Universitatea Transilvania din Brașov
Facultatea de Matematică și Informatică

DOCUMENTAȚIA PROIECTULUI

FlavorMix

STUDENȚI
Mihai Luciana
Popica Ștefania

Universitatea Transilvania din Brașov
Facultatea de Matematică și Informatică

Cuprins

½ - 1 pagina despre prezentarea proiectului, ce isi propune, ce probleme rezolva, solutii
½ - 1 pagina despre tehnologiile folosite
½ - 1 pagina despre partea de “backend” - request-uri folosite
1-3 pagini despre arhitectura (componentele principale, navigare, baza de date etc.)
1-2 pagini despre cum poate fi utilizata aplicatia: tipuri de utilizatori, ce vede fiecare, autentificare etc
½ - 1 pagina concluzii

Universitatea Transilvania din Brașov
Facultatea de Matematică și Informatică

Prezentarea proiectului

Aplicația de rețete își propune să ofere utilizatorilor o platformă intuitivă și accesibilă pentru a descoperi și partaja rețete culinare. Aplicația va include funcționalități precum căutarea de rețete, afișarea detaliată a modului de preparare și a timpului necesar pentru fiecare rețetă, posibilitatea de a adăuga rețetele favorite într-o secțiune specială denumită "Our users' favs", și crearea unei liste proprii de cumpărături.

Problemele Rezolvate

1. **Lipsa de Inspirație Culinara:** Aplicația va oferi sugestii și idei creative de rețete.
2. **Partajarea Rețetelor:** Secțiunea "Our users' favs" va permite utilizatorilor să descopere rețete testate și recomandate de alți utilizatori, creând astfel o comunitate activă și ajutându-i să găsească rețete delicioase și fiabile.
3. **Gestionarea Cumpărăturilor:** Funcționalitatea de creare a listei de cumpărături va simplifica procesul de achiziționare a ingredientelor necesare, permițând utilizatorilor să adauge ingredientele direct din rețete în lista lor de cumpărături.

Soluții Oferite

1. **Afișarea Detaliilor Rețetelor:** Fiecare rețetă va conține informații detaliate despre ingredientele necesare, pașii de preparare și timpul total de gătire, oferind utilizatorilor toate informațiile necesare pentru a găti cu succes.
2. **Secțiunea "Our users' favs":** Aceasta va permite utilizatorilor să adauge rețetele preferate într-o secțiune specială, vizibilă pentru toți utilizatorii, facilitând partajarea celor mai apreciate rețete din comunitate.
3. **Listă de Cumpărături Integrată:** Utilizatorii vor putea adăuga ingredientele necesare direct din rețete în lista lor de cumpărături, simplificând astfel procesul de planificare și achiziționare a ingredientelor.

Aplicația de rețete se dorește a fi un instrument esențial în bucătărie, oferind soluții practice și eficiente pentru gătit și planificarea meselor, promovând în același timp partajarea cunoștințelor culinare între utilizatori.

Tehnologii folosite

Kotlin: Kotlin este un limbaj modern, concis și sigur, recomandat pentru dezvoltarea de aplicații Android. Kotlin oferă interoperabilitate completă cu Java, ceea ce permite utilizarea bibliotecilor și codului Java existent. De asemenea, facilitează scrierea unui cod mai curat și mai expresiv, reducând posibilitatea apariției erorilor.

Room: O bibliotecă de persistență care oferă o abstractizare peste SQLite, permițând manipularea bazelor de date într-un mod mai sigur și mai eficient. Room include verificări la timpul de compilare pentru interogările SQL, ceea ce reduce riscul de erori la runtime.

Retrofit: O bibliotecă pentru efectuarea de apeluri HTTP. Este utilizată pentru a comunica cu API-uri web, transformând răspunsurile JSON în obiecte Kotlin și facilitând astfel integrarea serviciilor web.

Data Binding și View Binding Data Binding și View Binding sunt tehnologii care permit legarea componentelor UI cu sursele de date ale aplicației. Acestea reduc codul boilerplate necesar pentru actualizarea elementelor UI și îmbunătățesc performanța aplicației. Prin utilizarea acestor tehnologii, codul devine mai curat și mai ușor de gestionat, reducând riscul de erori.

Folosind aceste tehnologii, se pot crea aplicații Android robuste, eficiente și scalabile. Kotlin, împreună cu Android Studio și suitele de biblioteci moderne, oferă un ecosistem complet pentru dezvoltarea rapidă și eficientă a aplicațiilor mobile.

Backend - Request-uri HTTP și Integrarea cu Serverul

Am utilizat un Meal API pentru extragerea rețetelor în aplicație noastră pentru a avea acces la o varietate de rețete culinare direct de pe internet. Retrofit este o bibliotecă populară în comunitatea Android pentru gestionarea request-urilor HTTP.

