

**SAPIENTIA ERDÉLYI MAGYAR TUDOMÁNYEGYETEM
MAROSVÁSÁRHELYI KAR,
INFORMATIKA SZAK**



**SAPIENTIA
ERDÉLYI MAGYAR
TUDOMÁNYEGYETEM**

Kitchn: Egy recept alkalmazás modern ételrajongók számára

DIPLOMADOLGOZAT

Témavezető:

Dr. Jánosi-Rancz Katalin Tünde,
Egyetemi adjunktus

Végzős hallgató:

Incze Zsolt-Tamás

2023

**UNIVERSITATEA SAPIENTIA DIN CLUJ-NAPOCA
FACULTATEA DE ȘTIINȚE TEHNICE ȘI UMANISTE,
SPECIALIZAREA INFORMATICĂ**



**UNIVERSITATEA
SAPIENTIA**

Kitchn: O aplicație de rețete pentru entuziaștii de mâncare modernă

LUCRARE DE DIPLOMĂ

Coordonator științific: Dr. Jánosi-Rancz Katalin Tünde, Lector universitar
Absolvent: Incze Zsolt-Tamás
2023

**SAPIENTIA HUNGARIAN UNIVERSITY OF
TRANSYLVANIA**
FACULTY OF TECHNICAL AND HUMAN SCIENCES
COMPUTER SCIENCE SPECIALIZATION



SAPIENTIA
HUNGARIAN UNIVERSITY
OF TRANSYLVANIA

Kitchn: A Recipe Application for Modern Food Enthusiasts

BACHELOR THESIS

Scientific advisor: Dr. Jánosi-Rancz Katalin Tünde,
Lecturer Student: Incze Zsolt-Tamás
2023

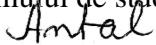
<p>UNIVERSITATEA „SAPIENTIA” din CLUJ-NAPOCA Facultatea de Științe Tehnice și Umaniste din Târgu Mureș Programul de studii: Informatică</p>	Viza facultății:
LUCRARE DE DIPLOMĂ	
<p>Coordonator științific: Dr. Jánosi-Rancz Katalin Tünde</p>	<p>Candidat: Incze Zsolt-Tamás Anul absolvirii: 2023</p>
<p>a) Tema lucrării de licență:</p> <p>Tema lucrării este dezvoltare unei aplicații mobile cross-plataformă iOS, Android și Web cu accent pe rețete culinare, gătit și gastronomie.</p>	
<p>b) Problemele principale tratate:</p> <p>Crearea unei aplicații cross-platforme cu buget și timp de creare minim. Crearea unui panel admin utilizând aceeași resurse din cod existent. Crearea unei aplicații de rețete pentru începători. Comparația între platforme native și Flutter.</p>	
<p>c) Desene obligatorii:</p> <p>Comparație între Flutter și platforme native. Comparație între MVVM și BLoC.</p>	
<p>d) Softuri obligatorii:</p> <p>Aplicație bazată pe tehnologii Flutter care realizează o aplicație mobilă pe platformele iOS, Android și Web. Compilarea la iOS se face prin XCode, și la Android din Android Studio. Baza de date este realizată cu Firebase Realtime Database. Aplicația va afișa rețetele primite de la Firebase. Prinț-un admin panel Web se poate modifica tabelul rețetelor.</p>	
<p>e) Bibliografia recomandată:</p> <p>Wenhao Wu. React native vs flutter, cross-platforms mobile application frameworks, 2018</p>	
<p>Ola Dahl. Exploring end user's perception of flutter mobile apps, 2019</p>	
<p>Gusti Pangestu, Ahmad Afif Supianto and Fitri Utaminingrum. Food recipe finder mobile applications based on similarity of materials. In 2018 International Conference on Sustainable Information Engineering and Technology (SIET), pages 156-161. IEEE, 2018.</p>	
<p>Paulo Meirelles, Carla SR Aguiar, Felipe Assis, Rodrigo Siqueira, and Alfredo Goldman. A students' perspective of native and cross-platform approaches for mobile application development. In Computational Science and Its Applications–ICCSA 2019: 19th International Conference, Saint Petersburg, Russia, July 1–4, 2019, Proceedings, Part V 19, pages 586–601. Springer, 2019</p>	
<p>f) Termene obligatorii de consultații: Studentul a început dezvoltarea aplicației în septembrie 2022 și ne am întâlnit la fiecare 2-3 săptămâni</p>	
<p>g) Locul și durata practiciei: Universitatea „Sapienția” din Cluj-Napoca, Facultatea de Științe Tehnice și Umaniste din Târgu Mureș, sala / laboratorul 327A</p>	
<p>Primit tema la data de: mai 2022</p>	

Termen de predare: 2 iulie 2023

Semnătura Director Departament



Semnătura responsabilului
programului de studiu



Semnătura coordonatorului



Semnătura candidatului



Declarație

Subsemnatul/a INCZE ZSOLT-TAMÁS, absolvent(ă) al/a specializării INFORMATICA, promoția 2023... cunoscând prevederile Legii Educației Naționale 1/2011 și a Codului de etică și deontologie profesională a Universității Sapientia cu privire la furt intelectual declar pe propria răspundere că prezenta lucrare de licență/proiect de diplomă/disertație se bazează pe activitatea personală, cercetarea/proiectarea este efectuată de mine, informațiile și datele preluate din literatura de specialitate sunt citate în mod corespunzător.

Localitatea, TÂRGU MUREŞ
Data: 07.06.2023.

Absolvent

Semnătura Incze.....

Kivonat

Dolgozatom témája egy átfogó applikáció készítése, amelynek témája az ételreceptek, főzés, és a gasztronómia. Mint tudjuk az ételkészítés az a téma, amelyhez mindenkinnek direkt vagy indirekt módon köze van, ezért a célcsoport nagyon széles.

Az applikáció célja, hogy könnyű hozzáférést biztosítson a legjobb és legváltozatosabb receptekhez, valamint inspirációt nyújtson az új ételek felfedezéséhez. Az applikáció intuitív felhasználói felülete és széleskörű funkciói lehetővé teszik a felhasználók számára, hogy könnyedén böngésszenek, keressenek és mentsenek el recepteket, valamint megosztanak saját kulináris alkotásait, véleményüket a közösség többi tagjával.

A választott technológia, a Flutter keretrendszer, egy viszonylag új résztvevője a piacnak, amely főleg a költség és időhatékonyusra fekteti a hangsúlyt. Mindezt a háttérben a Firebase egészíti ki, különböző szolgáltatásokkal, mint például adatbázis vagy engedélyezési folyamatok. A dolgozatban részletesen be fogom mutatni ezt a platformfüggetlen technológiát, előnyeit, hátrányait, mire kell odafigyelni ennek használata közben.

Ezen applikáció jelenleg futtatható iOS és Android rendszert használó eszközökön, illetve az admin felület bármilyen böngészőben. Ehhez ugyanazt a kódmezőt használják, ugyanazokat a függőségeket. Különböző platformokon különböző megjelenítése lehet az applikációnak, használva a natív elemeket, ezzel az átlag felhasználó észre sem veszi a különbséget egy natívan megírt applikációhoz képest.

Végül levonjuk a következtetéseket, hogy mikor érdemes és nem érdemes ezt a technológiát használni, illetve a felhasználóknak ebből milyen előnyei vagy hátrányai származhatnak az applikáció használatát illetően.

Rezumat

Tema tezei mele este dezvoltarea unei aplicații cuprinzătoare cu accent pe rețete culinare, gătit și gastronomie. Așa cum știm, gătitul este un subiect la care fiecare are o legătură directă sau indirectă, astfel că publicul țintă este foarte larg.

Scopul aplicației este de a oferi acces facil la cele mai bune și diverse rețete, precum și de a inspira utilizatorii să descopere noi feluri de mâncare. Interfața intuitivă a aplicației și gama sa extinsă de funcționalități permit utilizatorilor să navigheze, să caute și să salveze rețete în mod ușor, precum și să își împărtășească propriile creații culinare și opinii cu restul comunității.

Tehnologia aleasă, framework-ul Flutter, este o prezență relativ nouă pe piață, care punе accent pe eficiențа costurilor și a timpului. Aceasta este completată de Firebase în spatele scenei, care oferă diverse servicii precum o bază de date și procese de autentificare. În cadrul tezei, voi prezenta în detaliu această tehnologie cross-platform, discutând avantajele, dezavantajele și aspectele importante de luat în considerare în timpul utilizării sale.

Această aplicație poate fi rulată în prezent pe dispozitive care utilizează sistemele iOS și Android, precum și interfața de administrare în orice browser web. Aceasta utilizează aceeași bază de cod și dependențe. Aplicația poate avea aspecte diferite pe diferite platforme, folosind elemente native, însă utilizatorul obișnuit nu va observa nicio diferență față de o aplicație dezvoltată nativ.

În final, se vor trage concluzii cu privire la momentele în care este recomandată utilizarea acestei tehnologii și când nu, precum și avantajele și dezavantajele pe care utilizatorii le pot avea în urma utilizării aplicației.

Abstract

The topic of my thesis is to develop a comprehensive application focusing on food recipes, cooking, and gastronomy. As we know, cooking is a subject that everyone has a direct or indirect connection to, so the target audience is very broad.

The purpose of the application is to provide easy access to the best and most diverse recipes, as well as to inspire users to explore new dishes. The application's intuitive user interface and wide range of features allow users to easily browse, search, and save recipes, as well as share their own culinary creations and opinions with the rest of the community.

The chosen technology, the Flutter framework, is a relatively new player in the market that emphasizes cost and time efficiency. This is complemented by Firebase in the background, which provides various services such as a database and authentication processes. In the thesis, I will present this cross-platform technology in detail, discussing its advantages, disadvantages, and important considerations during its usage.

This application is currently runnable on devices using iOS and Android systems, as well as the admin interface on any web browser. It utilizes the same codebase and dependencies. The application may have different appearances on different platforms, making use of native elements, but the average user won't notice any difference compared to an application developed in native environment.

Finally, conclusions will be drawn regarding when it is advisable to use this technology and when it is not, as well as the advantages and disadvantages that users may derive from using the application.

Tartalomjegyzék

1. Bevezető	11
2. Elméleti megalapozás	12
2.1. Hasonló alkalmazások	12
3. Célkitűzés	13
4. Követelmény specifikáció	14
4.1. Funkcionális követelmények	14
4.2. Nem-funkcionális rendszerkövetelmények	18
4.3. Adminisztrátori követelmények	18
4.4. Rendszerkövetelmények	18
5. Programok, technológiák bemutatása	19
5.1. Publikus API	19
5.2. Saját API	19
5.2.1. Firebase Authentication	19
5.2.2. Firebase Storage	20
5.2.3. Firebase Realtime Database	20
5.3. Mobil applikáció	21
5.3.1. Dart	21
5.3.2. Flutter	21
5.4. Webes felület	22
5.5. Összehasonlítás natív megoldásokkal	22
5.5.1. Dizájn elemek	22
5.5.2. Kommunikáció a natív platformokkal	26
5.5.3. Platform specifikus konfigurációk	26
6. Fejlesztés folyamata	27
6.1. Dizájn - Figma	27
6.2. Verziókövetés - GitLab	27
7. Az alkalmazás felépítése	28
7.1. Architektúra	28
7.2. Állapot-kezelés	29
7.3. Függőség-befecskendezés	29
7.4. Adatátviteli objektum réteg	29

7.5. Lokalizáció	30
7.6. Szekvencia-diagramok	30
7.6.1. Bejelentkezés	30
7.6.2. Regisztráció	31
7.6.3. Keresés	31
7.6.4. Hozzáadás kedvencekhez	32
7.6.5. Hozzávalók kosárba tétele	32
7.6.6. Profil módosítása	33
8. Az alkalmazás működése	34
8.1. Töltőképernyő	34
8.2. Bejelentkezés	35
8.3. Regisztráció	36
8.4. Kezdő oldal	37
8.5. Profil	39
8.6. Profil módosítás	40
8.7. Keresés	41
8.8. Recept áttekintés	44
8.9. Recept részletek	45
8.9.1. Recept hozzávalók	45
8.9.2. Elkészítési lépések	46
8.10. Értékelések	47
8.11. Bevásárló kosár	48
9. Limitációk és továbbfejlesztési lehetőségek	49
9.1. Limitációk	49
9.2. Továbbfejlesztési lehetőségek	49
9.2.1. Felhasználói fórum	49
9.2.2. Kibővített keresés	50
9.2.3. Felhasználó általi recept feltöltés	50
9.2.4. Szociális funkcionálitások	50
Összefoglaló	51
Ábrák jegyzéke	52
Táblázatok jegyzéke	53
Irodalomjegyzék	54

1. fejezet

Bevezető

Az ételkészítés és a gasztronómia az emberiség történelmének alapvető részét képezi. Az ételek nem csak az életünk fenntartásához szükséges tápanyagok forrásai, hanem a kultúrák, az identitás és a közösség építésének alapjai is. A digitális korban azonban új lehetőségek nyílnak meg előttünk az ételkészítés, receptek és gasztronómia terén. Ebben a kontextusban az államvizsga dolgozatom célja egy modern és innovatív receptes applikáció, a "Kitchn" bemutatása és elemzése, amely mindenki számára hasznos tud lenni, akár van már tapasztalata gasztronómiában, akár teljesen kezdő.

Sokszor az emberek csak azért vonakodnak egy-egy új gasztronómiai kalandtól, mert az ahhoz szükséges információt túlságosan bonyolult összegyűjteni, esetleg nem elég valid vagy bevált az információ. A mai felgyorsult társadalomban a lehető legtöbb információt minimális idő és energia befektetés mellett szeretnénk megszerezni.

