

Szegedi Tudományegyetem

Informatikai Intézet

Recipe hoarder webes alkalmazás

(Recipe hoarder web application)

Szakdolgozat

Készítette:

Vas Laura

gazdaságinformatika szakos
hallgató

Témavezető:

Dr. Bilicki Vilmos

egyetemi adjunktus

Szeged

2021

Tartalomjegyzék

Feladatkiírás	4
Tartalmi összefoglaló	5
Motiváció	6
1. Piackutatás	7
1.1. Grocy	7
1.2. Delish	7
1.3. Yummly	8
1.4. BigOven	8
1.5. ChefTap	8
1.6. Összefoglaló	9
2. Funkcionális specifikáció	10
2.1. Bejelentkezés/Regisztráció	10
2.2. Receptek	11
2.2.1. Receptek hozzáadása	11
2.2.2. Receptgyűjtemény kezelés	11
2.2.3. Recept megtekintés	11
2.3. Bevásárló lista	11
2.3.1. Bevásárló listához adás	11
2.3.2. Bevásárló lista megtekintés	11
2.3.3. Bevásárló Listából törlés	11
3. Felhasznált technológiák	12
3.1. Angular	12
3.2. Angular Material	12
3.3. FireBase	13
3.4. PWA	13
3.5. Schema.org	14
3.6. Figma	14
4. A rendszer magas szintű áttekintése	15
4.1. Recept importálás	15
4.1.1. FetchUrlData	15
4.1.2. JsonLdExtractor	15
4.1.3. SeparateIngredients	15
4.1.4. ImageDownloader	15
4.1.5. CalorieCalculator	15
4.2. Kliens oldali adatbázis műveletek	15
4.3. Bevásárlólista ajánló	15

5. Architektúra	16
5.1. asd	16
6. Adatmodellek	17
6.1. Recipes	17
6.2. Users	18
6.3. Reports	19
7. Tesztelés	20
7.1. Recept importálás URL-en keresztül	20
7.1.1. FetchUrlData	20
7.1.2. JsonLdExtractor	20
7.1.3. SeparateIngredients	20
7.1.4. ImageDownloader	21
7.1.5. CalorieCalculator	21
7.2. GetData	21
8. Továbbfejlesztési lehetőségek	22
8.1. asd	22
Nyilatkozat	23
Köszönetnyilvánítás	24
Irodalomjegyzék	25

Feladatkiírás

A szakdolgozat során egy Angular keretrendszerben kialakított webes alkalmazás létrehozása volt a feladatom. A projekt a Firebase felhőalapú szolgáltatásait használja. A fejlesztés során a legfőbb cél a recept importálás más honlapokról volt. Az importálás második legfontosabb lépése az alapanyagok szétválogatása, hogy később a bevásárlólistába helyezésnél a megegyező anyagokat össze lehessen adni.

Tartalmi összefoglaló

- **Téma megnevezése:**
A szakdolgozat céljául kitűzött témám egy Angular-ban írt web applikáció, ami recept megjelenítésre és importálásra használható.
- **Feladat megfogalmazása:**
Az importálás funkció lehetővé teszi, hogy a felhasználók egy helyen gyűjtsék a receptjeiket. Továbbá, hogy a receptek összetevőit egy bevásárló listába ki tudják menteni, ezzel is megkönnyítve a mindennapi életet. A felhasználók a többiek által létrehozott receptek között tudnak keresni, és a nekik tetsző recepteket ki tudják menteni a saját receptgyűjteményeikbe.
- **Megoldási mód:**
Az applikáció egy weblap formájában lett megvalósítva, mivel így lehet a legtöbb eszközt elérni egyetlen kód bázissal. A megvalósításhoz a már említett Angular keretrendszert használtam, illetve a Firebase felhő alapú szolgáltatásait. Mivel mind a kettő (Angular, Firebase) a Google terméke, ezért várhatóan hosszútávon támogatva lesznek. A felhasználó a recept URL-je alapján tud receptet importálni, vagy manuálisan is lehet létrehozni újat. Ekkor az importáláshoz egy szerver oldali funkció fut le és próbálja értelmezni a megkapott URL-en lévő html fájlt. Ennek egy fontos lépése az, hogy az alapanyagok nevét, mértékegységét és mennyiségét az eredeti helyről kiolvassa. Miután a receptet sikeresen importáltuk, azokat a Firebase adatbázisában tároljuk.
- **Alkalmazott eszközök, módszerek:**
Mind az importálás mind az egész projekt során törekedtem, hogy minél modulárisabb legyen a felépítés. A webapp fejlesztése során a PWA-t alkalmazva elérhető, hogy bizonyos funkciók offline is működjenek. Ezt a modern, könnyen kezelhető weblapot számítógépen és telefonon egyaránt lehet használni.
- **Elért eredmény:**
A fejlesztés során sikerült egy modern telefonos és számítógépes környezetben is elérhető webes alkalmazást készíteni, ami bárki számára a regisztráció után elérhető és könnyen használható, ezzel egyszerűbbé téve a hétköznapiakat.
- **Kulcsszavak:**
Angular, Firebase, pipeline architektúra, PWA, telefonos nézet

Motiváció

Egyetemisták, mint én is egyre közelebb vagyunk ahhoz az életformához, ahol önellátók vagyunk, ennek fontos része a főzés és étkezés. Manapság nagyon egyszerű különböző recepteket, különböző országokból, kultúrákból találni, viszont ez temérdeknyi weblapot jelenthet. Ennek hátulütője, hogy egy idő után követhetetlen lesz, hogy egyáltalán hova regisztráltunk, valamint, hogy “melyik weblapon is volt az a bizonyos recept, amit egyszer már kipróbáltam, és tetszett”. Személyes tapasztalatom ezzel kapcsolatba pedig, hogy én egy chat alkalmazásban gyűjtöttük a barátommal az URL címeket, hogy legközelebb is megtaláljuk, de már kezdett nagyon követhetetlen lenni.

