MINISTERUL EDUCAȚIEI

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

MY LIBRARY - APLICAȚIE DE MOBIL CU SISTEM DE RECOMANDARE PENTRU CĂRȚI

LUCRARE DE LICENȚĂ

Absolvent: **Denisa-Mihaela DUNCA**

Coordonator științific: **Conf.Dr.Ing. Anca MĂRGINEAN**

2023

MINISTERUL EDUCAȚIEI

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

DECAN,
Prof. dr. ing. Liviu MICLEA

DIRECTOR DEPARTAMENT,
Prof. dr. ing. Rodica POTOLEA

Absolvent: **Denisa-Mihaela DUNCA**

MY LIBRARY - APLICAȚIE DE MOBIL CU SISTEM DE RECOMANDARE PENTRU CĂRȚI

1. **Enunțul temei:** *My Library este o aplicație de mobil care permite utilizatorului să își creeze un cont, să se autentifice, să caute și să adauge cărți în propria bibliotecă digitală și să primească recomandări de cărți pe baza celor pe care acesta le are deja în bibliotecă.*
2. **Conținutul lucrării:** *(enumerarea părților componente) Exemplu: Pagina de prezentare, aprecierile coordonatorului de lucrare, titlul capitolului 1, titlul capitolului 2, titlul capitolului n, bibliografie, anexe.*
3. **Locul documentării:** *: Universitatea Tehnică din Cluj-Napoca, Departamentul Calculatoare*
4. **Consultanți:**
5. **Data emiterii temei:** 1 Noiembrie 2022
6. **Data predării:** 6 iulie 2023

Absolvent: _____

Coordonator științific: _____

MINISTERUL EDUCAȚIEI

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

Declarație pe propria răspundere privind autenticitatea lucrării de licență

Subsemnatul(a) _____,
_____ legitimat(ă) cu
_____ seria _____ nr. _____
CNP _____, autorul lucrării
_____
_____
elaborată în vederea susținerii examenului de finalizare a studiilor de licență la Facultatea de Automatică și Calculatoare, Specializarea _____ din cadrul Universității Tehnice din Cluj-Napoca, sesiunea _____ a anului universitar _____, declar pe propria răspundere că această lucrare este rezultatul propriei activități intelectuale, pe baza cercetărilor mele și pe baza informațiilor obținute din surse care au fost citate, în textul lucrării și în bibliografie.

Declar că această lucrare nu conține porțiuni plagiate, iar sursele bibliografice au fost folosite cu respectarea legislației române și a convențiilor internaționale privind drepturile de autor.

Declar, de asemenea, că această lucrare nu a mai fost prezentată în fața unei alte comisii de examen de licență.

În cazul constatării ulterioare a unor declarații false, voi suporta sancțiunile administrative, respectiv, *anularea examenului de licență.*

Data _____

Nume, Prenume _____

Semnătura

Instrucțiuni generale.

De citit înainte (această pagină se va elimina din versiunea finală):

1. Cele trei pagini anterioare (foaie de capăt, foaie sumar, declarație) se vor lista pe foi separate (nu față-verso), fiind incluse în lucrarea listată. Foaia de sumar (a doua) necesită semnătura absolventului, respectiv a coordonatorului. Pe declarație se trece data când se predă lucrarea la secretarii de comisie.
2. Pe foaia de capăt, se va trece corect titulatura cadrului didactic îndrumător, în engleză (consultați pagina de unde ați descărcat acest document pentru lista cadrelor didactice cu titulaturile lor).
3. Fiecare capitol începe pe pagină nouă.
4. Marginile paginilor nu se modifică.
5. Respectați restul instrucțiunilor din fiecare capitol.
6. Am inclus pachetul `hyperref` pentru a genera legături de navigare atât în document cât și la link-uri de web. Pentru listarea pe hârtie a fișierului pdf decomentați linia care conține `%\hypersetup{hidelinks}` aflată în partea de început a fișierului principal `thesis_rom.tex`.

Cuprins

Capitolul 1. Introducere

Cărțile reprezintă pentru omenire un izvor infinit de cunoaștere, ele conțin gândurile și experiențele scriitorului lăsate în dar cititorului. Indiferent de autor, perioada în care au fost scrise, genul lor sau chiar limba în care au fost tipărite, cărțile reușesc să transmită emoție, poveste sau noi taine ale lumii, pe care lectorul, dornic să le afle, le citește pierzând-se printre rândurile și paginile scrise.