Ezt saját tapasztalattal alá tudom támasztani, a legtöbb ilyen applikáció vagy útmutató túl bonyolult volt a számomra kezdőként, illetve sokszor nagyon egyedi hozzávalókat sorakoztattak fel, amit nem tudtam honnan lehet beszerezni. Ezen applikációval mindenkor problémát próbáltam orvosolni, amely talán nem csak engem, de sokunkat érint.

Jelen dolgozatomban erre a szituációra dolgozok ki egy megoldást, egy mobilapplikációt, amely különböző platformokat összekötve minimalista mégis ismerős dizájn-elveket követve, intuitív felületet biztosít a felhasználók számára, hogy részesei legyenek egy közösségnek, illetve szélesítsék a tudásukat, ételkultúrájukat.

A dolgozat során részletesen bemutatom az applikáció tervezési folyamatát, beleértve a felhasználói felület és élmény tervezését, a funkcionális követelményeket, az adatmodellt és az adatbázis-kezelést. Emellett elemzem az alkalmazás fejlesztési folyamatát, a technológiai megoldásokat és a kihívásokat, amelyekkel szembesülttem az applikáció készítése során.

Továbbá fontos volt számomra, hogy a jelen gazdasági helyzetet figyelembe véve költség és időhatékony legyen a technológia, valamint a fejlesztés folyamata. Ezekben az időkben a befektetők nehezen köteleződnek el egy-egy hosszútávú projekt mellett, ilyen egy mobilapplikáció is, ezért fontos a rugalmasság és a gyors ötlet validálás.

A Kitchn receptes applikáció kifejlesztése és elemzése segítséget nyújt a gasztronómia és az innovatív technológia összekapcsolásának megértésében. Az applikáció hasznos eszköz lehet mindenknak, akik szenvedélyteljesen fordulnak az ételkészítés felé, új inspirációt és módszereket keresnek a kulináris élmények gazdagításához.

2. fejezet

Elméleti megalapozás

2.1. Hasonló alkalmazások

Több hasonló alkalmazást is megvizsgáltam a piacon, ezek elég nagy részt megegyező funkcionálisokat sorakoztatnak fel, amely érthető is, hiszen ugyanazt a célt próbálják elérni: Minél több ember használja ezeket az applikációkat.

1. **SideChef:** Ez az alkalmazás a felhasználóknak lehetőséget nyújt az ételreceptek keresésére kategóriák, összetevők vagy ételtípusok alapján. Emellett a felhasználók recepteket is megoszthatnak a különböző közösségi felületeken. Csak regisztráció után használható az applikáció, amire lehetőség van Google, Facebook, Twitter által és standard e-mail, jelszó párossal. Ami nem válik előnyére az alkalmazásnak, az a bonyolultság. A felhasználó túl van terhelve információval, bonyolultnál bonyolultabb receptek jelennek meg a kezdőoldalon, ami lehet egy plusz a gyakorlottabb szakácsoknak, de egy mínusz a kezdőknek.
2. **Tasty:** Ez az applikáció használható bejelentkezés nélkül is egy bizonyos szintig. A felhasználók számára interaktív recepteket kínál, amelyek lépésről lépésre vezetik őket az étel elkészítésében. Az alkalmazásban elérhető videók és interaktív eszközök segítik a felhasználókat az optimális eredmény elérésében. Emellett az alkalmazás tartalmaz egy vásárlólistát is, amely segít a felhasználóknak a szükséges összetevők beszerzésében.

A felhasználói felület minden esetben intuitívnek mondható, a felhasználói élmény is kielégíti az elvárásokat. Ami minden esetben megvan, az az, hogy a temérdek információt egyszerre jeleníti meg a felhasználónak, így az képes elveszni a részletekben, ahelyett, hogy egyszerű lépésenként tudna haladni. Ehhez képest az általam bemutatásra kerülő alkalmazás minimalista, lényegretörő, fő tulajdonsága az, hogy kezdőbarát. A navigáció mindenki applikáció esetében a jól bevált alsó navigációs sáv, kis ikonokkal és opcionálisan egy címmel.

3. fejezet

Célkitűzés

Azt tűztem ki célul, hogy bemutassam egy új technológia segítségével hogyan lehet tervezni és fejleszteni egy felhasználóbarát és funkcionális mobilalkalmazást, amely minimális módosításokkal egyéb platformokra is adaptálható, mindenkorral lehetővé teszi a főképp konyhai környezetben nem otthonosan mozgó felhasználók számára a receptek keresését, megosztását, elmentését, és az elkészítés lépésről-lépésre való nyomon követését. Ezen problémára már léteznek különböző megoldások, viszont én szeretném személyre szabottan, új nézőpontból megvalósítani ezt a projektet.

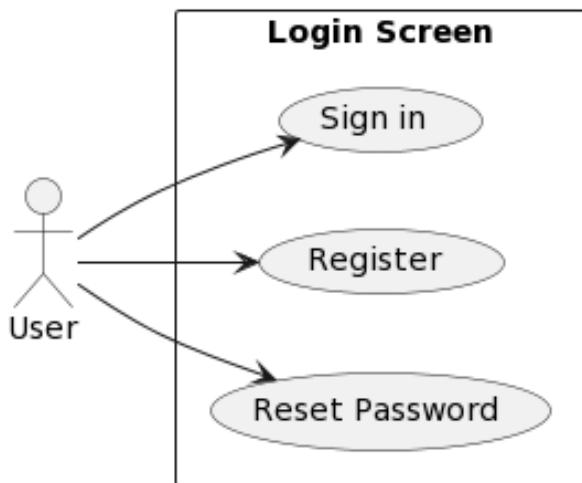
A projekt másik célja a Flutter platformfüggetlen technológia bemutatása, mint eszköz a költséghatékony, gyors fejlesztésre, kiegészítve a Firebase-szel, mint back-end felhőszolgáltalás.

4. fejezet

Követelmény specifikáció

4.1. Funkcionális követelmények

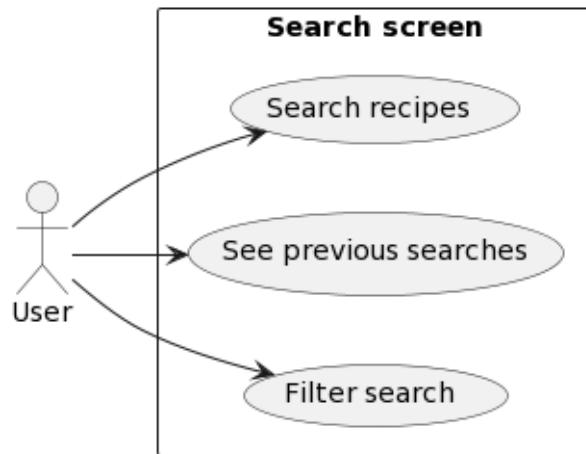
- Bejelentkezés: A felhasználók bejelentkezhetnek a már meglévő fiókjukba, amennyiben helyesek a bejelentkezéshez szükséges adatok.



4.1. ábra. Bejelentkezés oldal

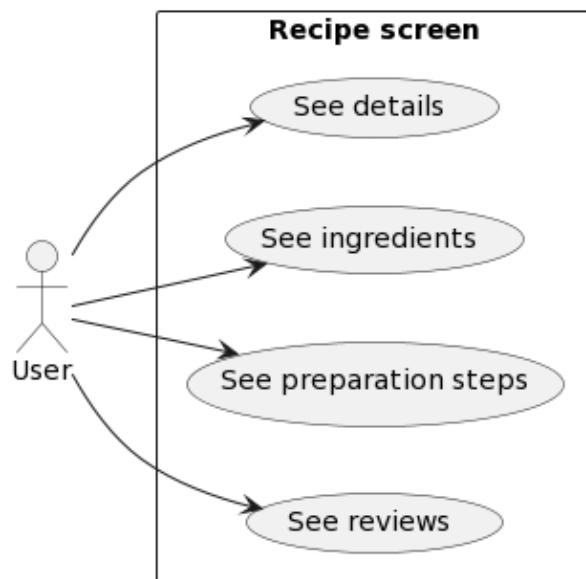
- Regisztráció: A felhasználók létrehozhatnak új fiókot, ha a regisztrációnál kért és megadott adatok helyesek. Továbbá regisztráció lehetséges meglévő Google fiókkal is, amely által automatikusan be lesz állítva a profilképük, nevük és e-mail címük az applikációban.
- Jelszó-visszaállítás: Amennyiben a felhasználó elfelejtette jelszavát, az e-mail címét megadva kap egy visszaállító linket a postafiókjába.

- Receptkeresés és böngészés: A felhasználók recepteket kereshetnek kulcsszavak, kategóriák, hozzávalók, kalóriasűrűség vagy akár elkészítési idő alapján.



4.2. ábra. Kezdő oldal

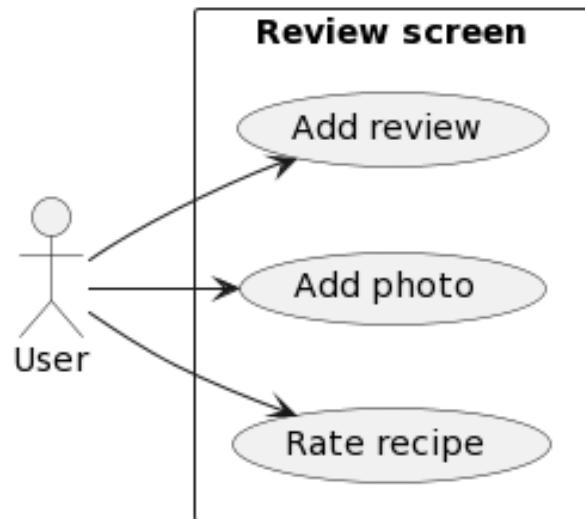
- Receptek részletei: A felhasználók megtekinthetik a receptek részletes leírását, hozzávalók listáját, mennyiségét és az elkészítési lépéseket. Továbbá, képeket is megtekinthetnek a receptkről.



4.3. ábra. Recept felvétele a kedvencek közé

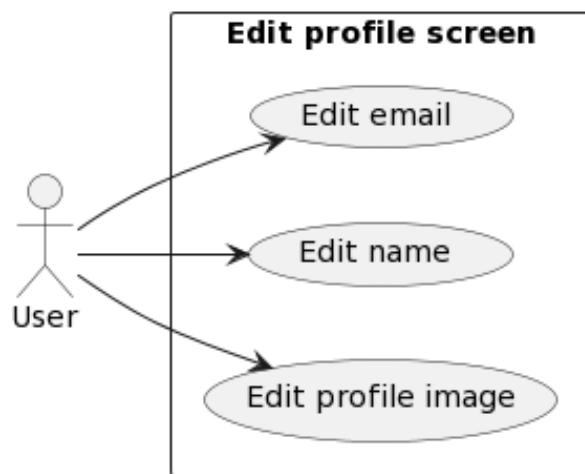
- Kedvenc receptek: A felhasználóknak lehetőségeük van elmenteni a kedvenc receptjeiket és könnyedén hozzáférni azokhoz a későbbiekben.

- Értékelések és vélemények: A felhasználók értékelhetik a recepteket egytől ötig terjedő skálán, leírhatják véleményüket azzal kapcsolatban, illetve képet is tölthetnek fel mellé. Ezek publikusak és bárki, bármikor láthatja, ki, mikor, milyen értékelést adott.



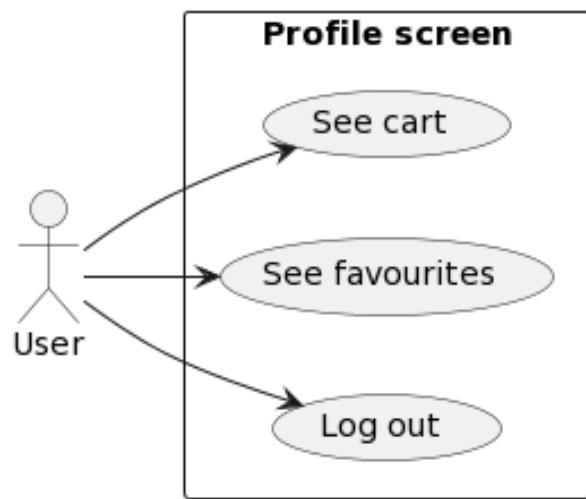
4.4. ábra. Recept értékelés

- Profilkezelés: A felhasználók szerkeszthetik profiljaikat, beleértve név, e-mail cím és profilképet.



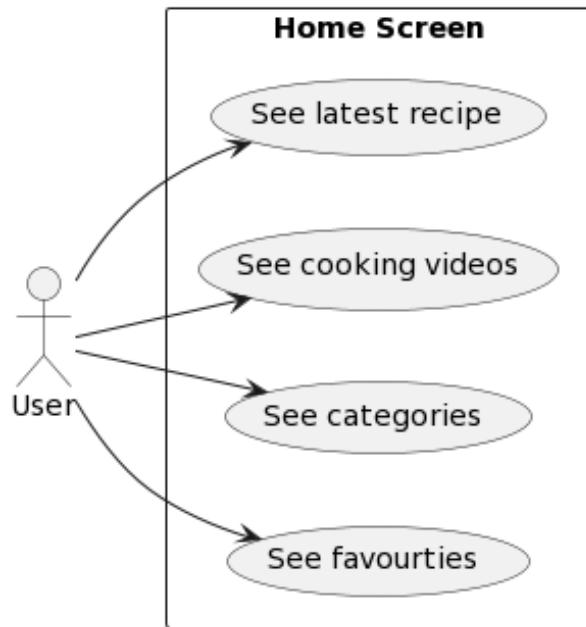
4.5. ábra. Profil módosítása

- Bevásárlókosár: A felhasználóknak van egy bevásárlókosaruk, amelybe hozzávalókat lehet tenni. minden receptnél van egy opció, hogy a szükséges hozzávalókat kerüljenek bele e a kosárba, amelyet majd a későbbiekben szerkeszthetnek is.