Azért választottam ezt az ötletet a szakdolgozatom témájának, mert ez egy személyes problémám már hosszú ideje. Láttam már korábban próbálkozásokat, de egyik sem volt az én elképzelésemnek megfelelő. A célom az volt, hogy egy egyszerű URL cím másolással pillanatok alatt egy helyen lehessen a megtalálni mindent.

A továbbiakban részletesen kifejtem az általam tervezett és megvalósított webes applikáció felépítését és funkcióit. A bemutatót a konkurencia ismertetésével kezdem.

1. fejezet

Piackutatás

Már létező programokra öt példát hoztam, amik mind valamilyen szinten különböznek.

Felhasználókörük, funkcióik, előnyök és hátrányok az én tervemhez képest. Az én terveimhez képest egy rövis ismertető az előnyeikről valamint hátrányaikról. Miért jobb vagy mivel tudnak többet, mint amit én terveztem egyenlőre kialakítani.

1.1. Grocy

A Grocy egy lokálisan hosztolható weblap, ez azt jelenti, hogy minden felhasználónak rendelkeznie kell egy szerver géppel, amin magát a weblapot tudja üzemeltetni. Maga a weblap kifejezetten sok funkcióval rendelkezik, de ezeknek a teljes kihasználásához sok időt és munkát kell befektetni. Sajnos mivel mindenki saját magának futtatja, ezért a beépített adatbázisa üresen kezd. Ellenben, ha valaki hosszú ideig használja, akkor a befektetett munka kifizetődő tud lenni.

A recept kezelő lapja csak manuálisan feltölthető, tehát nincs importálásra lehetőség. Rendelkezik bevásárlólista és “sufni” opciókkal is. Az otthon lévő alapanyagokat egyessével, tetszőleges részletességgel fel lehet venni a “sufniba”, ezzel leltározva, hogy milyen alapanyagok vannak otthon. Ezekről eltárolható adatok közé tartozik, hogy mennyi van belőle, meddig jók, képet, de akár a vonalkódját is. A bevásárló lista pedig egyértelműen a vásárlást segítő funkció, aminek a végén, egy kattintásra átrakható “sufniba”.

Már ezen leírás alapján is látszik, hogy ahhoz, hogy ez a rendszer használható legyen, egy komoly lokális adatbázist kell létrehozni az alapanyagokból és azok adatairól, valamint a receptekről. Ez a rendszer csak olyan emberek számára használható, akik rendelkeznek hardverrel és tudással, hogy maguknak futtassák a weblapot.

1.2. Delish

Ezt a weblapot azért választottam példaként, mert ez egy tökéletes példa egy átlagos, egyszerű receptes weblapokra. A honlapon csak recepteket és talán pár blog bejegyzés található regisztráció után is. Ez a weblap reprezentálja a legtöbb hasonló, csak blogként működő weboldalt.

Egy receptre kattintva látjuk az alapadatokat, hozzávalókat, elkészítési javaslatot valamint alap adatokat mint az elészítési idő. A weblapon találhatóak még hasonló recept ajánlások, de ezen felül több szolgáltatást nem nyújt.

1.3. Yummly

Ez egy fejlettebb verziója a korábban említett "átlagos" weblapoknak. Bejelentkezés nélkül kevesebb funkció érhető el, viszont utána már kifejezetten sok funkcionalitása van. A webes kinézetén felül applikációval is rendelkezik.

Az alap recept keresésén kívül, itt már lehetőségünk van azok elmentésére a sajátjaink közé. A weblap rendelkezik bevásárló lista funkcióval, valamint képes azonnal a receptből áthelyezni az alapanyagokat is. Egy kiemelkedő funkciója az étkezés tervező. Ez, figyelembe véve esetleges allergiákat, vagy étrendeket ajánj és segít tervezni a következő időszakra.

Ami hátrány az egész weblapon, hogy nem közösség bővíti a recept adatbázist, ezért limitált a receptek száma és nem lehet mindent megtalálni. Még akkor is, ha figyelembe vesszük a manuális recept készítést, nem feltétlenül a legegyszerűbb használni, mivel mindig egy külső helyről ki kell kikeressük amit akarunk, majd kézzel beírunk.

1.4. BigOven

A legnagyobb különbség az eddigiekhez képest, hogy ez a weblap már rendelkezik recept importáló funkcióval is. Azon felül négy különböző módon lehet újjakat létrehozni, manuálisan leírva, másolni egy már létező fájlból és beilleszteni, képről vagy scennelésről beolvasni vagy URL címmel importálni. Az importálás során nem tárolják el az egész receptet, ha más honlapról származik. "Our Pledge to Food Bloggers" leírja, hogy miért, viszont ez azt jelenti, hogy a teljes receptet megtekintsük, át kell navigálni az eredeti oldalra. Ezen felül a bevásárló listában nem adódnak össze a termékek, valamint nincsenek kategóriák a receptekhez.