Odată cu evoluția tehnologiei, cărțile nu s-au lăsat mai prejos, cărțile electronice numite și E-books, cărțile audio, cărțile în format PDF, sau pur și simplu cărțile online, toate au colaborat spre un mai ușor acces la informație pentru utilizator. Dar cum rămâne cu bibliotecile, acele comori cu mii si mii de diamante de înțelepciune? Nu au fost în totalitate uitate, sunt încă vizitate de iubitorii de cârti tipărite, însă au evoluat și ele sub forma a o mulțime de aplicații de tip bibliotecă online, care pun la dispoziție mii de cărți electronice și care permit utilizatorilor să țină evidența lecturilor citite și să descopere cărți, autori si idei noi.

Digitalizarea bibliotecilor a permis o multitudine de avantaje. Pe lângă accesul în orice moment la orice carte dorită, domenii precum inteligența artificială sau statistica și-au găsit locul în astfel de aplicații. Extensii care calculează timpul în care o carte a fost citită, sau statistici care arată câte cărți au fost citite pe an, genurile preferate, autorii îndrăgiți, chiar și funcționalități care permit utilizatorilor să adauge diferite notițe și evidențieri pe carte, toate au contribuit la o experiență mai ușoară și o motivație mai puternică de a citii, mai ales pentru generațiile noi. De asemenea, inteligența artificială nu s-a lăsat mai prejos nici aici, aplicațiile tip bibliotecă o integrează pentru a oferii utilizatorului recomandări de cărți pe baza celor deja citite prin diferiți algoritmi, sau oferă funcționalități prin care utilizatorul poate să își scaneze cartea pentru a căuta informații despre ea sau chiar să genereze un rezumat al cărții. Toate aceste extensii și îmbunătățiri a modului de a citii, aduc speranța că generațiile noi nu își vor pierde dorința de a lectura și de a afla noi informații.

Sistemele de recomandare au avut un impact puternic asupra internetului, atât asupra aplicațiilor web, cât și a celor de mobil. Aceste sisteme, care au la bază o serie de algoritmi de inteligență artificială, au fost concepute pentru a analiza datele puse la dispoziție de către fiecare utilizator, date care pot proveni dintr-o multitudine de activități și informații pe care acesta le face în platforma pe care o utilizează. De la numărul de vizionări și click-uri, poziția geografică și orele de activitate, până la like-uri, adăugare în wishlist sau produse comandate, toate construiesc seria de date necesare pentru a oferii utilizatorului recomandări personalizate. Aceste sisteme de recomandare pot fi regăsite în foarte multe domenii, în special pot fii observate în domeniul comerțului, magazine online, reclamele personalizate în funcție de preferințe, dar și în domeniul divertismentului, filme și seriale recomandate, videoclipuri pe Youtube sau alte platforme de streaming, muzică și cărți.

De-a lungul timpului, sistemele de recomandare au evoluat de la simple abordări bazate pe filtrare colaborativă, până la sisteme cu învățare automată precum arbori de decizie, rețele neuronale și mașini cu vectori suport (SVM), evoluție care a dus la crearea unor sisteme din ce în ce mai precise care oferă recomandări mai relevante pentru utilizatori.

Primul sistem de recomandare recunoscut a fost „Tapestry", un sistem de e-mail dezvoltat în anul 1992 de Xerox Palo Alto Research Center [1]. Din cauza numărului mare de e-mail-uri și documente trimise, unui utilizator ii era destul de greu să răspundă la toate e-mail-urile importante în timp, era nevoie de un filtru pe bază de conținut care să scaneze listele de e-mail-uri și să le păstreze doar pe cele despre care utilizatorul era interesat. Dar „Tapestry" nu s-a oprit aici, a găsit un mod mult mai eficient de a filtra e-mail-uri prin introducerea acțiunilor umane în acest proces. Astfel s-a dezvoltat filtrarea colaborativă, în care utilizatorii colaborează pentru a se ajuta unul pe celălalt prin înregistrarea acțiunilor lor, în cazul de față, prin înregistrarea documentelor pe care le-au citit și recomandarea lor altor utilizatori care au citit documente similare. Însă sistemul era bazat pe diferite interogări spre baza de date care verificau utilizatorii care au citit documentul și cât timp a durat până un document a fost deschis sau dacă nu a fost deschis deloc, deci nu avea algoritmi de dezvoltați cu ajutorul inteligenței artificiale. Desigur, odată cu trecerea timpului, domeniul inteligenței artificiale și învățării automate s-a dezvoltat, iar sistemele de recomandare au implementat noi algoritmi și noi idei care le-au făcut din ce în ce mai eficiente și mai exacte.