4.6. ábra. Profilkezelés

- Kulináris képességeket fejlesztő videók: A felhasználók megtekinthetnek különböző videókat, amely által bővíthetik konyhai képességeiket, ezen videók megtekinthetők a böngészőben vagy a Youtube applikációban.



4.7. ábra. Recept felvétele a kedvencek közé

4.2. Nem-funkcionális rendszerkövetelmények

- Teljesítmény: Az alkalmazás könnyedén, könnyedén kell fusson minden mobileszközön, például gyors betöltési és válaszidővel kell rendelkeznie, mindezt anélkül, hogy a háttérben feleslegesen sok erőforrást foglaljon le, például memóriát, sávszélességet vagy akkumulátor teljesítményt.
- Felhasználói élmény: Az alkalmazás felhasználóbarát és intuitív kell hogy legyen, könnyen kezelhető navigációs struktúrával és áttekinthető elrendezéssel, minimalista dizájnt előnyben részesítve.
- Skálázhatóság: Az alkalmazásnak képes kell lennie kezelni az esetlegesen növekvő felhasználók számát, anélkül, hogy a teljesítmény észrevehetően romlana.
- Biztonság: Az alkalmazásnak megfelelő biztonsági intézkedéseket kell tartalmaznia az adatok védelme érdekében. Ez magában foglalhatja a felhasználói hitelesítést, adatvédelmi szabályozások betartását és a biztonságos adatátvitelt.
- Kompatibilitás: Az alkalmazásnak kompatibilisnek kell lennie különböző platformokkal, operációs rendszerekkel, böngészőkkel és eszközökkel, hogy a lehető legtöbb felhasználót el lehessen érni.
- Megbízhatóság: Az alkalmazás stabil és megbízható kell legyen, minél kevesebb hibával, lefagyással és esetleges adatvesztéssel.
- Karbantarthatóság: Az alkalmazásnak könnyen karbantarthatónak kell lennie, hogy a későbbiekbén rövid idő alatt lehessen új funkciókat bevezetni, hibák kijavítani, akár új nyelvet hozzáadni.

4.3. Adminisztrátori követelmények

- Az adminisztrátorok be tudnak jelentkezni fiókukba a webes felületet használva.
- Az adminisztrátornak lehetőségük van az adatbázis szerkesztésére, receptek hozzáadására és eltávolítására.

4.4. Rendszerkövetelmények

Habár az applikáció a legújabb Flutter verziót használja, a rendszerkövetelmények így is viszonylag alacsonyak, bármelyik újonnan elérhető okostelefon könnyedén futtatja, a régebbi modellekknél a következő megszorításokra kell odafigyelni.

- Android 5+ (Lollipop)
- iOS 11+
- Szabad memória: 20-25 Megabyte
- 1 gigabyte RAM

5. fejezet

Programok, technológiák bemutatása

5.1. Publikus API

Az adatok legnagyobb része egy publikus API-ról származik, a [TheMealDB](#) oldalról. Ezeket az adatokat Python programok segítségével alakítottam át olyan formába, amilyenre szükség volt. Például mindegyik recepthez hozzá kellett rendelni egy egyedi azonosítót, illetve külön dokumentumba kinyertem a hozzávalók és a kategóriák listáját.

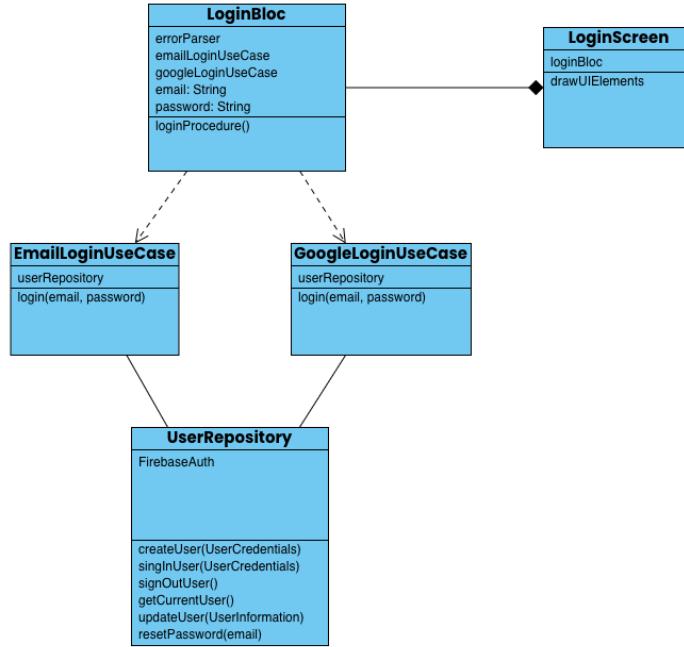
5.2. Saját API

A projekt érdemi back-end részét a Firebase felhő alapú rendszere biztosítja. A Firebase egy bizonyos szintig ingyenesen használható platform, amelyet a Google birtokol, akárcsak a Flutter technológiát. Ebből adódóan maximális a kompatibilitás a kettő között, naprakész könyvtárcsomagok fellelhetőek az összes Firebase szolgáltatásra, kiemelten jó használati útmutatókkal.

5.2.1. Firebase Authentication

A Firebase Authentication csomagja kezeli a felhasználók hitelesítését, ebbe beletartozik a regisztráció, a bejelentkezés, az e-mail cím megerősítés, jelszóváltoztatás és jelszó visszaállítás. Lehetőség van emellett különböző platformokat használva regisztrálni, mint például Google, Facebook, Twitter vagy Microsoft. Ilyen regisztráció esetében az applikáció megkapja a publikus információkat, amilyen a név, profilkép vagy e-mail cím. Hogyha az adott fiókhöz tartozó e-mail cím már regisztrálva van az applikációba, akkor a Firebase automatikusan összekapcsolja.

Az [5.1](#) ábrán az összefüggések szerepelnek a bejelentkezésért felelős osztályok között.



5.1. ábra. Bejelentkezés osztály diagram

5.2.2. Firebase Storage

A Firebase Storage főképp fájlok tárolására alkalmas felhőtároló. Jelen applikációban a felhasználók profilképeit, képeket a receptekről, illetve a receptek értékeléshez csatolt képeket tartalmazza.

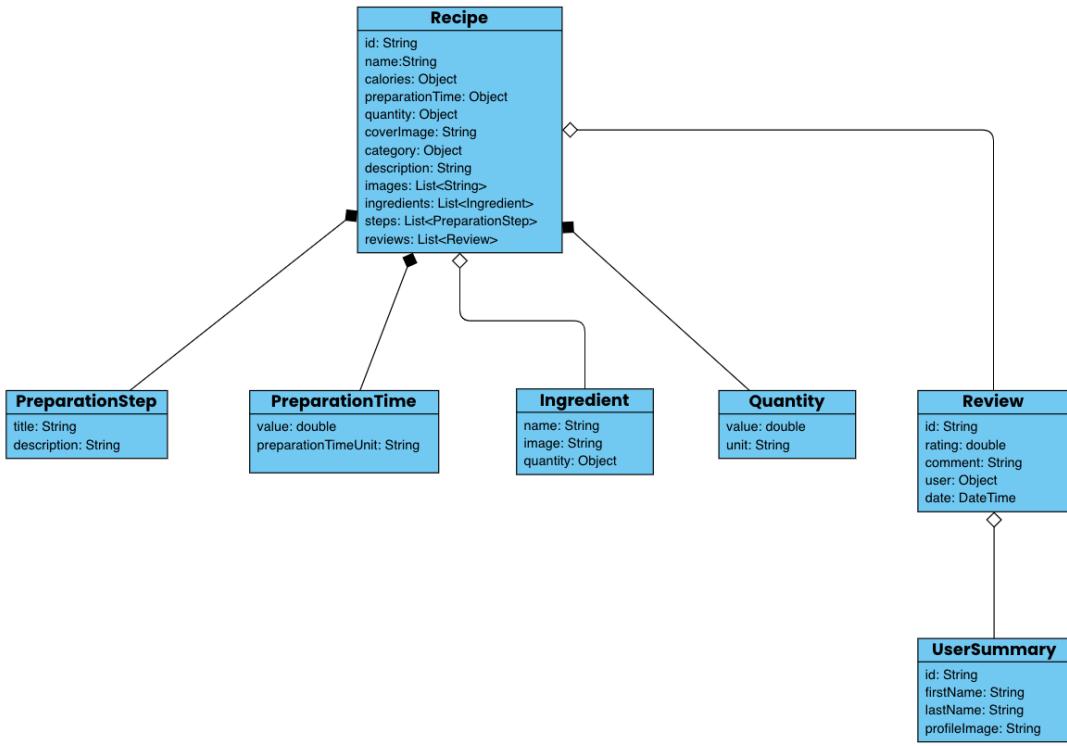
5.2.3. Firebase Realtime Database

A Firebase valós idejű adatbázisa egy felhőalapú NoSQL adatbázis. Ez a platform a fejlesztők számára egy erőteljes és skálázható megoldást nyújt valós idejű adatok tárolására és szinkronizálására több kliens között.

Egyik kulcsfontosságú előnye a valós idejű szinkronizációs képessége. Ez lehetővé teszi, hogy a változtatások azonnali módon megjelenjenek a csatlakoztatott kliensek között, ami egy zökkenőmentes és reaktív felhasználói élményt eredményez.

Továbbá, komplex biztonsági szabályokat lehet kreálni, ezek pedig lehetővé teszik a finomhangolt olvasási és írási engedélyek meghatározását, akár felhasználói hitelesítés alapján, együttműködve a Firebase Authentication-nel így biztosítva az adatvédelmet. Nem csak biztonsági szabályokat, hanem például indexelést is beállíthatunk a tábláinkra, hogy melyik táblát melyik mezők alapján indexelje, a gyorsabb adatelérés érdekében.

Az 5.2 ábrán láthatjuk az összefüggéseket a recept, a felhasználó, a hozzávaló és az értékelés osztályok között.



5.2. ábra. Recept osztály diagram

5.3. Mobil applikáció

A projekt fő része a mobil platformokat célozza meg, mint Android és iOS, ezeken a felületeken érik el a felhasználók az applikációt.

5.3.1. Dart

A Dart programozási nyelv egy modern, hatékony és rugalmas nyelv, amelyet a Google fejlesztett ki. A nyelv kifejezetten a Flutter keretrendszer támogatására lett tervezve, amely lehetővé teszi az egy kódázból több platformra való fejlesztést, mint például Android, iOS, Windows, Web és MacOS. Kompilálható gépi kódba, JavaScript-be vagy WebAssembly-be.

5.3.2. Flutter

A Flutter egy nyílt forráskódú keretrendszer, amelyet szintén a Google fejlesztett ki, 2017-ben jelent meg az első verziója. Egy felhasználói felületre összpontosító rendszer, amely lehetővé teszi a több platformra való fejlesztést. A hivatalos dokumentációja egyéb keretrendserekhez képest nagyon részletes, interaktív, amely segít a fejlesztőknek hamar elsajtítani a Fluttert, ahogy ezt a [MAA⁺¹⁹] tanulmány is alátámasztja.

5.4. Webes felület

A webes felület az adminisztrátori tevékenységekre lettek fejlesztve. Ugyanazt a kód-bázist használva ezt relatív egyszerű volt hozzáadni utólag, nem kellett külön definiálni hálózati réteget, osztályokat, de még kezelőfelületi dizájn elemeket is csak minimálisan. Bármilyen böngészőben futtatható.

5.5. Összehasonlítás natív megoldásokkal

A Flutter jó választás lehet olyankor, amikor gyorsan szeretnénk egy ötletet validálni vagy akár piacra dobni, mivel a fejlesztési idő a natív platformokhoz képest másfél-szer gyorsabb és költséghatékonyabb. Két fejlesztői csapat helyett, csak egyre van szükség, egy kód-bázist kell karban tartani. Teljesítményben minimálisan marad alul a natív megoldásokkal szemben, amilyen a Kotlin vagy a Swift, viszont a mai hardver annyira fejlett, hogy ez használat során nem vehető észre, ezt alátámasztja [Dah19] tanulmány is. Továbbá legyszerűsödik a hibakeresés és tesztelés folyamata is, ahogy ez a [Wu18] tanulmányban is olvasható.

Sok nagyszabású applikáció is használ már Fluttert, mint például Google Classroom, vagy akár a Google Pay, amelyet világszerte 70 millió felhasználó használ. Érdekesség lehet, hogy a Toyota is használ Fluttert az autói fedélzeti rendszeréhez, illetve a BMW mobilapplikációja is így készült.

Limitációk természetesen léteznek, például nem kompatibilis sok esetben olyan Apple termékekkel, mint az Apple TV, vagy Apple Watch, amelyet a Swift könnyedén kezel. Még egy hátrány, hogy viszonylag kevés fejlesztő használ régóta Fluttert, ezért nehéz lehet egy nagyobb professzionális csapatot találni a befektetőknek.

5.1. táblázat. Összehasonlítás a Flutter és a natív platformok között

Aspektus	Flutter	Natív Platformok
Keresztplatformosság	Igen	Nem (Platform-specifikus)
UI fejlesztés	Egyetlen kódalap	Platform-specifikus
Teljesítmény	Jó	Platform-optimalizált
Fejlesztési sebesség	Gyors	Közepes
Közösségi támogatás	Növekvő	Kialakult
Hozzáférés az API-khoz	Bővítményeken keresztül	Natív SDK-k
Eszközök és ökoszisztemája	Kiterjedt eszközök és widgetek	Platform-specifikus
Tanulási görbe	Közepes	Platform-specifikus
Tesztelés	Hatókony	Komplikáltabb

5.5.1. Dizájn elemek

Ha már több platformról van szó, meg kell említeni az ezek közötti felhasználói felület különbségeket. Amíg az iOS Cupertino stílusú elemeket használ, addig az Android a Material Design vonalat követi. Az 5.1-es ábrán megfigyelhető több különbség is:

- A fejlécben található cím elhelyezése.