Egy nagy hátránya a weblapnak, hogy kissé régi stílusú. Az oldal nem használ túl sok modern stílust. A képek, beviteli mezők, lista nézetek mind úgy néznek ki, mint amin épp hogy van egy kis formázás. A weboldal navigációja nem túl felhasználóbarát.

Annak ellenére, hogy a weblapnak mennyire nem modern stílusa van, az applikáció igenis követhető. A funkciók szintén jól működnek. Elméletileg IOS-en is létezik, viszont arra nincs lehetőségem, hogy felmérjem milyen különbségek lehetnek a két verzió között. Egy nagy előny, hogy az ingyenes verzióban is használhatóak az alapfunkciók.

1.5. ChefTap

Az összes közül valószínűleg ez az applikáció, ami a legtöbb funkcióval rendelkezik. Technikailag van webes és telefonos verziója is, viszont a webes csak recept megtekintésre használható. Minden egyéb, beleértve a recept importálást, bevásárló listákat és étkezés tervezőt csak az applikációk keresztül lehet elérni és szerkeszteni. A weben volt lehetőség Google segítségével bejelentkezni, viszont az applikációnak nem volt ilyen lehetősége. Ennél az appnál az ingyenes verzió elég limitált, a recept importáláson kívül semmi sem működik a próbaidőszak lejárta után.

Ezen a felületen nincs mások által, vagy akár csak egy közös adatbázisból való keresésre és importálása lehetőség a recepteknél. A felhasználónak mindent magának kell beszerezni.

A beimportált recepteket könnyű módosítani, valamint rengeteg kis adatot megadni, hogy otthonosan lehessen használni a környezetet. Ebben az applikációban nincs lehe-

tőség közvetlenül a receptből a bevásárló listába rakni alapanyagokat, menüket összekészíteni vagy az étkezéstervezőt használni az ingyenes próbaverzió után.

1.6. Összefoglaló

Egy táblázat a korábban összefoglalt példákról és tudásaikról egy egyszerű összehasonlításhoz.

Összegzés

weblapok	Grocy	Delish	Yummly	BigOven	ChefTap	RecipeHoarder
open source	✓	✗	✗	✗	✗	✗ ¹
fizetős	✗	✗	✗	✓	✓	✗ ²
recept importálás - URL	✗	✗	✗	✓	✓	✓
receptekhez vannak lépések	✗	✓	✓	✗	✓	✓
manuális recept hozzáadás	✓	✗	✗	✓	✓	✓
bevásárlólista	✓	✗	✓	✓	✓	✓
bevásárlólista ajánló	✓	✗	✗	✗	✗	✓
kategóriák használata receptekhez	✗	✓	✓	✗	✓	✓
étkezés tervező	✓	✗	✓	✗	✗	✗
bejelentkezés google fiókkal	✗	✓	✓	✓	✗	✓
reszponzív web-lap	✓	✗	✗	✗	✗	✓

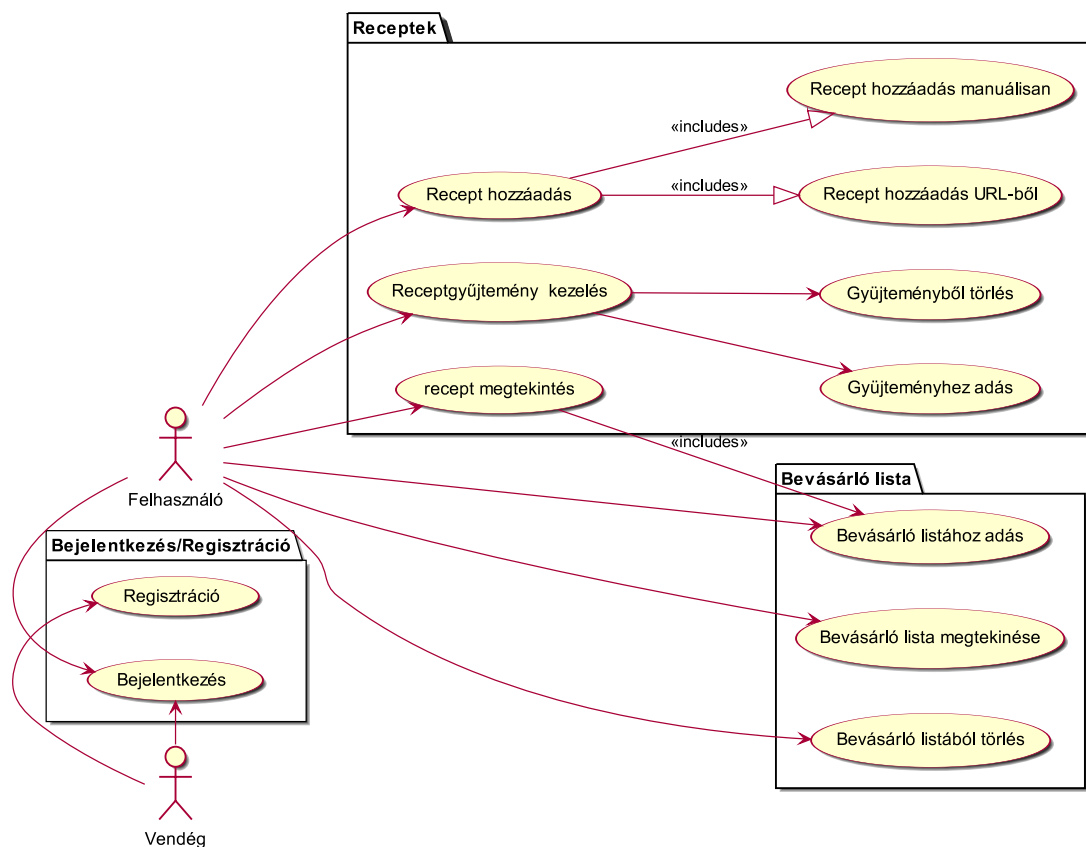
¹Technikailag az, de nem könnyű hosztolni.

