 2 dl Szénsavas ásványvíz
- 1 g Só
- 0 dl Olaj

NOTES

Figyeljünk oda a töltelék arányokra! Nehogy túlcsozduljon!

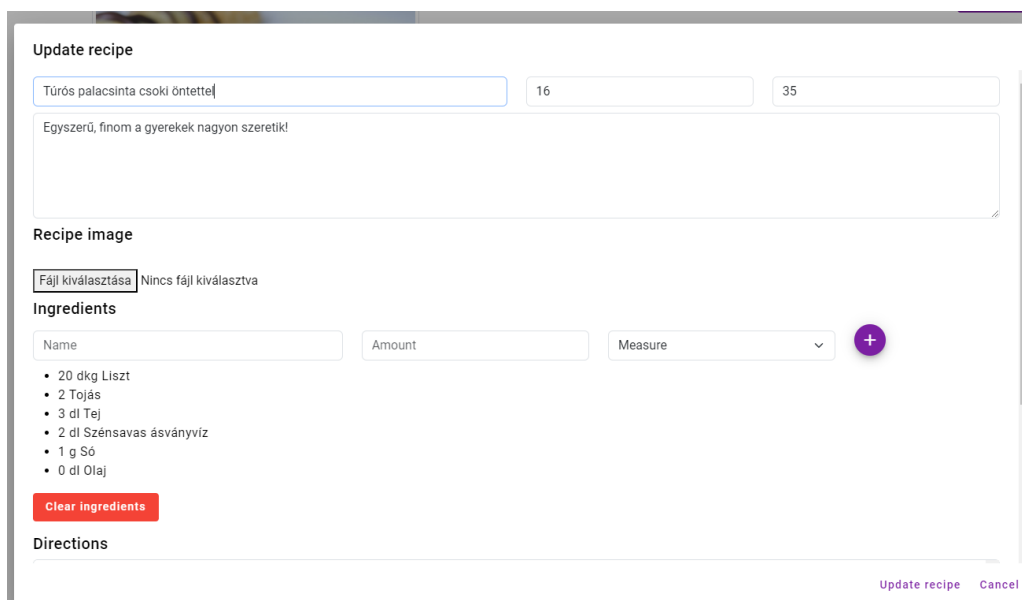
Egyszerű, finom a gyerekek nagyon szeretik!

Update recipe

Directions

A lisztet csomómentesre keverjük a tojással, tejjel, ásványvízzel, a sóval és egy kevés olajjal, majd egy órán át pihentetjük. Az első palacsinta sütése előtt a forró serpenyőbe egy kevés olajat öntünk (a következő palacsintáknál erre már nincs szükség, maximálisan elegendő, ami kisul a tésztából). A palacsintatésztát egyenként kisütjük. A túrót villával összetörjük, hozzáadjuk a tejszínt, a cukrot, a vaníliás cukrot és reszelt citromhéjat, és jól kikeverjük. A palacsintákat a töltelékkel megtöltjük és felfekerjük. A csokoládét vízgőz fölött megolvasztjuk, majd meglocsoljuk vele a palacsintákat.

A jobb felső sarokban található gomb megnyomása után megjelenik egy mentéshez hasonló felugró ablak, itt viszont már a jelenlegi receptek adatai betöltésre kerültek.

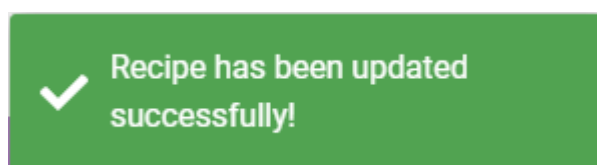


The screenshot shows a modal window titled "Update recipe". It contains several input fields: a text field for the recipe name (filled with "Túrós palacsinta csoki öntettel"), two numeric fields for servings (16 and 35), and a text area for the description (filled with "Egyszerű, finom a gyerekek nagyon szeretik!"). Below these is a "Recipe image" section with a "Fájl kiválasztása" button and the text "Nincs fájl kiválasztva". The "Ingredients" section has a table with columns "Name", "Amount", and "Measure", and a list of ingredients: "20 dkg Liszt", "2 Tojás", "3 dl Tej", "2 dl Szénsavas ásványvíz", "1 g Só", and "0 dl Olaj". There is a "Clear ingredients" button. At the bottom is a "Directions" section. The modal has "Update recipe" and "Cancel" buttons at the bottom right.

Ha az összetevőkben szeretnék módosítást tenni a felsorolás alatt található gomb segítségével kiüríthetjük a listát, hogy feltöltsük őket a helyes adatokkal.

3.5. Hibajelzések

A felhasználót nem hagyhatjuk kételyek között, egy-egy művelet során így azokat értesítési ablak jelzi az alkalmazás. Ha sikeres a művelet akkor zöld ablakban jelenik meg az információ.



Ha viszont hibára fut az adott művelet akkor piros színű ablak jelenik meg.

Az ablakok elegendő ideig maradnak láthatóak, hogy a felhasználó kényelmesen el tudja olvasni a keletkezett eredményről szóló információt.

3.6. A program készítőjének elérhetősége

Amennyiben hibát tapasztal, esetleg kérdése van az alkalmazáshoz kapcsolódóan, illetve fejlesztési javaslatai lennének, az alábbi e-mail címre várjuk levelét: info@recipebook.hu

4. Összefoglalás

Összességében, az elkészült eredménnyel elégedettek vagyunk, természetesen további fejlesztésekre van lehetőség, amelyekkel kényelmesebb és még inkább felhasználóbarát lehet az alkalmazás. A fejlesztés során a legnehezebb rész volt, hogy felismerjük, milyen kialakítás lehet a felhasználó részére optimális. Mivel mindenkinek van tapasztalata hasonló alkalmazások terén, így mindenkinek rengeteg javaslata, ötlete volt arra vonatkozóan, hogy hogyan tudjuk a felhasználói élményt növelni. Ezek a javaslatok gyakran egymást kizáróak voltak, nehéz volt eldönteni, hogy melyik megoldás optimális és meg is tudjuk valósítani. A közös munka során megtanultuk, hogy hogyan tudjuk támogatni egymás munkáját és azt, hogy milyen fontos egy fejlesztés során az, hogy összehangoltan dolgozzunk.

5. Irodalomjegyzék

Könyvek:

Fain, Yakov, és Anton Moiseev. Angular Development with TypeScript. 2. kiadás. Packt Publishing, 2021. ISBN 978-1800205266.

Bloch, Joshua. Effective Java. 3. kiadás. Addison-Wesley Professional, 2018. ISBN 978-0134685991.

Webes hivatkozások:

Introduction to the angular docs - <https://angular.io/docs>

Reactive Extensions Library for JavaScript - <https://rxjs.dev/guide/overview>

Getting Started with Angular Material - <https://material.angular.io/guide/getting-started>

Build fast, responsive sites with Bootstrap - <https://getbootstrap.com/docs/5.0/getting-started/introduction/>