- A fejlécben található navigációs gomb.
- A felugró ablak formája, azon belül a gomb és a szöveg elhelyezése, orientálása.



5.3. ábra. Összehasonlítva a Material Design és Cupertino Design

Ennek a dinamikus megoldásnak az implementációja alább látható. Bárhol egy ilyen típusú ablakot kell megjelenítenem, csak használom a *PlatformDialog* osztályt, amely automatikusan a kellő dizájnt téríti vissza.

```
class PlatformDialog {
  /**
   * Presents a dialog based on the underlying platform. Works with text
   * content only and simple, text based buttons.
   * For iOS, Cupertino themed dialog is presented, and for the rest,
   * material is used.
   */
   * The dialog automatically closes after an action has been tapped.
  static Future<dynamic> presentDialog(
    BuildContext context, {
      required String title,
      String? description,
      /// placed on the right side
      required DialogAction positiveAction,
      /// placed on the left side
      DialogAction? negativeAction,
      bool dismissible = true,
    }) {
    dismiss(action) async {
      Navigator.of(context).pop();
      // wait a bit for the dialog to close
      await Future.delayed(const Duration(milliseconds: 200));
    }
  }
}
```

```

        action();
    }

    if (Platform.isIOS) {
        return showCupertinoDialog(
            barrierDismissible: dismissible,
            context: context,
            builder: (_) {
                final actions = <CupertinoDialogAction>[];
                if (negativeAction != null) {
                    actions.add(
                        CupertinoDialogAction(
                            onPressed: () => dismiss(negativeAction.onPressed),
                            isDestructiveAction: negativeAction.isDestructive,
                            child: negativeAction.child,
                        ),
                    );
                }
                actions.add(
                    CupertinoDialogAction(
                        onPressed: () => dismiss(positiveAction.onPressed),
                        isDestructiveAction: positiveAction.isDestructive,
                        child: positiveAction.child,
                    ),
                );
            }

            return CupertinoAlertDialog(
                title: Text(title),
                content: description != null ? Text(description) : Container(),
                actions: actions,
            );
        );
    } else {
        return showDialog(
            barrierDismissible: dismissible,
            context: context,
            builder: (_) {
                final actions = <TextButton>[];
                if (negativeAction != null) {
                    actions.add(
                        TextButton(
                            child: negativeAction.child,
                            onPressed: () => dismiss(negativeAction.onPressed),
                        ),
                    );
                }
                actions.add(
                    TextButton(
                        child: positiveAction.child,
                    ),
                );
            }
        );
    }
}

```

```
        onPressed: () => dismiss(positiveAction.onPressed),
    ),
);
}

return AlertDialog(
    backgroundColor: AppColors.background,
    titleTextStyle:
        Theme.of(context).textTheme.bodyText1?.copyWith(color:
            AppColors.textPrimary),
    contentTextStyle:
        Theme.of(context).textTheme.bodyText2?.copyWith(color:
            AppColors.textPrimary),
    title: Text(title),
    content: description != null ? Text(description) : Container(),
    actions: actions,
    // do not stretch dialog vertically
    scrollable: true,
);
});
}
}
}
```

5.5.2. Kommunikáció a natív platformokkal

Vannak olyan esetek, amikor bizonyos limitációk miatt natívan kell megoldani egy problémát, ez azt jelenti, hogy az adott kódrészletet például Kotlinban kell megírni és ezt valahogy hozzákapcsolni. Ilyen például a szöveg felolvasás, applikáción belüli vásárlás vagy éppen egy töltőképernyő beállítása. Ennek megvalósítására használnak úgynevezett platform csatornákat, ami tulajdonképpen egy exportált natív függvény, amit elér a Flutter, úgy működik, mint egy csővezeték. Itt lehet a kommunikáció egy vagy kétirányú is.

5.5.3. Platform specifikus konfigurációk

Akármennyire is szeretnénk egy ütésre két legyet lecsapni, bizonyos teendőket kötelező módon natívan kell, illetve lehetve csak megoldani. Ilyen például az applikáció nevének vagy ikonjának beállítása. Léteznek erre harmadik féltől származó könyvtárcsomagok, viszont ezek csak a projekt komplexitását növelik, új függőségek bevezetésével és későbbi menedzselésével.

Célzottan be kell állítani, hogy melyik verziójú Android, illetve iOS rendszer a cél, amire fejlesztünk, itt figyelembe kell venni az általunk használt könyvtárcsomagokat, ugyanis soknak van egy minimum operációs szint követelménye.

Érdemes figyelembe venni, hogy az Apple termékeire fejlesztve jelen pillanatban csak úgy lehet bizonyos funkcionálitásokat implementálni, hogyha van előfizetésünk apple fejlesztői felhasználóra, egy gyakori példa erre az applikáción kívüli értesítések.

Ezen projekt esetében a Firebase is kiemelt szerepet játszik, ennek a konfigurációja a Firebase webes felületén kezdődik, ahol regisztrálni kell az applikációt minden platformra külön. A regisztráció után generálódik egy konfigurációs fájl, amit be kell helyezni az Android és iOS mappákba, illetve a fő HTML fájlba web esetében.

6. fejezet

Fejlesztés folyamata

6.1. Dizájn - Figma

Az applikáció életútja egy dizájn elkészítésével kezdődött, ami a Figma interfész tervezőben történt, figyelembe véve a modern trendeket, mint ahogy a [Nil09] tanulmány is leírja. Itt kerültek meghatározásra az applikáció szabvány méretei, például az ikonok, képek mérete, a betűtípus, a betűk mérete, a térközök és így tovább. A szövegek a Telegraf stílust használják, különböző színekben és méretekben.

6.2. Verziókövetés - GitLab

Az alkalmazás fejlesztése lépéseként történt, amennyire csak lehet Agile metodológiát követve. Ahhoz, hogy ez zökkenőmentesen menjen, verziókövetést használtam, a git szoftverforráskód-kezelő rendszert, azon belül is a GitLab-ot, mint adattár hosting szolgáltatás. Egy fő ága volt a projektnek, amelyből származtattam külön ágat minden funkcionálisra. Amikor egy funkcionális elkészült, ezt visszaillesztettem a fő ágba, majd jött az integrációs tesztelés.



6.1. ábra. Git Repository aktivitás

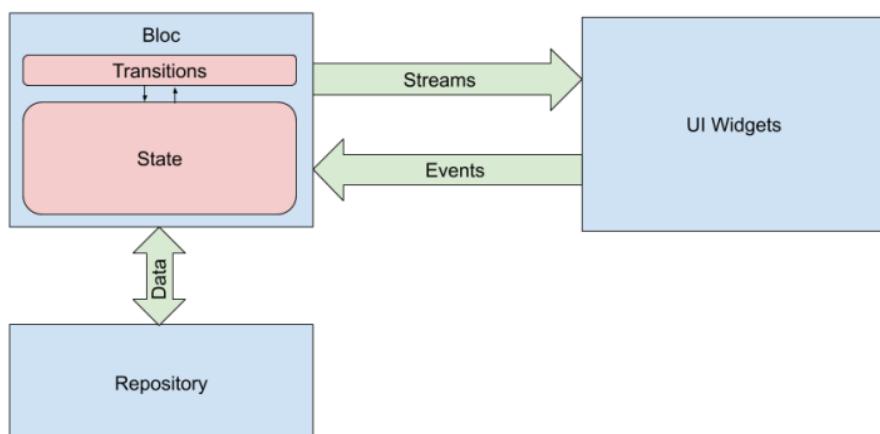
7. fejezet

Az alkalmazás felépítése

Az alkalmazás több rétegből áll, amelyek kommunikálnak egymással, de nem függnek egymástól. minden rétegnek jól elkülönülő szerepe van, így a karbantartás és hibakeresés egyszerű.

7.1. Architektúra

Az alkalmazás tervezésénél első sorban próbáltam arra összpontosítani, hogy jól különálló, egyenként tesztelhető komponensekből álljon össze a végső termék. A felhasználói felület csomag magába foglalja a vizuális függőségeket (képek, ikonok), témákat (világos, sötét), ami még három alegységre van felosztva, ezek az applikáció szín konstansai, a méret konstansok és a konkrét téma implementációk. Továbbá az oroszlánrész a nézetek (Views) csomagban található. Ez tartalmazza magát a design fájlt (view), a BLoC-ot (Business Logic Component), a különböző állapotokat (Bloc States), illetve egyéb újrahasznosítható widgeteket. Az alábbi 7.1 ábrán ezt jobban megfigyelhetjük.



7.1. ábra. BLoC működése

7.2. Állapot-kezelés

A BLoC leginkább az MVVM (Model-View-ViewModel) programtervezési mintához hasonlítható. Röviden a BloC mondja meg, hogy milyen adatok kerüljenek megjelenítésre, ő tartalmaz mindenféle kezelési logikát is ezekhez, a View pedig konkrétan megjeleníti az adatokat a képernyőn. A BloC tartalmaz egy Streamet, amely állapotokat hoz létre. A View minden állapot esetében tudja, hogy mit kell megjelenítsen, ez a gerince a felhasználói felületnek. Példa esetekre: Hiba, Töltődés, Több elem betöltése (pagination), Adat, Kezdeti állapot. Ezek az esetek implementáció szintén osztályok, amelyek tartalmazzák a megjeleníteni kívánt adatot, így a View még a BloC-kal sem áll közvetlen összekötésben, ő csak az állapotokhoz tartozó adatokról tud, hogy az miként kerül oda, arról nem.

7.1. táblázat. Összehasonlítás az MVVM és a BLoC között

Aspektus	MVVM	BLoC
Koncepció	Modell-nézet-keretmunka	Üzleti logika komponens
Kommunikáció	Adatkötés	Eseményalapú kommunikáció
Felépítés	Nézet, Modell, Nézetmodell	Blokkok, Események, Állapot
Tesztelhetőség	Jó	Jó
Kódismétlés	Kevés	Kevés
Tanulási görbe	Könnyű	Mérsékelt
Közösségi támogatás	Széles	Széles
Használat	Nagy projektek	Kisebb projektek

7.3. Függőség-befecskendezés

A View és a BLoC összekötését tárgyaltuk, viszont még valamilyen módon össze kell kötni a BLoC-ot az adat réteggel. A BLoC nem fér hozzá direkt módon az adat layerhez, hanem csak Use-Caseket tárol, tehát tudja, milyen funkcionalitásra van szüksége. Ezek a Use-Casek viszont nem a BLoCban lesznek létrehozva minden egyes alkalommal, hanem a konstruktörben megkapja már a BLoC. Erre a legelterjedt csomag a GetIt package. Ez tulajdonképpen egy service-locator. mindenféle létező dependenciát a service locatorban kell regisztrálni az applikáció betöltésekor. Ide tartoznak a servicek, a Use-Casek, a BLoC-ok, Repositoryk és egyéb függőségek is. A service locatorban kerül meghatározásra, hogy egy osztály például singleton-e vagy sem, és ő is hozza létre az egyedeiket, amelyek majd továbbkerülnek az osztályok konstruktoraiba.

A Use-Casek tartalmazzák az üzleti logika részét az applikációnak, ők tudják, hogy egy funkcionalitás végrehajtásához milyen lépések és függőségek szükségesek.

7.4. Adatátviteli objektum réteg

Az adatátviteli objektum réteg, angolul DTO (Data Transfer Object) a köztes réteg az API kérések és a Use-Case között. Erre azért van szükség, hogy ne fedjük fel az applikáción belső implementációit, illetve csak azokat az értékeket használjuk, tároljuk el a

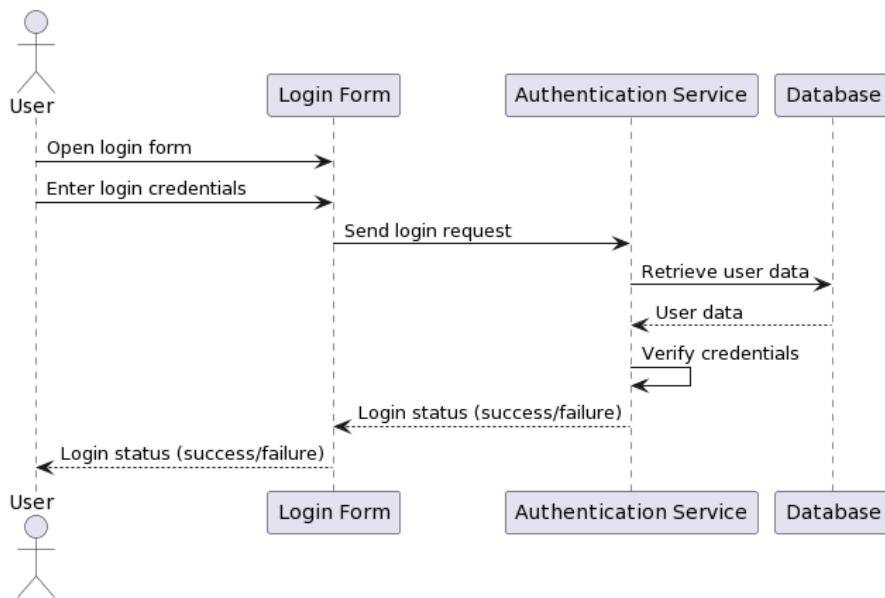
szervertől, amelyeket valóban használunk is. NoSQL adatbázist használva a duplikációk elkerülése érdekében legtöbb esetben csak azonosítót tárolunk egy adatról, ha az összefüggésben egy áll egy másik adattaggal is, a DTO réteg pedig ezt is megoldja, hogy az azonosító alapján lekéri a konkrét adattagot, amire szükség van.