²Jelenleg nem fizetős, de amennyiben fellendül az applikáció népszerűsége, akkor Firebase-el könnyen bevezethető.

2. fejezet

Funkcionális specifikáció

Egy összefoglaló részletesebb arról, hogy minek pontosan hogyan kell működnie az eredeti terv szerint. Az ábra a vizuális megfelelője ennek, valamint lentebb az al bekezdésekben ki lesz fejtve az ábra egyes részei.



2.1. Bejelentkezés/Regisztráció

A weblap legelső megnyitásakor a bejelentkező oldalra jutunk. Az oldalt látogatók első sorban mindig vendég szerepkörben lesznek képesek megtekinteni a tartalmat, ami ebben az esetben kizárólagosan a bejelentkező és regisztrációs lap.

Ahhoz, hogy a wb applikációt a vendég használni is tudja regisztrálni kell. Erre két lehetőség van. Egyrészt lehetőség van a regisztrációs oldalon a mezők helyes kitöltésével manuálisan regisztrálni. Másrészt, ha a vendégnek van Gmail-es fiókja, akkor

egyszerűen egy lépésben azonnal regisztrálhat azon keresztül. Ezek a diagramon a “Bejelentkezés/Regisztráció” alatt lévő ábrának felelnek meg.

2.2. Receptek

AA következőkben a receptekkel való főbb funkciókról lesz szó. Ez az ábrán a “Receptek” kijelölt részben lévőket fogja részletezni.

2.2.1. Receptek hozzáadása

A weblap egyik jellegzetes alapfunkciója, hogy a közösség tölti fel az adatbázis receptekkel. Ez azért jó, mert így nem a adminisztrátoroknak kell folyamatosan dolgozni, hogy minél több recept legyen elérhető a felhasználók számára. Másrészt, ez egy szinten hátrány is, mivel így több lehetőség van arra, hogy a feltöltés hibás receptet kapjunk, amit a közösség nem tud javítani. Kizárólag bejelentésre van lehetőség, ha hibás receptet találnak.

Recept feltöltésre két lehetőség van. Az egyik, hogy manuálisan kitölti a felhasználó az importáló oldalon a mezőket. A másik, hogy egy másik receptes weblapról kimásolja az URL címet a böngésző sáv tetejéről és beilleszti a linket RecipeHoarder importáló felületén. Amennyiben a receptes oldal olyan, amit ez a weblap képes feldolgozni, akkor elindul egy folyamat, ami feldolgozza és kiszedi a kellő adatokat az eredeti honlapról, majd beilleszti az ellenőrző, módosítható mezőkbe. Ezeket a felhasználó ellenőrzi, akár kiegészíti ha kell, majd egy gombnyomással már el is lett mentve az adatbázisba.

Ez az ábrán a “Receptek” rész alatt, a “Recept hozzáadás”-át írta le a hozzátartozó manuális és URL-s verzióval.

2.2.2. Receptgyűjtemény kezelés

A

2.2.3. Recept megtekintés

A

2.3. Bevásárló lista

2.3.1. Bevásárló listához adás

2.3.2. Bevásárló lista megtekintés

2.3.3. Bevásárló Listából törlés

3. fejezet

Felhasznált technológiák

Itt fogom részletezni a kiemeltebb felhasznált technológiákról alapvető információkat, valamint szerepüket ebben a projektben.

3.1. Angular

Egy TypeScript alapú Framework, ami a eredetileg a Google fejlesztette ki, de most már nyílt forráskódú. Az Angular egy teljes újraírása a régi AngularJS-nek. Egyik legnagyobb előnye, hogy komponens alapú keretrendszer, ami segítségével újra felhasználható kódot lehet írni. Támogatja a Single Page alkalmazás készítését. Ezeken a weblapokon a weblap újratöltése nélkül tud a felhasználó navigálni, ezzel is gyorsabbá téve azt.

A felhasználói felület tervezés során is igyekeztem komponenseket kialakítani mind telefonos mind desktopos kinézetre.

...

A tervezés során a Figmában is komponensekkel terveztem meg a weblap kinézetét mind telefonos mind desktopos formában. A komponenseket egyszerű létrehozni, valamint felhasználni a megfelelő helyeken. Ha magunknak akarjuk tervezni a komponenseket, akkor sima HTML/CSS-el elérhető, amennyiben szeretnénk már egy népszerű és könnyen integrálható komponens bázist felhasználni, akkor az Angular Material használható.

3.2. Angular Material

Az Angular Material egy a Google által fejlesztett komponens bázis. Ez segít a fejlesztőknek már kész és modern dizájnos komponenseket egyszerűen felhasználni a projektekben. Ez elsősorban egy UI komponens dizájn könyvtár, ami integrálás után, csak importokkal már azonnal működik. A weblapján pedig példákkal és részletes dokumentációval felsorolja a létező felhasználható komponenseket és különböző változataikat.