Configurarea Retrofit pentru Meal API

Pentru a integra un Meal API în aplicația noastră am urmat pașii:

1. **Adăugarea Dependințelor:** Adaugă Retrofit și convertorul Json în fișierul build.gradle al modulului proiectului.
2. **Crearea unui Model de Date:** Definește clasa de date care corespunde structurii JSON a răspunsului de la Meal API (de exemplu, dacă răspunsul include o listă de rețete).
3. **Definirea Interfeței API:** Creează o interfață care definește endpoint-urile API și metodele corespunzătoare request-urilor.
4. **Inițializarea Retrofit:** Configurează Retrofit și creează o instanță a serviciului API.
5. **Efectuarea unui Request:** Utilizează serviciul API pentru a efectua request-uri și a obține date.

Universitatea Transilvania din Brașov
Facultatea de Matematică și Informatică

Arhitectura aplicației

Componentele Principale ale Aplicației

1. **Pagini Principale:**
   - **Pagina de Login:** Utilizatorii existenți se pot autentifica în aplicație.
   - **Pagina de Înregistrare:** Noii utilizatori pot crea un cont.
   - **Pagina de Rețete:** Utilizatorii pot căuta rețete folosind o bară de căutare, vizualiza rețeta zilei și explora rețetele pe categorii.
   - **Pagina de Favorite:** Afișează toate rețetele marcate ca favorite de utilizatori.
   - **Pagina cu Lista de Cumpărături:** Utilizatorii pot adăuga sau șterge produse din lista lor de cumpărături.
   - **Pagina de Profil:** Afișează datele personale ale utilizatorului și oferă opțiunea de delogare.

Navigare

Navigarea între paginile aplicației este gestionată de 'Navigation Component' din Android Jetpack. Acesta permite definirea unui grafic de navigare care descrie toate posibilele rute și acțiuni între fragmente sau activități. Acest sistem facilitează tranzacțiile de navigare într-un mod coerent și ușor de urmărit, oferind suport pentru argumente de navigare, animații de tranziție și legături adânci (deep links).

Arhitectura MVVM (Model-View-ViewModel)

Pentru a asigura o separare clară a responsabilităților și pentru a facilita testarea și întreținerea, se folosește arhitectura MVVM:

1. **Model:**
   - Reprezintă datele și logica de business. Include clase de date care definesc structurile pentru utilizatori, rețete și categorii.
   - Repository: Un layer de abstractizare care gestionează interacțiunile cu sursele de date, cum ar fi API-urile externe și baza de date locală.
2. **ViewModel:**
   - Conține logica de prezentare specifică fiecărei pagini. Este responsabil pentru pregătirea datelor pe care View-ul le va afișa și pentru gestionarea stării UI.

   - Folosește LiveData pentru a permite View-urilor să observe modificările de date și să reacționeze în mod corespunzător.
3. **View:**
   - Componentele UI, cum ar fi activitățile și fragmentele, care afișează datele și interacționează cu utilizatorul. Ele observă LiveData din ViewModel-uri și actualizează UI-ul când datele se schimbă.

Baza de Date

Room Database:

- Folosită pentru persistența datelor locale, Room oferă o abstractizare peste SQLite, facilitând gestionarea bazelor de date.
- Entitățile definesc structura tabelelor din baza de date.
- DAO-urile (Data Access Objects) conțin metode pentru accesarea și manipularea datelor din baza de date.

Repository:

- Repository-ul interacționează cu API-ul extern prin Retrofit și cu baza de date locală prin Room.
- Se ocupă de sincronizarea datelor între sursele locale și cele remote, oferind un layer de abstractizare pentru ViewModel-uri.

Interacțiunea cu API-ul

Retrofit:

- Retrofit este utilizat pentru a efectua request-uri HTTP către un API extern care furnizează date despre rețete.
- Transformă răspunsurile JSON în obiecte Kotlin, care sunt mai ușor de gestionat în cadrul aplicației.
- Repository-ul folosește Retrofit pentru a obține date din API și pentru a le stoca în baza de date locală, dacă este necesar.

Fluxul de Date și Sincronizare

1. **Autentificare și Înregistrare:**
   - Utilizatorii introduc datele în paginile de login și înregistrare.