Recomandările pot rezulta în urma mai multor moduri de filtrare, aici se evidențiază cel mai mult filtrarea bazată pe conținut și filtrarea colaborativă. Filtrarea bazată pe conținut recomandă obiecte bazate pe similaritatea dintre conținutul lor și preferințele utilizatorului. Analizează atributele și caracteristicile fiecărui obiect în parte și creează profiluri de utilizator bazate pe preferințele pentru anumite atribute precum genul cărții, autor sau anul publicării. Filtrarea colaborativă, pe de altă parte, recomandă obiecte și pe baza acțiunilor altor utilizatori similari asupra lor. Există două tipuri de filtrare colaborativă, cea bazată pe utilizator, care identifică utilizatori care au aceleași preferințe și recomandă cărțile care sunt comune amândurora, și cea bazată pe obiect, care identifică obiecte similare cu cele care sunt deja preferate de utilizator și le recomandă.Desigur, există mai multe tipuri de filtrare pentru un sistem de recomandare precum filtrare hibridă, care unește multiple abordări de recomandare pentru a îmbunătății eficiența și precizia, bazată pe cunoștințele explicite pe care le au despre utilizator și obiecte, bazată pe popularitate sau context, sau care folosesc Deep Learning și rețele neuronale care manevrează date de mari dimensiuni.

În ceea ce privește viitorul sistemelor de recomandare, se poate spune că se urmărește crearea unor sisteme cu o cât mai buna precizie, care să satisfacă preferințele utilizatorului, să personalizeze cât mai mult rezultatele date și să ofere recomandări cât mai relevante și diverse. Multe companii caută aceste cerințe pentru utilizatorii lor, iar toate aceste cerințe au la bază algoritmii de filtrare amintiți mai sus. Toate aceste tipuri de filtrare a sistemelor de recomandare, modul cum sunt implementați algoritmii, precum și o implementare proprie a unui sistem de recomandare în cadrul unei aplicații de mobil tip bibliotecă, vor fi prezentate în decursul următoarelor capitole.

Capitolul 2. Obiectivele proiectului

Obiectivul principal al proiectului reprezintă dezvoltarea unei aplicații pentru mobil care permite utilizatorului să își creeze propria bibliotecă de cărți virtuală. În cadrul aplicației, utilizatorul poate să caute cărți într-o arhivă pusă la dispoziție de aplicație sau să adauge manual cărți noi completând datele necesare pentru carte. Pe baza acestor cărți din bibliotecă, utilizatorul va primi recomandări de cărți noi și va putea vedea statistici personalizate. Pentru a atinge obiectivul principal, aplicația va pune la dispoziție o serie de funcționalități care vor oferi, atât libertatea utilizatorului de a-și personaliza biblioteca virtuală, cât și o experiență cât mai ușoara și sigură în ceea ce privește utilizarea aplicației.

2.1. Cerințe funcționale

Utilizatorului îi vor fi oferite o varietate de funcționalități complexe și interactive, concepute cu scopul final de a crea și personaliza datele într-un mod complet adaptat nevoilor și preferințelor sale, deschizând astfel posibilitatea de a beneficia de recomandări de cărți personalizate și de a accesa statistici relevante pentru îmbunătățirea experienței sale de lectură.

2.1.1. Crearea contului și Autentificarea

Pentru a putea beneficia de toate funcționalitățile aplicației, utilizatorul va trebui să se înregistreze prin crearea unui cont personalizat folosind email și parolă. După finalizarea procesului de înregistrare, utilizatorul va trebui sa se autentifice. Odată ce credențialele sunt introduse, contul utilizatorului rămâne autentificat până când acesta se deconectează explicit (sign out). De asemenea, odată conectat la un cont, utilizatorul va putea să își adauge și alte date personale și să își modifice fotografia de profil.

2.1.2. Căutarea și adăugarea de cărți

Utilizatorul va putea adăuga cărți în biblioteca proprie prin două posibilități. Fie prin căutare după titlu, autor, gen sau ISBN într-o bază de date alcătuită din 6000 de cărți și pusă la dispoziție de aplicație. Fie prin adăugare manuală a unei cărți, unde utilizatorul va completa datele despre carte precum titlu, autor, gen, descriere, număr de pagini, ISBN, anul publicării și, desigur, imaginea ce conține coperta cărții.

2.1.3. Gestionarea cărților din bibliotecă

Odată ce cărțile au fost adăugate în bibliotecă, utilizatorul va avea posibilitatea să caute printre cărțile din bibliotecă după titlu, autor, gen sau ISBN, să modifice informații despre fiecare carte, să adauge informații noi precum o evaluare proprie asupra cărții, numărul de pagini citi, data în care a început să fie citită cartea și data când a fost terminată și notițe personale despre cartea respectivă. Utilizatorul va putea vizualiza aceste informații pentru fiecare carte în parte și, desigur, va putea șterge cărți din bibliotecă.