Az én projektem is nagy részben ennek a könyvtárnak a komponenseit felhasználva épült fel. Csak pár helyen lett kiegészítve a már létező verzió, hogy illeszkedjek az eredeti tervekhez. Ezen felül az Theming azaz a projekt témája is az Angular Material alapján lett felépítve, hogy a komponenseknek könnyen meglehessen adni az alap színeket és tipográfia szabályokat.

3.3. Firebase

Mostmár a Google által fejlesztett webes és mobilos alkalmazások készítéséhez létrehozott platform. Manapság nagyon népszerű az egyszerűsített adatbázis felépítése valamint a könnyű és könnyen értelmezhető kezelő felülete, arról nem beszélve, hogy rengeteg dokumentáció létezik a fejlesztők számára.

Ebben a projektben elég sok helyen használom a Firebase funkcióit. Az egész weblap a Firebase Hosting felületéről működik. A GitHub-bal összekötve, ha a Main Branch-re van feltöltés, akkor a Firebase-en automatikusan deploy-oldódik az új production verzió.

A Firestore Database-t használom a weblap adatok tárolásához. Itt vannak eltárolva receptek, felhasználók, felhasználókhöz különböző kollekciók és bevásárló lista meta adatok. A Cloud Firestore-on belül lehet beállítani az adatbázis Rule-okat. Ezt azt jelenti, hogy be lehet állítani, hogy különböző kollekciókat kik érhetik el, milyen tevékenységek vannak engedélyezve nekik, valamint, hogy milyen jogokkal milyen adatbázis műveleteket lehet csinálni. Amíg nem volt autentikáció beállítva és a development folyamat volt, akkor csak egy dátumot frissítettem, hogy meddig lehet az összes felhasználónak bármit csinálni. Viszont most már le van tiltva a receptek törése, valamint bejelentkezés nélküli lekérdezések is.

A Firebase Storage-ban tárolódnak a receptekből kimentett képek, amik az import során kerülnek az adatbázisba.

A legnagyobb biztonság érdekében a Firebase Auth-ot használom a weblap bejelentkezés és regisztrációs műveletekhez. Így lehetséges manuálisan regisztrálni majd bejelentkezni, viszont, akinek van Gmail fiókja annak lehetősége van egy lépésben azonnal regisztrálni és belépni annak segítségével.

A recept importálás és a bevásárlólista egy része a Firebase Cloud Function keresztül szerver oldalon fut. Ez nem csak olcsóbb, mivel szerver oldali lekérdezések ingyenesek a kliens oldalival szemben, hanem az importálás során felmerült problémát is megoldotta. Az importálásnál felmerült, hogy egy honlap nem érdekez le adatot egy másiktól kliens oldalon keresztül, valamint a CORS okozta nehézségek elkerülése érdekében is hasznos volt. Azon kívül, mikor a bevásárló listába új alakanyag kerül, akkor egy trigger lefut a szerver oldalon, ami segítségével állítjuk elő az ajánlóhoz a metaadatokat.

3.4. PWA

PWA azaz Progressive Web Application egy koncepció, amit követve lehetséges megbízható, alkalmazható és telepíthető webes applikációkat fejleszteni. Az alap szabályait követve a weblap platformspecifikus alkalmazássá alakítható. A PWA lehetővé teszi az alkalmazások használatát bárhol, bármikor, bármilyen eszközön.

Ez eseten az Angular-t kellett kiegészíteni egy kevés plusz JavaScript kóddal, hogy ha az eszköznek nincs internet elérése, akkor a script alapján a localStorage-ba elmentett, valamint cache adatokat használja az indításhoz. Ezzel elérhető, hogy valamilyen szinten használható a weblap offline környezetben is.

3.5. Schema.org

A shema.org egy weblap, ami azért készült, hogy a közösség a központosított sémákat használja különböző weblapokon, e-mailekben, modellekben, adatbázisok tervezésében vagy sok más helyen. Sok különböző kódolással használható, de a legnépszerűbbek a JSON-LD, Microdata valamint a RDFa. Ezek a szótárak olyan entitásokat és kapcsolatokat írnak le, amik segítik a fejlesztők számára a megfelelő sémát kiválasztani. Tehát összességében, ez egy tervgyűjtemény.

A projektben fontos szerepet játszott a shema.org által biztosított “Recipe” séma feldolgozása és megértése, mivel a recept importálás más weblapokról ezen alapul. A jól felépített weblapok manapság tartalmaznak egy vagy több JSON-LD meta információt, amiben a weblapon jelenleg megjelenő adatok vannak eltárolva JSON formátumban a Shema.org szerinti felépítés alapján.

3.6. Figma

Egy vektorgrafikus szerkesztő, ahol web prototípusokat lehet tervezni. Ez legfőképp webalapú, de letölthető desktopos verziója is. A program képes a tervezett lapok élő tükrözésére akár saját telefonon az applikáción keresztül, vagy csak az emuláló felületen keresztül is. A figma egy modern dizájn tervező felület, ahova egyszerűen integrálni lehet a material design stílust és alap komponenseket, ezzel is segítve az Angular component szerű gondolkodást. Másik nagy előnye, hogy interaktívvá lehet tenni a tervezés során a gombokat, komponenseket. Valamint a kész dizájnoknak meglehet nézni a pontos CSS stílus kódját is, ezzel is segítve a konkrét kódolás folyamatát.