7.5. Lokalizáció

Lokalizációs lehetőséget is építettem be az az alkalmazásba, amely lehetővé teszi, hogy különböző nyelvekre legyenek lefordítva a szövegek. Tulajdonképpen minden megjelenítendő szövegnek van egy kulcsa, amelyhez hozzá tudunk rendelni értéket, és ezeket könnyedén lecserélhetjük egy másik nyelven adott értékre. Jelenleg csak az angol nyelv támogatott.

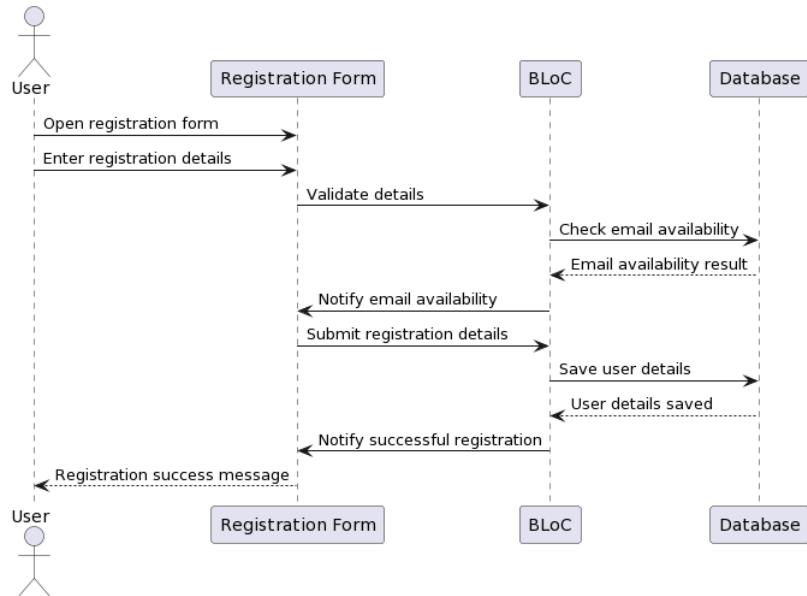
7.6. Szekvencia-diagramok

7.6.1. Bejelentkezés



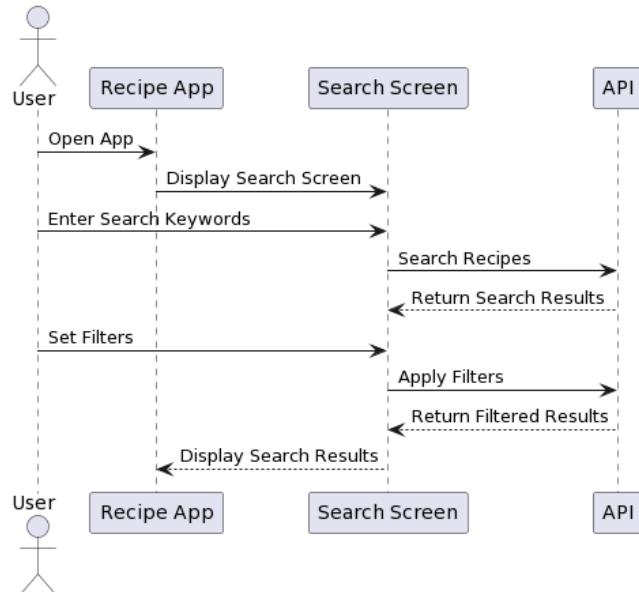
7.2. ábra. Bejelentkezés szekvencia-diagram

7.6.2. Regisztráció



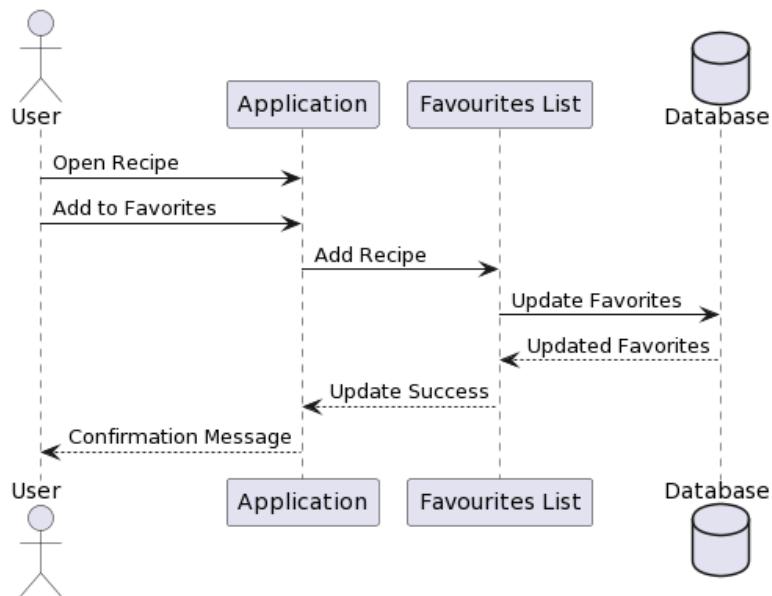
7.3. ábra. Regisztráció szekvencia-diagram

7.6.3. Keresés



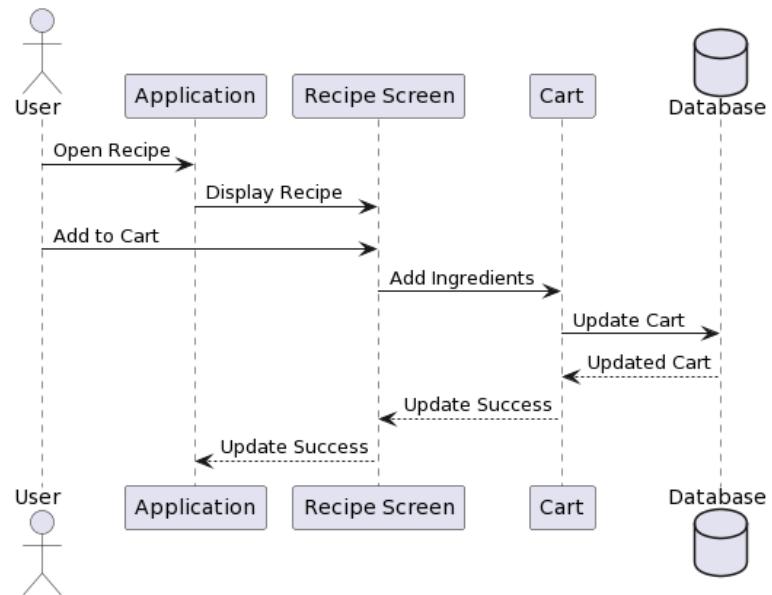
7.4. ábra. Receptek keresése szekvencia-diagram

7.6.4. Hozzáadás kedvencekhez



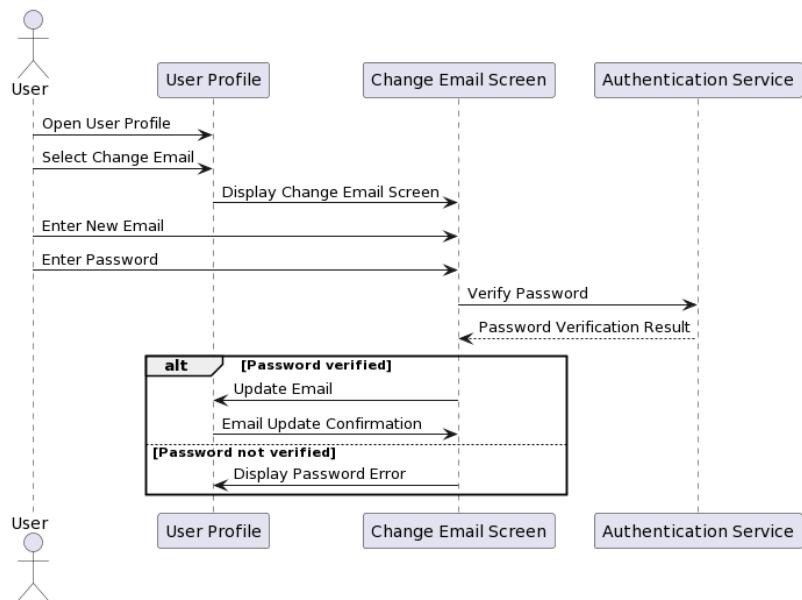
7.5. ábra. Recept mentése szekvencia-diagram

7.6.5. Hozzávalók kosárba tétele



7.6. ábra. Hozzávalók mentése szekvencia-diagram

7.6.6. Profil módosítása



7.7. ábra. Profil módosítás szekvencia-diagram

8. fejezet

Az alkalmazás működése

Az alábbiakban részletesen lesz tárgyalva az applikáció felhasználói felülete és működése, képernyőképekkel kiegészítve.

8.1. Töltőképernyő

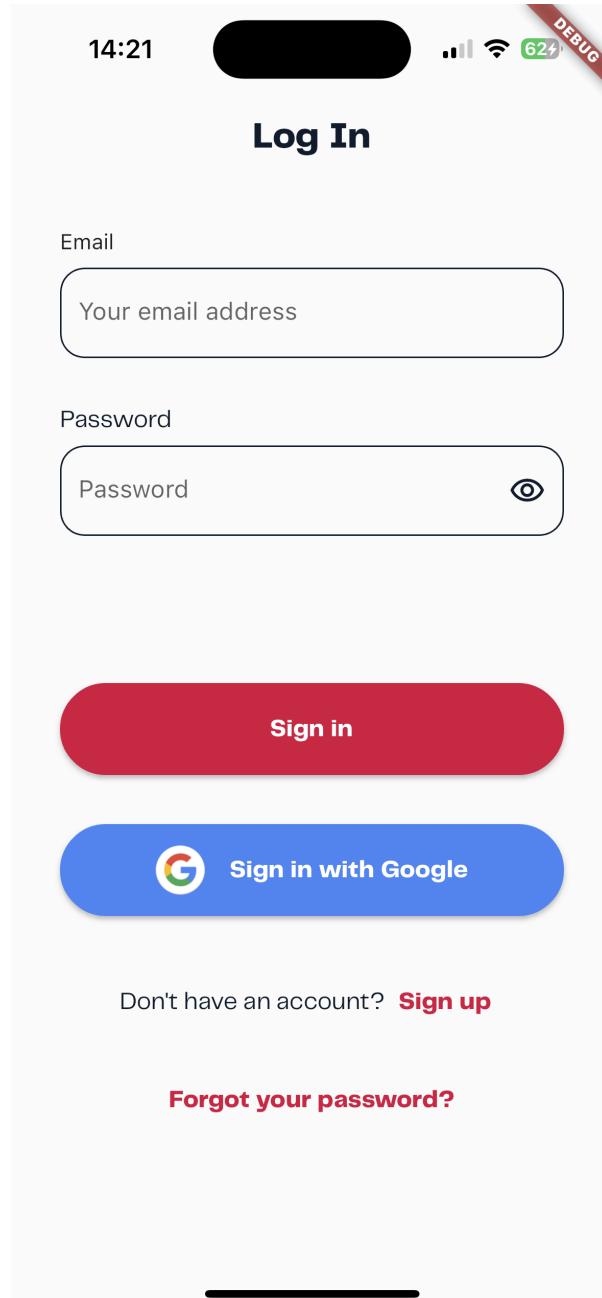
Amíg az applikáció betölti az erőforrásokat, addig egy töltőképernyőt jelenítünk meg, ez Android-on az applikáció ikonja, iOS rendszeren pedig egy kép, amit a 8.1 ábrán láthatunk.



8.1. ábra. Töltőképernyő

8.2. Bejelentkezés

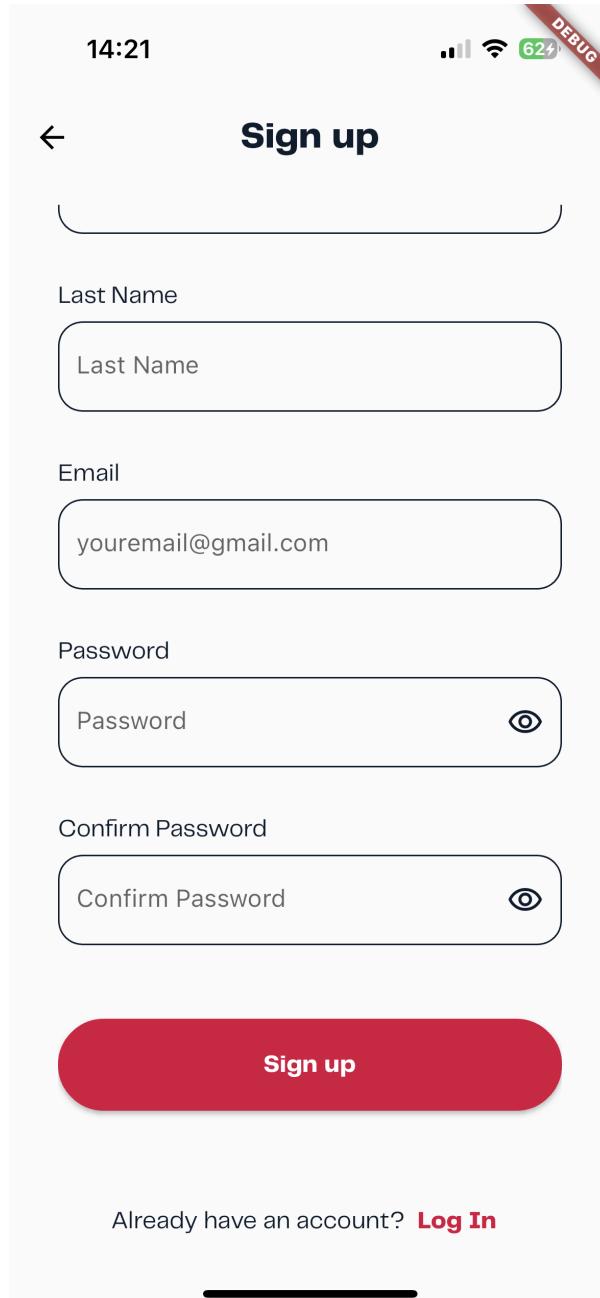
A bejelentkezés oldalon több funkció is van. Ha van már fiókunk, akkor bejelentkezünk, akár Google fiókkal is, illetve innen navigálhatunk a regisztrációs oldalra, vagy a jelszó-visszaállítás oldalra.