2.1.4. Vizionarea recomandărilor și a statisticilor

În funcție de evaluările proprii lăsate de utilizator fiecărei cărți, acesta primește o serie de alte cărți recomandate. Recomandarea se face, atât pe baza evaluărilor comune între utilizatori și cărți, cât și după elemente precum gen sau autor. Din lista de cărți recomandate, utilizatorul poate să vizioneze detalii despre ele și, de asemenea, să le adauge și pe ele în biblioteca proprie. În funcție de cărțile din bibliotecă și de datele personalizate adăugate de utilizator, date precum numărul de pagini citite, data de început și final și evaluarea lăsată cărții, se vor genera o serie de statistici care să arate genurile preferate, cartea preferata, numărul de cărți pe care utilizatorul a terminat de citit și numărul de pagini citite în total în fiecare lună.

2.2. Cerințe nonfuncționale

Pentru a asigura o experiență optimă a utilizatorilor și pentru a dezvolta o aplicație eficientă și fiabilă, se vor urmări îndeplinirea următoarelor cerințe non-funcționale. Aceste cerințe sunt cruciale pentru a crea o aplicație de calitate, care să funcționeze corespunzător și să îndeplinească așteptările utilizatorilor. Cerințe precum utilizabilitate, securitate, disponibilitate, compatibilitate, sincronizare, gestionarea erorilor și recuperarea vor fi detaliate în cele ce urmează.

2.2.1. Ușurință în utilizare

Interfața aplicației trebuie să fie cât mai ușor de utilizat, acest lucru va fi atins atât prin dezvoltarea unei interfețe cât mai intuitive, cu un meniu care să conțină principalele pagini pe care utilizatorul le poate accesa, dar și prin punerea la dispoziție a unei documentații detaliate despre utilizarea aplicației și modul de instalare al acesteia.

De asemenea, design-ul aplicației trebuie să fie unul user-friendly, ușor de înțeles și de adaptat. Trebuie să aibă o structură clară și o navigație rapidă pentru ca utilizatorul să poată interacționa cu ușurință cu toate funcționalitățile și pentru a se orienta rapid. Aplicația trebuie să ofere o experiență plăcută din punct de vedere vizual, cu o paletă de culori adecvată și elemente cu dimensiuni potrivite astfel încât utilizatorul să fie mulțumit în utilizarea ei.

2.2.2. Securitate

Securitatea aplicației va fi oferită în principal de platforma Firebase, aceasta oferă facilități active pentru autentificarea si stocarea datelor în siguranță. Extensia de autentificare oferită de platforma Firebase gestionează procesul de înregistrare și autentificare într-un mod securizat, parolele utilizatorilor fiind stocate criptat și nu pot fi accesibile sau vizibile în mod direct. Firebase Authentication folosește pentru autentificarea cu email și parolă, un sistem propriu bazat pe token-uri JWT (JSON Web Token) care trimite email-ul și parola către serviciul Firebase Authentication, acesta verifică și validează informației și primește înapoi un token de autorizare utilizat ulterior pentru identificare și autorizarea utilizatorului în cadrul aplicației.

Pentru stocarea de date a utilizatorului, atât a informațiilor personale cât și a cărților și pozelor, Firebase oferă servicii precum Realtime Database și Storage care țin datele utilizatorului într-un mod sigur prin declararea de reguli de securitate tot in interiorul platformei.

2.2.3. Compatibilitate și portabilitate

În cadrul aplicației compatibilitatea va fi oferită prin modul cum este construită aplicația folosind măsurători care se adaptează mediului în care aplicația este pornită, deci indiferent de versiunea și tipul telefonului interfața se ajustează pentru a arăta estetic și corect. Design-ul și layout-ul aplicației vor putea să se adapteze la diferite dimensiuni de ecran și rezoluții asigurând o experiență plăcută utilizatorului.

Portabilitatea va fi oferită prin faptul ca aceasta este o aplicație pe telefon, deci biblioteca virtuală poate fi oricând și oriunde la dispoziția utilizatorului. Acest lucru permite ca utilizatorul să își adauge cărți în bibliotecă oricând vede ceva interesant sau nou pe care l-ar interesa, precum și să caute recomandări de cărți când nu mai are idei de noi cărți.

2.2.4. Conectivitate și sincronizare

Utilizatorii, odată autentificați, își păstrează conectat contul în aplicație atâta timp cat nu se face acțiunea de deconectare, sign out din aplicație sau cât timp aplicația nu este dezinstalată. Astfel, utilizatorii vor putea să acceseze mult mai rapid și ușor aplicația oferind un plus de confort și rapiditate utilizării.