Sokat segített a tervezésben, mivel a kész material design komponenseket tudtam próbálgatni anélkül, hogy először mélyebben tanulmányoznám az Angular felépítését. Az egységes dizájn és téma felépítésében pedig segített konzisztensnek maradnom. Ez igaz mint a színekre, mind a tipográfiára.

4. fejezet

A rendszer magas szintű áttekintése

Ebben a fejezetben fogom leírni a különböző fontosabb funkciókról a részletesebb felépítését és tudnivalókat.

4.1. Recept importálás

A moduláris felépítés miatt ezt több al részre szedem.

4.1.1. FetchUrlData

asd

4.1.2. JsonLdExtractor

asd

4.1.3. SeparateIngredients

asd

4.1.4. ImageDownloader

asd

4.1.5. CalorieCalculator

asd

4.2. Kliens oldali adatbázis műveletek

Services...

4.3. Bevásárlólista ajánló

trigger és működés

5. fejezet

Architektúra

5.1. asd

6. fejezet

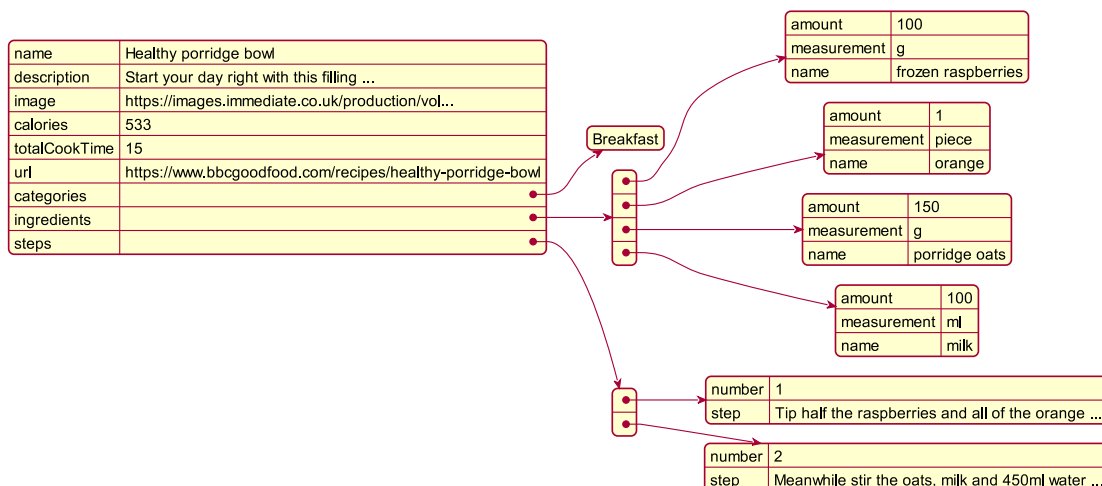
Adatmodellek

Az adatbázis felépítése különbözik a megszokott táblás módszertől, mivel a Firebase egy saját kifejlesztett módszerrel tárolja az adatokat, amivel gyorsabb a keresés és írás az adatok között. A szerkezete nagyon egyszerű, mivel csak egyszerű kollekciókból és dokumentumokból áll, felépítése hasonlít a JSON fájl típusokra. Minden adat egy-egy dokumentumban tárolódik, és ezek a dokumentumok vannak elrendezve kollekcióban. Ennek a felépítésnek a hátránya viszont, hogy nehéz az adatbázis struktúrát frissíteni, mert csak az új adatoknál fog megjelenni, ha nem indítunk rá egy külön funkciót. Ezért is macerásabb a régi adatokhoz változtatások után az új adattalba értéket adni.

6.1. Recipes

Az adatbázisban az egyik legfontosabb adat, mivel erre épül az egész weblap témája. A receptek a feltöltés során automatikus azonosító ID értéket kapnak, ami a weblapon az URL címben az azonosításért is felelős. Ezen belül tároljuk az adott recept adatait, ezek közé tartozik például: kalória, teljes főzési idő (percben mérve), recept neve, eredeti URL, stb. Ezek az alap adatok, csak simán kulcs érték párok.

Jelenleg még kulcs-tömb értékben tárolva az alapanyagok és lépések, viszont a jövőben ez lehet megváltozik subcollection-re a gyorsabb betöltés idő eléréseért. Ezek a kulcs-tömb párosok hasonlóak a korábbi típushoz, viszont itt egy azonosító értékhez egy tömbnyi belső adat tartozik. Ez lehetővé teszi, hogy az alapanyagokon belül eltárolhassuk egyesével a mennyiséget, mértékegységet és az alapanyag nevét. Valamint a recept elkészítésének lépésére el van tárolva a sorrendbeli szám és maga a lépés.