8.2. ábra. Bejelentkezés képernyő

8.3. Regisztráció

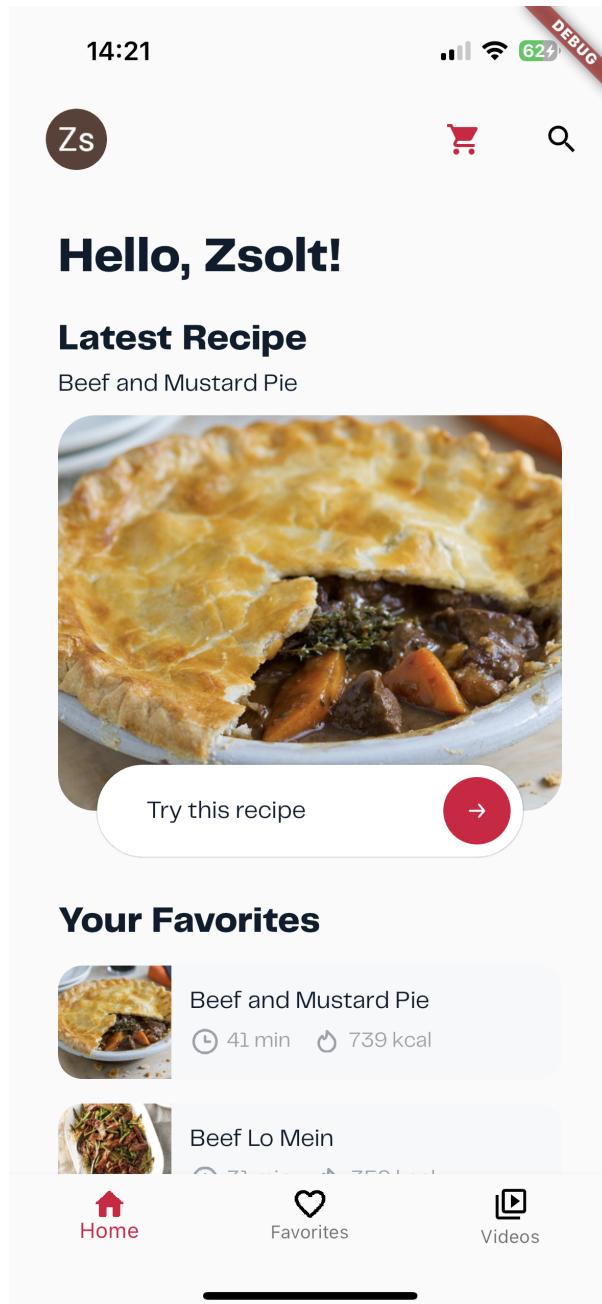
A regisztrációs oldalon regisztrálni tudunk, amennyiben a beírt adatok megfelelnek az elvárásoknak, mint például helyes e-mail cím és olyan jelszó, amely minimum 8 karakter hosszú, tartalmaz különleges jeleket, számot, kis- és nagybetűt.



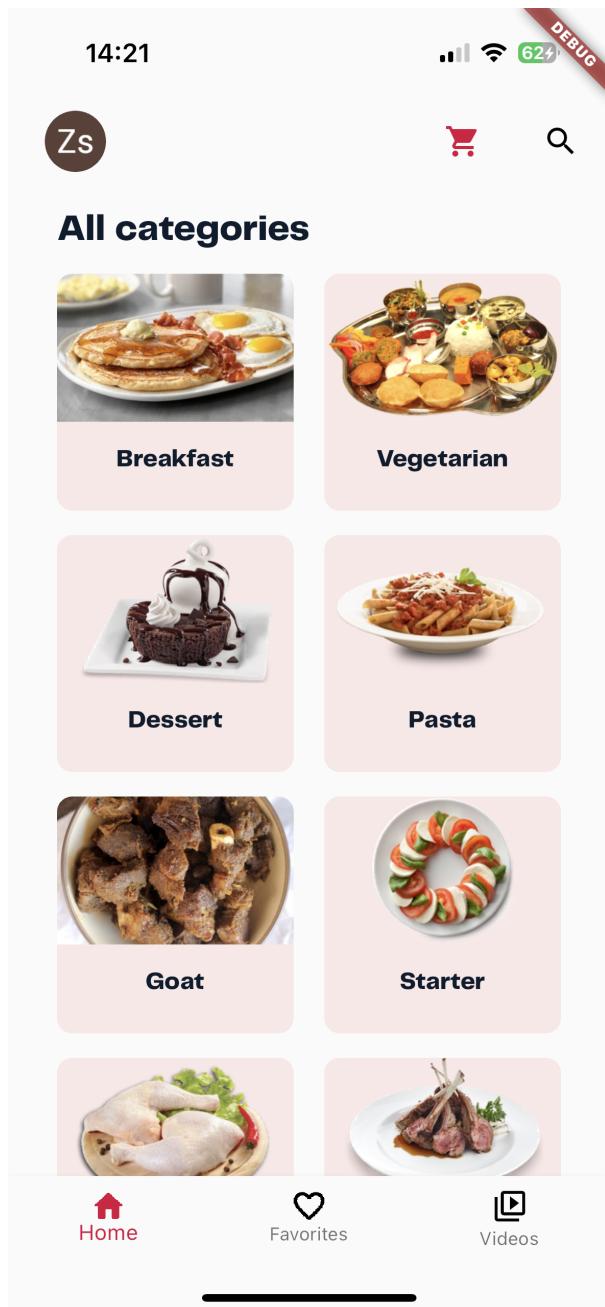
8.3. ábra. Regisztrációs képernyő

8.4. Kezdő oldal

A kezdőoldalra lépve több minden láthatunk. A felső sávból tudunk navigálni a bevásárló kosárba, a kereséshez, és a profil képernyőre is. A görgethető részen megjelenik a legutóbbi recept, a kedvencekből maximum öt darab recept, illetve a kategóriák.



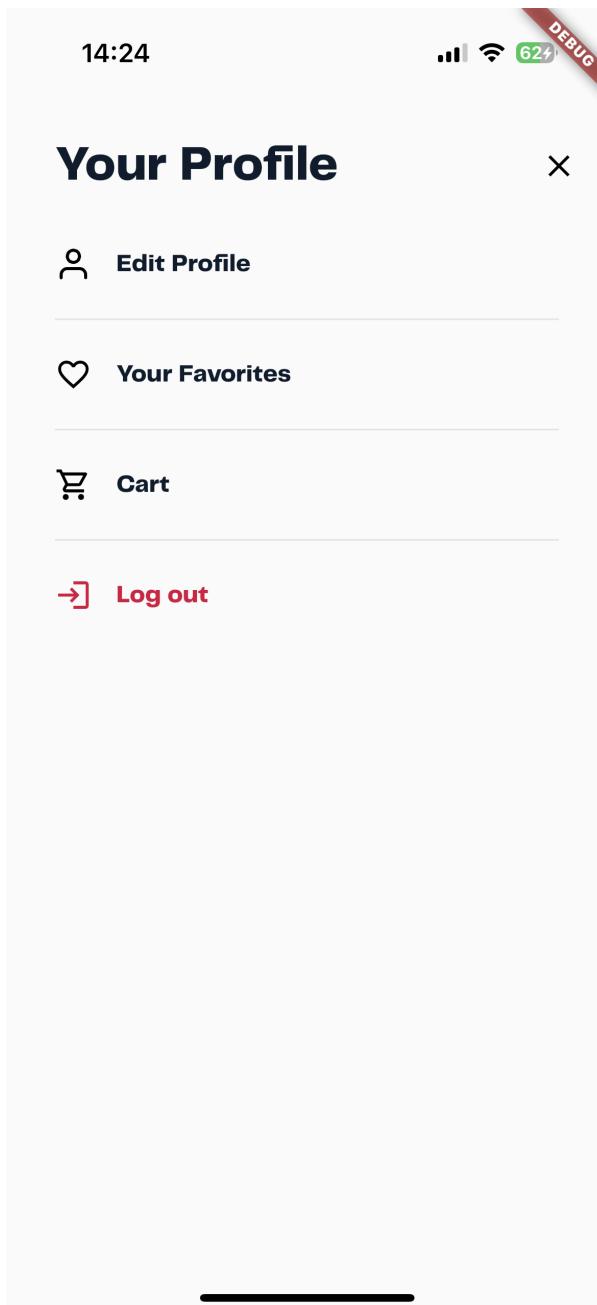
8.4. ábra. Kezdőképernyő



8.5. ábra. Kategóriák a kezdőképernyőn

8.5. Profil

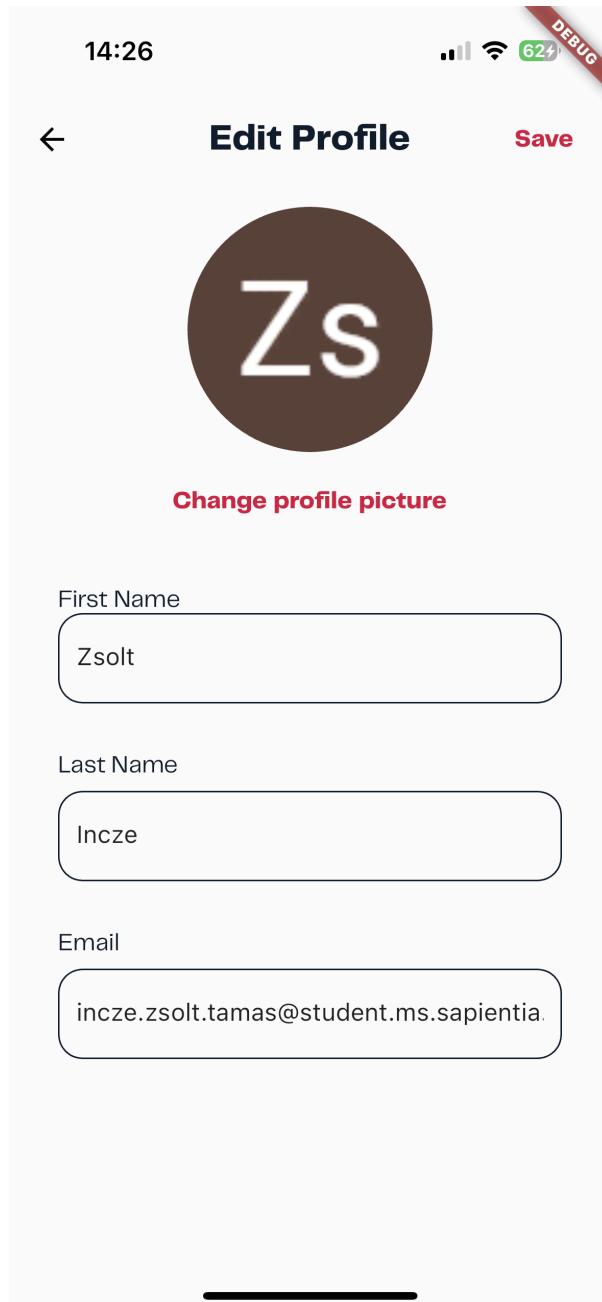
Erről az oldalról tudunk tovább navigálni a profil szerkesztéséhez, a bevásárló kosárhoz, a kedvencek oldalra, illetve ki tudunk jelentkezni.



8.6. ábra. Profil képernyő

8.6. Profil módosítás

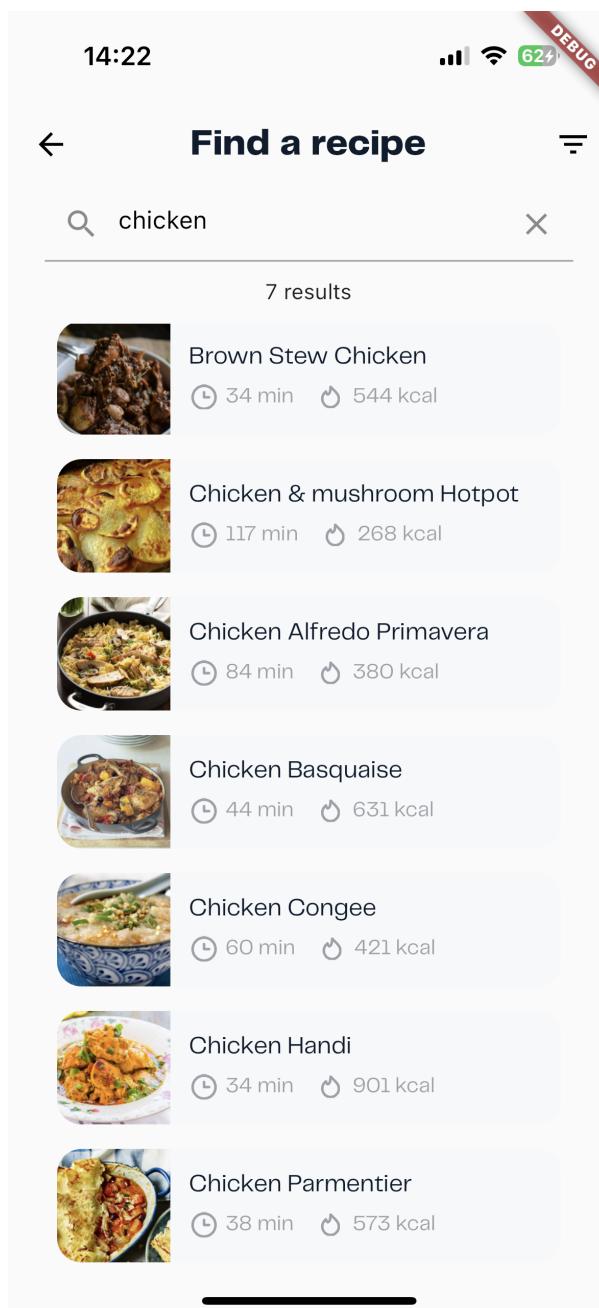
Ezen az oldalon tudjuk módosítani felhasználó adatainkat, mint például profilkép, név vagy e-mail cím. Egy fontos dolog, hogy e-mail cím változtatás esetén újra meg kell adnunk a jelszavunkat, ha ez sikeres, akkor egy e-mailt is fogunk kapni az új postafiókba.