   - Datele sunt trimise către server pentru autentificare sau creare de cont.
2. **Accesarea Rețetelor:**
   - Utilizatorul accesează pagina de rețete unde poate căuta sau explora.
   - Request-urile sunt trimise către API-ul extern pentru a obține rețete relevante, care sunt apoi afișate utilizatorului.
3. **Adăugarea la Favorite:**
   - Utilizatorii pot marca rețetele ca favorite prin apăsarea unui buton specific.
   - Rețetele favorite sunt stocate în baza de date locală și afișate în pagina de favorite.
4. **Gestionarea Listei de Cumpărături:**
   - Utilizatorii pot adăuga sau șterge produse din lista de cumpărături.
   - Modificările sunt persistate în baza de date locală, permițând accesul offline.
5. **Profilul Utilizatorului:**
   - Pagina de profil afișează datele personale și oferă opțiuni de delogare.
   - Datele sunt preluate din baza de date locală și, dacă este necesar, sincronizate cu serverul.

Arhitectura descrisă asigură modularitatea, scalabilitatea și ușurința de întreținere a aplicației. Utilizarea MVVM separă clar logica de business de UI, în timp ce Navigation Component facilitează navigarea între pagini. Room și Retrofit asigură gestionarea eficientă a datelor și interacțiunea cu sursele externe. Această structură oferă o bază solidă pentru dezvoltarea unei aplicații de rețete robuste și ușor de utilizat.

Utilizarea aplicației

Tipuri de Utilizatori

Aplicația de rețete este destinată utilizatorilor atentificați, cei neautentificați pot vedea doar paginile de autentificare și înregistrare. Pentru a accesa funcționalitățile complete ale aplicației, aceștia trebuie să se autentifice sau să își creeze un cont nou. După autentificare, utilizatorii au acces complet la toate funcționalitățile aplicației.

Funcționalități pentru Utilizatori Autentificați

Pagina de Login permite utilizatorilor să se autentifice introducând email-ul și parola. După validarea credențialelor, aceștia sunt redirecționați către pagina principală cu rețete.

Pagina de Înregistrare este destinată utilizatorilor noi care doresc să își creeze un cont. Utilizatorii trebuie să furnizeze informațiile necesare pentru crearea contului, precum numele, prenumele, adresa de email și parola.

Pagina de Rețete este locul unde utilizatorii pot căuta și explora rețete. Bara de căutare permite utilizatorilor să găsească rețete specifice introducând termenii de căutare. În plus, utilizatorii pot vedea rețeta zilei, o rețetă special selectată pentru ziua respectivă. Pagina de rețete oferă și posibilitatea de a explora rețetele pe categorii facilitând astfel găsirea rapidă a rețetelor dorite.

Pagina de Favorite este dedicată rețetelor marcate ca favorite de utilizatori. În timp ce navighează prin rețete, utilizatorii pot adăuga rețetele preferate la favorite prin apăsarea butonului cu inimă din descrierea rețetei.

Pagina cu Lista de Cumpărături permite utilizatorilor să gestioneze produsele necesare pentru gătit. Utilizatorii pot adăuga ingrediente din rețete direct în lista de cumpărături și pot, de asemenea, să adauge sau să șterge manual produse din listă. Lista de cumpărături este stocată local și sincronizată periodic cu serverul, permițând accesul offline și asigurând că datele sunt actualizate și disponibile pe multiple dispozitive.

Pagina de Profil afișează informațiile personale ale utilizatorului și oferă opțiunea de delogare,

Concluzii

Aplicația de rețete dezvoltă o soluție completă și integrată pentru utilizatorii care doresc să descopere, să gestioneze și să partajeze rețete culinare. Folosind Kotlin ca limbaj principal, aplicația beneficiază de o sintaxă modernă și de siguranța tipurilor, ceea ce reduce semnificativ riscul apariției bug-urilor și facilitează scrierea unui cod curat și ușor de întreținut.

Arhitectura MVVM, împreună cu componentele LiveData și ViewModel, asigură o separare clară a logicii de business de logica UI, făcând aplicația mai modulară și mai ușor de testat. Room Database gestionează eficient persistența datelor locale, oferind o interfață abstractă peste SQLite care simplifică manipularea datelor și reduce codul boilerplate.

Retrofit este utilizat pentru interacțiunea cu API-urile externe, facilitând realizarea request-urilor HTTP și deserializarea răspunsurilor JSON, ceea ce permite integrarea facilă a datelor externe în interfața utilizatorului. Navigation Component asigură o navigare intuitivă și consistentă între diferitele pagini ale aplicației, îmbunătățind experiența utilizatorului.

Firebase Authentication gestionează procesul de autentificare și înregistrare a utilizatorilor, oferind o soluție sigură și ușor de implementat pentru gestionarea conturilor.

Această structură bine definită și setul de funcționalități robuste contribuie la succesul pe termen lung al aplicației, satisfăcând nevoile utilizatorilor de a descoperi și gestiona rețetele într-un mod simplu și eficient.