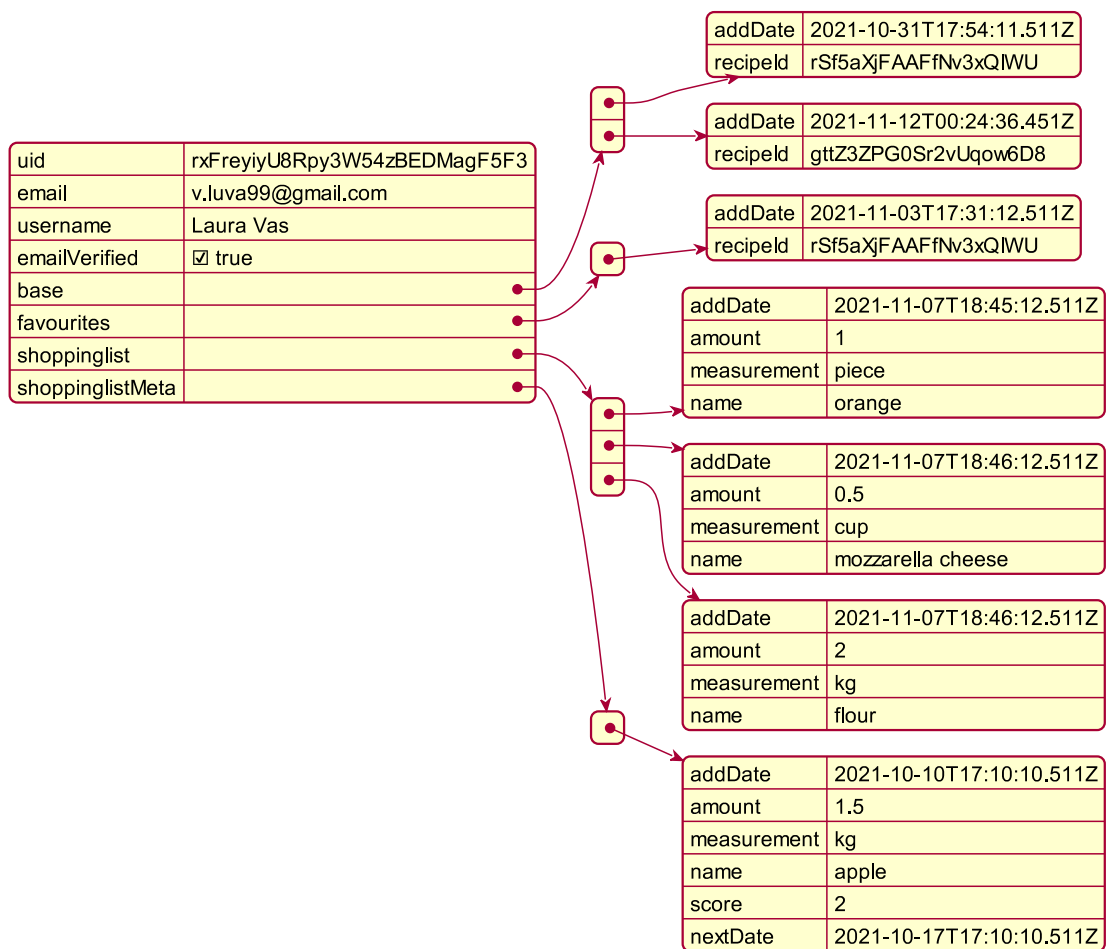
6.2. Users

Az adatbázisban a következő adathalmaz a felhasználók kollekciója. A regisztráció során létrejön egy új dokumentum az adatbázis felületen is. Itt tároljuk el az egyéb adatokat, amit a felhasználó az alap e-mail és jelszón kívül megad. Az egyéb adatok segíthetnek a jövőben személyre szabott recept ajánlót készíteni.

Itt a korábbi recepthez tartozó alapanyagok tömbhöz képest már egy subcollection-ban vannak eltárolva az egyéb gyűjtemények, amit a felhasználó a böngészés során elmentett. Ezek közé tartozik a kimentett receptek listája, kedvenc receptek, bevásárló lista valamint a bevásárló listához tartozó ajánlónak a metaadatai.

A subcollection azért fontos ebben az esetben, mert mikor le akarjuk kérdezni a felhasználó kedvenc receptjei listát, akkor nem kell mindig az egész felhasználó adatot lekérdezni az összes többi kollekcióval együtt, hanem lekérdezhető ugyanúgy, mint a többi kollekció egyszerűen. Ezzel gyorsabb lesz a weblap, valamint spórolunk a felesleges adatok kezelésével is. Amikor a felhasználó először elment a könyvtárába egy receptet, akkor a saját ID-ja alatt létrejön egy “base” kollekció, ami eltárolja, hogy mikor lett a listához adva a recept, valamint, hogy melyik lett kimentve a recept ID-ja alapján. Ugyanez igaz a “favourites” kollekcióra. Amint a felhasználó kimentett egy receptet a saját listájába, akkor van lehetősége ezt tovább elmenteni a kedvencek közé. Ez ugyanúgy működik mint a sima listába mentés. Amennyiben viszont a saját listájából törli a receptet, akkor a kedvencek közül is eltűnik.

A bevásárló listába a mentés dátumát és a pontos adatokat is elmentjük az alapanyag-ról, hogy utána az egész könnyen kilistázható legyen. A létrehozás dátuma a sorrendbe rendezés és a metaadat számítás miatt fontos. Ha a felhasználónak nem kell többé az adat, akkor törlődik a listából. Végül a metaadat, ami kettő listához adás értékéből számolunk ki. Az ajánló jelenleg csak az alapanyag nevét ajánlja fel, viszont a mértékegység és mennyiség is el van tárolva, hátha a jövőben pontosabb számítást szeretnénk csinálni. Ezen kívül, mikor az adat létrejön, a “score” értéke 2. Ez egy érték, ami segít az ajánlás sikerességét követni, max értéke 5, minimum értéke 0. Végül a “nextDate” egy számot TimeStamp érték, ami a eredeti hozzáadás dátumából és az újonnan hozzáadás dátumádól számolunk ki.