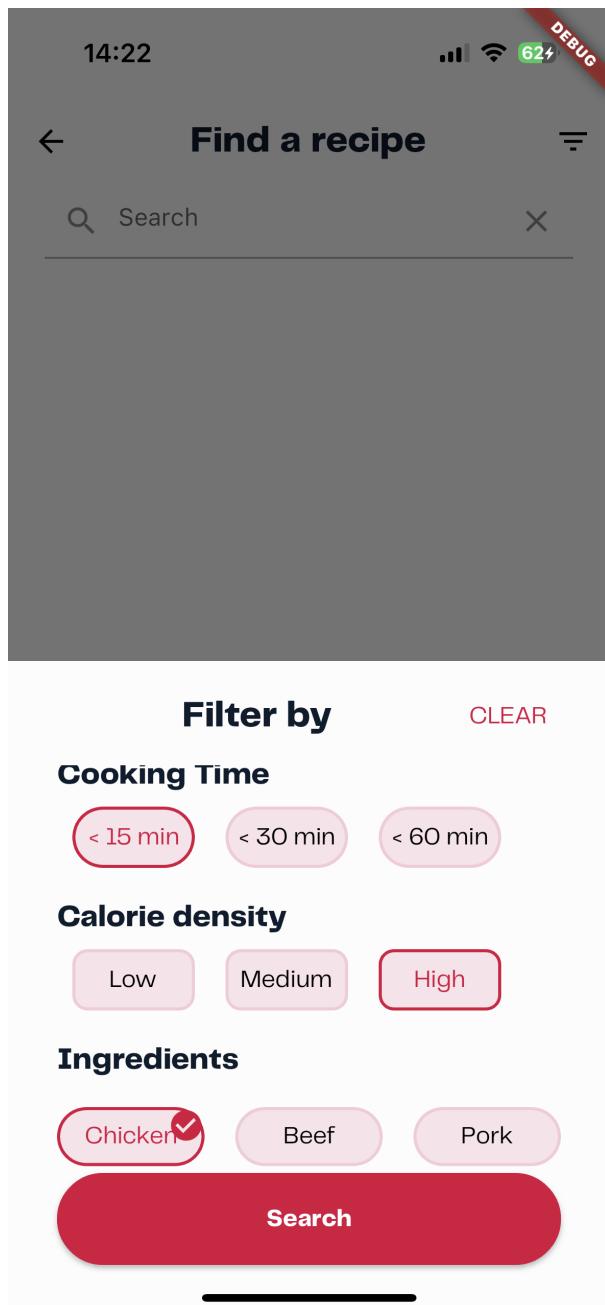
8.7. ábra. Profil módosítása

8.7. Keresés

A keresés oldalon legfelül beírhatjuk, hogy milyen receptet keresünk, illetve látjuk az utolsó öt keresési előzményünket, amelyek lokálisan vannak tárolva a készüléken. Továbbá különböző szűrőket tudunk beállítani, ami alapján keresünk, például elkészítési idő, kalóriasűrűség és hozzávalók, ez utóbbi a legtöbb hasonló applikációban benne van, ahogy leírja ezt a [PSU18] tanulmány is. Fontos itt megjegyezni, hogy egyszerre maximum tizenöt receptet kérünk le a szerverről, és ahogy görgetünk lefelé, mielőtt a lista végére érnénk indul a következő API kérés, hogy töltön be még több receptet. Így feleslegesen nem terhelődik le az eszköz.



8.8. ábra. Keresés képernyő



8.9. ábra. Szűrők beállítása

A következő kódrészlet a recept keresés képernyőn oldja meg az oldalakra bontást, tehát ahogy görgetünk lefelé, úgy dinamikusan egyre több receptet töltön be, ezzel is javítva a teljesítményt.

```
Future<void> loadMore() async {
  if (_hasNextPage == true && _isLoadingMore == false) {
    _isLoadingMore = true;
    _pageCount += 1;
    _stateController.add(
      DataBlocState(
```

```

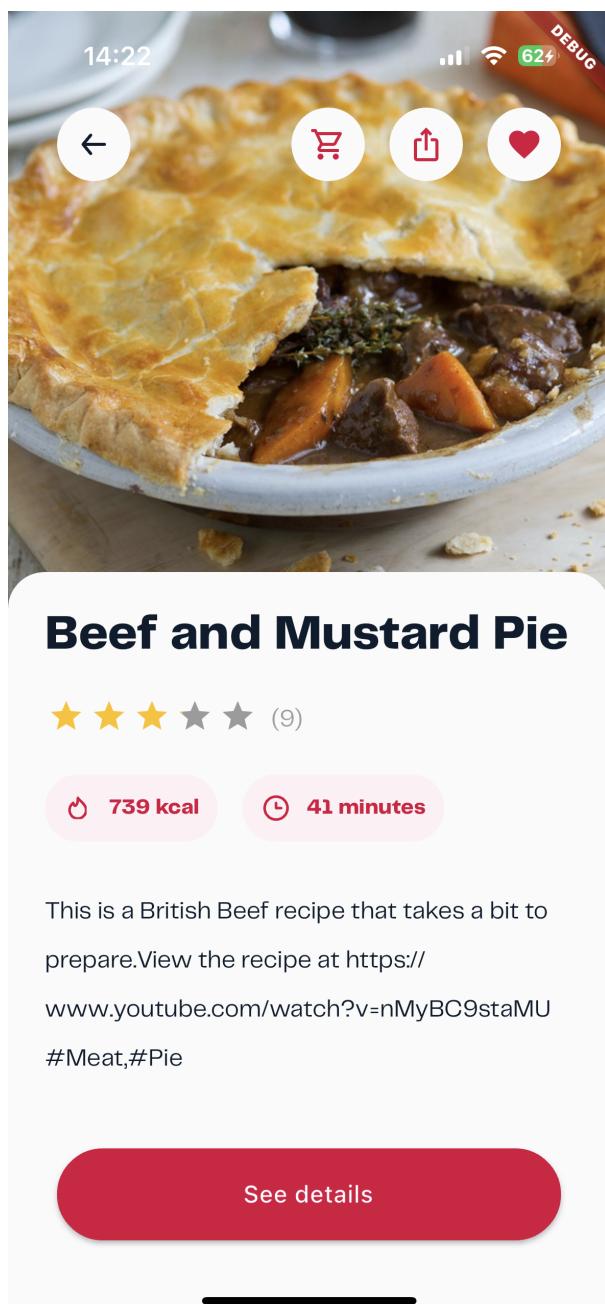
        recipes: _recipes,
        isLoadMoreRunning: _isLoadingMore,
        loadMoreException: null,
    ),
);
final result = await getFilteredRecipesUseCase.call(
    _latestQuery!,
    page: _pageCount,
    calorie: _caloriesController.value,
    cookingTime: _cookingTimeController.value,
    ingredients: _ingredientsController.value,
);

result.when(
    success: (tempRecipes) {
        if (tempRecipes.length < _perPage) {
            _hasNextPage = false;
        }
        _recipes.addAll(tempRecipes);
        _isLoadingMore = false;
        _stateController.add(
            DataBlocState(
                recipes: _recipes,
                isLoadMoreRunning: _isLoadingMore,
                loadMoreException: null,
            ),
        );
    },
    error: (error) {
        _isLoadingMore = false;
        final appError = _errorParser.parseErrorToAppError(
            context,
            dataLayerException: error,
            retry: () {
                loadMore();
            },
        );
        _stateController.add(ErrorBlocState(appError));
    },
),
);
}
}

```

8.8. Recept áttekintés

Ezen a képernyőn láthatunk egy összefoglalót a receptről. Itt a bevásárló kosár ikonra kattintva hozzáadódnak a recept hozzávalói a kosarunkhoz. A szív ikonra kattintva pedig a recept bekerül a kedvencek közé. Egy harmadik ikonnal pedig meg tudjuk osztani a receptet ismerőseinkkel a különböző szociális felületeken, vagy akár SMS-ben is. Láthatunk a receptről képet, leírást, és alapvető információkat, mint például kalória vagy elkészítési idő. A csillag ikonokra kattintva jutunk az értékelés képernyőre.

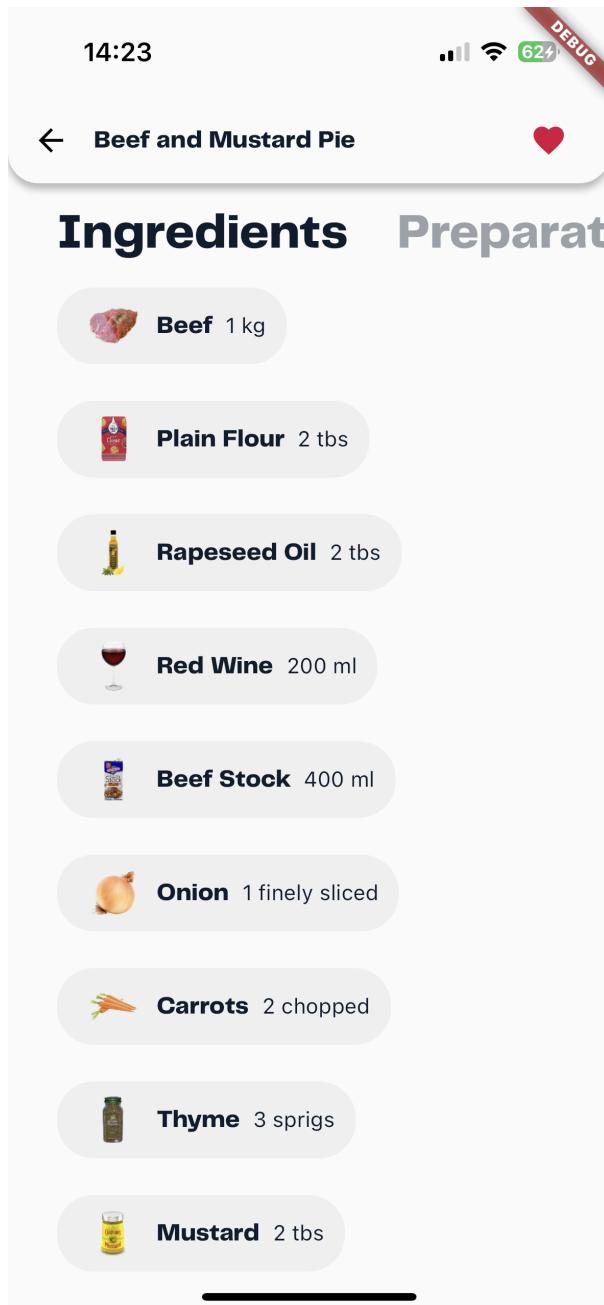


8.10. ábra. Recept áttekintés

8.9. Recept részletek

8.9.1. Recept hozzávalók

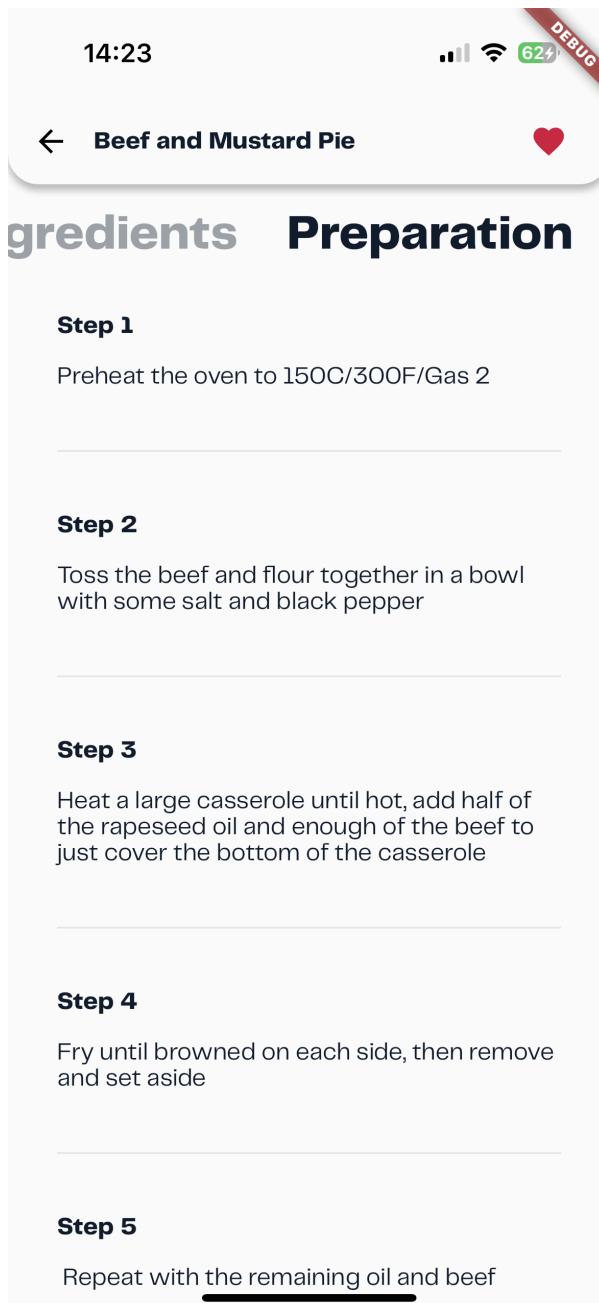
Itt láthatóak az elkészítéshez szükséges hozzávalók, mindegyikhez jár egy mérték-egység és mennyiség, illetve illusztráció.