6.3. Reports

Mivel a weblap úgy lett tervezve, hogy a közösség építi, és mindenki ugyanazokat a recepteket látja, ezért lennie kell egy lehetőségnek, ahol a felhasználók hibát tudnak jelezni. Tehát minden recept végén van egy felület, ahol jelzést lehet küldeni, ha hibás receptet találtak. A hiba bejelentés opciók előre be vannak állítva, így könnyebb lehet a jövőben megkeresni, hogy mi lehet a hiba az adott recepttel.

A “creationDate” a bejelentés elküldését jelzi. A “problem” pedig az opciókból választott a problémához legközelebbi lehetőség, amit a felhasználó választ. Valamint végül feljegyezzük, hogy melyik receptnél volt ez és hogy ki küldte. A “userId” meg egyezik a korábban említett “uid” értékkel a “users” kollekciónál, mivel ez az érték, ami a bejelentkezés során az auth adott a felhasználónak.

creationDate	2021-11-26T17:03:24.087Z
problem	Other
recipeld	1A3QX5z5n8nx2AWAo8ag
userId	rxFreyiyU8Rpy3W54zBEDMagF5F3

7. fejezet

Tesztelés

A fontosabb funkciók, amik meghatározzák az alap témáját a weblapnak tesztelve lettek. A szerver oldali funkciók mind, az alap működések közé tartoznak, ezért a cloud function moduloknak be lett állítva automatikus tesztelés. Ennek segítségével, ha valami nagyobb változtatást csináltunk, mint például egy rendszer szintű frissítés a Firebase csomagnak, akkor egyszerűen le lehet tesztelni, hogy még minden jól működik-e. Ezt egyszerű unit tesztekkel oldottam meg. A teszteket az npm-ről szedett Jasmine csomaggal készítettem.

7.1. Recept importálás URL-en keresztül

Mivel ez a folyamat modulárisan lett felépítve, ezért minden egyes lépését egyszerű tesztelni. A modulok egyesével és a végeredmény is külön tesztelve van.

7.1.1. FetchUrlData

7.1.2. JsonLdExtractor

7.1.3. SeparateIngredients

Az alapanyag szétválasztás az egyik legkényesebb része a importálás folyamatnak, mivel ebben a részben egy string adatot kell megvizsgálni és különböző adattagokra osztani. Ehhez különböző regex formulák lettek kifejlesztve, hogy a különböző eseteket minél jobban és pontosabban tudjuk feldolgozni. Ehhez, egy egyszerű alap és egy komplexebb teszt készült.

Az első célja, hogy le ellenőrizze, hogy a tört szám karakter és egyéb tört szám változatok helyesen vannak átkonvertálva float értékre. A második teszt pedig egy tömböt kap értéknek, amiben rengeteg különböző eset van felsorolva olyan példákkal, amit élő importálás során kaphatunk és szélsőséges probléma lehet. Ezek közé tartozik a három alapeset, ahol vagy szépen visszaadja a mennyiséget, mértékegységet és az alapanyag megnevezését, vagy ha nem talál mértékegységet, de van szám, akkor berendezi, hogy mennyi datab alapanyagot talált, vagy végül, ha nincs szám se, akkor csak simán visszaadja alapértelmezett módon. Ezen kívül le kell ellenőrizni azokat az eseteket, amik komplexé tették a regex teszteket például, a korábban említett tört karakter helyes észlelése és formázása.

7.1.4. ImageDownloader

7.1.5. CalorieCalculator

7.2. GetData

8. fejezet

Továbbfejlesztési lehetőségek

8.1. asd

Nyilatkozat

Alulírott szakos hallgató, kijelentem, hogy a dolgozatomat a Szegedi Tudományegyetem, Informatikai Intézet Tanszékén készítettem, diploma megszerzése érdekében.

Kijelentem, hogy a dolgozatot más szakon korábban nem védtem meg, saját munkám eredménye, és csak a hivatkozott forrásokat (szakirodalom, eszközök, stb.) használtam fel.

Tudomásul veszem, hogy szakdolgozatomat / diplomamunkámat a Szegedi Tudományegyetem Informatikai Intézet könyvtárában, a helyben olvasható könyvek között helyezik el.

Szeged, 2021. november 27.

.....
aláírás

Alulírott szakos hallgató, kijelentem, hogy a dolgozatomat a Szegedi Tudományegyetem, Informatikai Intézet Tanszékén készítettem, diploma megszerzése érdekében.

Kijelentem, hogy a dolgozatot más szakon korábban nem védtem meg, saját munkám eredménye, és csak a hivatkozott forrásokat (szakirodalom, eszközök, stb.) használtam fel.

Tudomásul veszem, hogy szakdolgozatomat / diplomamunkámat a TVSZ 4. sz. mellékletében leírtak szerint kezelik.

Szeged, 2021. november 27.

.....
aláírás

Köszönetnyilvánítás

Ezúton szeretnék köszönetet mondani **X. Y-nak** ezért és ezért . . .

Irodalomjegyzék

- [1] J. L. Gischer, The equational theory of pomsets. *Theoret. Comput. Sci.*, **61**(1988), 199–224.
- [2] J.-E. Pin, *Varieties of Formal Languages*, Plenum Publishing Corp., New York, 1986.