8.11. ábra. Recept hozzávalói

8.9.2. Elkészítési lépések

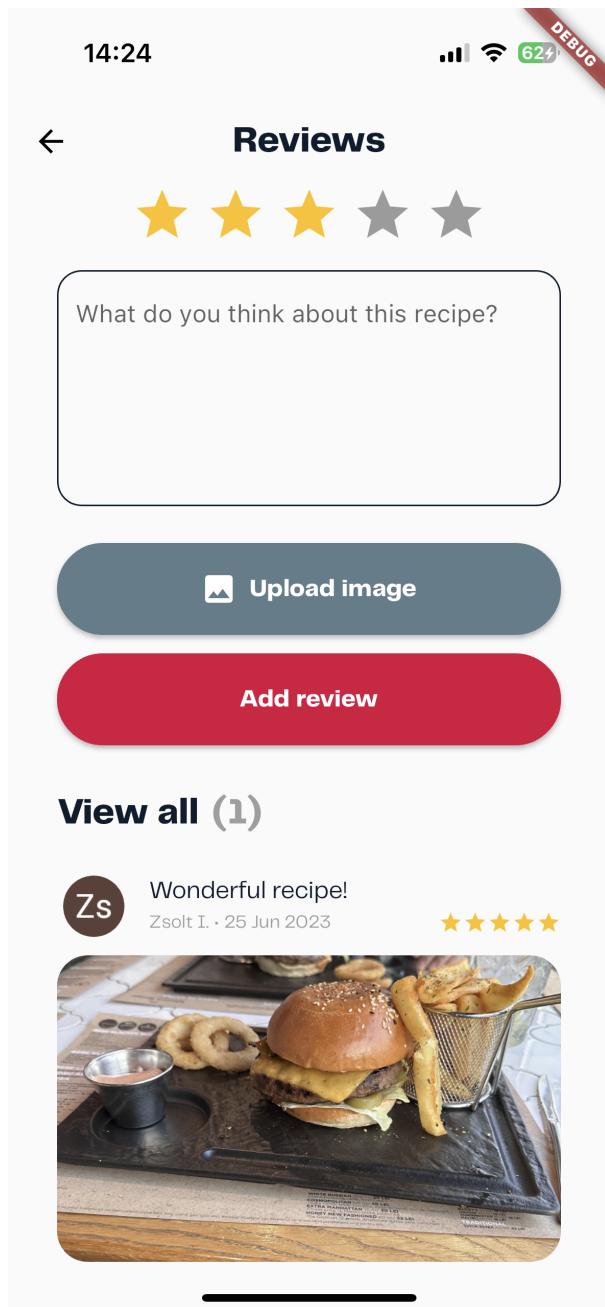
Ezen az oldalon fentről lefelé haladva vannak számozva az elkészítéshez szükséges lépések. Az előző [8.11](#) oldal és ezen oldal között húzással tudunk navigálni.



8.12. ábra. Elkészítési lépések

8.10. Értékelések

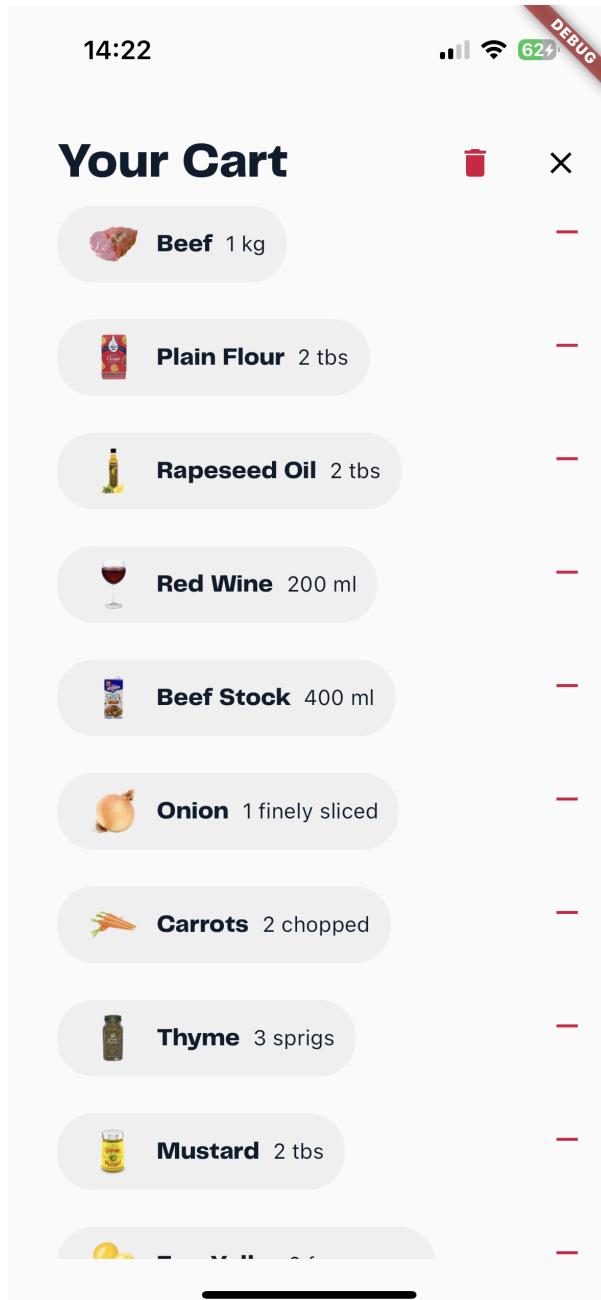
A recepteket tudjuk értékelni is, egytől ötig terjedő skálán, a csillagokra kattintva. Megjegyzést is tudunk hozzáfűzni és opcionálisan képet feltölteni. A kép kiválasztása után lehetőségünk van visszavonni a feltöltött képet és akár újat választani. Továbbá láthatjuk a többi felhasználó által írt értékeléseket.



8.13. ábra. Értékelés képernyő

8.11. Bevásárló kosár

Itt láthatjuk a kosarunkban lévő termékeket, amelyeket egyesével el tudunk távolítni, ha jobbra, vagy balra húzzuk az adott hozzávalót, illetve a kuka ikonnal ki tudjuk üríteni a kosarat.



8.14. ábra. Bevásárló kosár

9. fejezet

Limitációk és továbbfejlesztési lehetőségek

9.1. Limitációk

Az alábbiakban néhány limitációt említenék meg, amelyek előfordulhatnak a jelenlegi applikációban.

- Az applikációban elérhető receptek száma korlátozott lehet, ami azt eredményezheti, hogy a felhasználók nem találják meg azt amit keresnek.
- Az applikációban lehetőség van felhasználói értékelések és vélemények hozzáfűzésére a receptekhez, de ezeknek az értékeléseknek a megbízhatósága kérdéses lehet. A felhasználók eltérő ízlése és tapasztalatai miatt az értékelések nem mindenkor objektívek.
- Internetkapcsolat nélkül nem használható az applikáció, ez a gyenge lefedettségű területeken okozhat problémákat főleg.
- Az applikációban hiányognak olyan személyre szabási lehetőségek, amelyek lehetővé teszik a felhasználók számára a saját preferenciáik, ételallergiáik vagy diétás igényeik beállítását.

9.2. Továbbfejlesztési lehetőségek

Ugyan a projekt a jelenlegi formájában elvégzi azt a feladatot, amire ki lett találva, de természetesen rengeteg fejlődési lehetőség rejlik benne. Ezek a fejlesztések nagy mértékben képesek lehetnek növelni a felhasználói élményt, valamint képes lehet még több feladatot ellátni és ezzel új lehetőségeket nyitni meg a felhasználóknak. A következő néhány pontban röviden összefoglalom, hogy mik azok a területek, ahol az alkalmazás még bővíthetne.

9.2.1. Felhasználói fórum

Létre lehetne hozni egy fórum szekciót, amely valamilyen szinten hasonlítana a jelenlegi értékelés funkcióról, viszont itt nem recepteket, hanem konkrét témákat lehetne

megbeszélni, mint például főzési technikák vagy eszközök. Esetlegesen kérdőíveket is be lehetne illeszteni, amelyből érdekes statisztikák készülhetnének.

9.2.2. Kibővített keresés

A keresést jelenleg 4 kritérium alapján lehet leszűkíteni: Kalóriasűrűség, elkészítési idő, kategória és hozzávalók. Hasznos lenne még egy funkció, ahol a felhasználó kiválasztja milyen hozzávalókra allergiás, és ez alapján nem jeleníti meg az érintett recepteket. Ehhez viszont a receptek adatbázisát is módosítani kéne, mivel ezek jelenleg nem tartalmazzák az allergének listáját.

9.2.3. Felhasználó általi recept feltöltés

Jelenlegi formában csak az adminisztrátorok tölthetnek fel új recepteket a webes felületet használva. A felhasználók is biztosan sok esetben szeretnének saját recepteket megosztani társaikkal, ezért ez a jövőben egy hasznos újítás lenne.

9.2.4. Szociális funkcionalitások

A felhasználók felvehetnének más felhasználókat a baráti listába, így láthatnák például az általuk véleményezett vagy kedvelt recepteket, illetve későbbi fázisban az általuk feltöltött recepteket.

Összefoglaló

Dolgozatomban egy receptes applikáció létrehozásával foglalkoztam, amelyet különböző programozási technológiák segítségével valósítottam meg, mint például Flutter, Dart és Firebase. Ez utóbbi segítségével sikerült megvalósítani egy valós időben működő felhő alapú adatbázis rendszert, amely kiszolgálja az Android, iOS és Web platformokon futó alkalmazást. Tesztjeim során kiválóan teljesített az alkalmazás, a projekt összességében sikeresnek mondható. Úgy érzem sikerült beleásnom magam ebbe a modern technológiába, és rámutatni ennek működési elveire is. Ahol hagy maga után kívánni valót az applikáció, az a felhasználói felület, ugyanis animáció viszonylag kevés van, és a részletek sincsenek mindenhol profin kidolgozva, viszont a logikai rész, amely az applikáció gerincét adja, hibátlan, könnyen bővíthető a későbbiekbén.

Jövőbeli tervezem közé tartozik a mobil applikáció fejlesztés minél több aspektusának felfedezése és tanulmányozása. Érdemesnek tartom a továbbfejlesztést offline irányba, ami által a kisebb hálózati lefedettségű területeken is használhatóvá válik az applikáció, ezáltal több felhasználót lehetne elérni.

A forráskód elérhető itt: [Kitchn-GitLab](#)

Ábrák jegyzéke

4.1.	Bejelentkezés oldal	14
4.2.	Kezdő oldal	15
4.3.	Recept felvétele a kedvencek közé	15
4.4.	Recept értékelés	16
4.5.	Profil módosítása	16
4.6.	Profilkezelés	17
4.7.	Recept felvétele a kedvencek közé	17
5.1.	Bejelentkezés osztály diagram	20
5.2.	Recept osztály diagram	21
5.3.	Összehasonlítva a Material Design és Cupertino Design	23
6.1.	Git Repository aktivitás	27
7.1.	BLoC működése	28
7.2.	Bejelentkezés szekvencia-diagram	30
7.3.	Regisztráció szekvencia-diagram	31
7.4.	Receptek keresése szekvencia-diagram	31
7.5.	Recept mentése szekvencia-diagram	32
7.6.	Hozzávalók mentése szekvencia-diagram	32
7.7.	Profil módosítás szekvencia-diagram	33
8.1.	Töltőképernyő	34
8.2.	Bejelentkezés képernyő	35
8.3.	Regisztrációs képernyő	36
8.4.	Kezdőképernyő	37
8.5.	Kategóriák a kezdőképernyőn	38
8.6.	Profil képernyő	39
8.7.	Profil módosítása	40
8.8.	Keresés képernyő	41
8.9.	Szűrők beállítása	42
8.10.	Recept áttekintés	44
8.11.	Recept hozzávalói	45
8.12.	Elkészítési lépések	46
8.13.	Értékelés képernyő	47
8.14.	Bevásárló kosár	48

Táblázatok jegyzéke

5.1. Összehasonlítás a Flutter és a natív platformok között	22
7.1. Összehasonlítás az MVVM és a BLoC között	29

Irodalomjegyzék

- [Dah19] Ola Dahl. Exploring end user's perception of flutter mobile apps, 2019.
- [MAA⁺19] Paulo Meirelles, Carla SR Aguiar, Felipe Assis, Rodrigo Siqueira, and Alfre-do Goldman. A students' perspective of native and cross-platform approaches for mobile application development. In *Computational Science and Its Applications–ICCSA 2019: 19th International Conference, Saint Petersburg, Russia, July 1–4, 2019, Proceedings, Part V 19*, pages 586–601. Springer, 2019.
- [Nil09] Erik G Nilsson. Design patterns for user interface for mobile applications. *Advances in engineering software*, 40(12):1318–1328, 2009.
- [PSU18] Gusti Pangestu, Ahmad Afif Supianto, and Fitri Utaminingrum. Food recipe finder mobile applications based on similarity of materials. In *2018 International Conference on Sustainable Information Engineering and Technology (SIET)*, pages 156–161. IEEE, 2018.
- [Wu18] Wenhao Wu. React native vs flutter, cross-platforms mobile application frameworks. 2018.