De asemenea, utilizatorii vor putea să își sincronizeze propria bibliotecă între diferite dispozitive deoarece odată autentificați, indiferent de dispozitivul pe care își accesează aplicația, aceștia vor putea să vadă aceeași listă de cărți și poate să își gestioneze și să facă modificări care automat se vor reflecta pe toate dispozitivele conectate. Acest lucru este realizat deoarece datele vor fi stocate în cloud prin platforma Firebase.

2.2.5. Gestionarea erorilor și recuperarea

Pe tot parcursul utilizării aplicației, utilizatorii vor primi mesaje legate de acțiunile pe care le fac în aplicație. Tipurile de mesaje sunt atât pentru a-i anunța că o anumită acțiune a funcționat corect, dar și pentru cazurile în care o funcționalitate a eșuat din diferite motive, telefonul nu mai are acces la internet sau baza de date devine indisponibilă. De asemenea, în formularele pe care utilizatorul le poate completa, apar mesaje de validare în cazul în care introduce date care nu sunt potrivite pentru intrarea respectivă, astfel încât utilizatorul să știe ce anume să modifice pentru a-și crea un cont, pentru a se autentifica sau pentru a adăuga o carte.

Recuperarea datelor în caz de defecte va fi asigurată automat de Firebase. Platforma păstrează o copie a datelor care poate fi utilă în cazul în care apare o problemă sau o pierdere accidentală a datelor. Backup-urile în Realtime Database sunt create automat și sunt gestionate de platformă în fundal.

Capitolul 3. Studiu bibliografic

3.1. Sisteme de recomandare

https://sci-hub.se/https://doi.org/10.1007/s41870-018-0138-8
https://arxiv.org/pdf/2005.12210.pdf

3.1.1. Filtrare bazată pe conținut

http://users.ics.forth.gr/ potamias/mlnia/paper$_6$.*pdf*
https://sci-hub.se/10.1007/978-3-642-19896-0$_1$1

3.1.2. Filtrare colaborativă

https://sci-hub.se/https://doi.org/10.1145/358916.358995
https://sci-hub.se/10.1007/978-3-540-72079-9$_9$

3.1.3. Filtrare hibridă

https://aclanthology.org/W03-1103.pdf

3.1.4. Modele

https://sci-hub.se/10.1109/ICACCS51430.2021.9441927
https://sci-hub.se/https://doi.org/10.3102/1076998619872761

3.2. Dezvoltarea aplicațiilor Android

Dezvoltarea aplicațiilor Android reprezintă un subiect actual în cercetările dezvoltatorilor. Tendințele merg spre dezvoltarea de aplicații cât mai ușor de utilizat, cu un sistem de securitate cât mai bun și o gestiune bună a datelor. În [2] autorii își propun să facă o prezentare generală legată de dezvoltarea aplicațiilor Android și să pună accent pe evoluția sistemului de securitate în cadrul acestor aplicații.

Dezvoltatorii de aplicații Android se bucură de o serie de avantaje. În primul rând, sistemele de operare Android OS sunt majoritare pe piață și sunt cele mai întâlnite și folosite de utilizatori având ca principale calități modul ușor de utilizare, flexibilitatea și gama mare de variante de aplicații dintre care pot alege. Aplicațiile Android sunt dezvoltate utilizând limbajele de programare Java sau Kotlin ceea ce oferă o serie de alte avantaje. Pe lângă platformele de dezvoltare Java IDE care oferă dezvoltatorilor eficiență și o scriere mai ușoară a codului, mai sunt și bibliotecile oferite de Java și Android care pun la dispoziție o multitudine de avantaje indiferent de domeniul pentru care aplicația este dezvoltată.

Arhitectura unui sistem Android se împarte pe mai multe nivele în funcție de funcționalitățile pe care fiecare nivel se focusează. Nivelul Aplicație este nivelul în care componentele aplicației sunt executate, acestea folosesc librării API pentru a se folosi de resursele oferite de dispozitiv. În acest nivel sunt disponibile aplicații precum meniul de contacte și mesageria. Nivelul de dezvoltare al aplicațiilor este cel care permite dezvoltatorilor să aibă acces la serviciile de bază precum managerul de pagini, de pachete sau

resurse. Ultimul nivel este cel de runtime care este în principal focusat pe statusul de rulare al proceselor. Fiecare program are o mașină virtuală a lui, iar kernel-ul Android, asemănător cu cel al unui Linux, se ocupă de gestionarea proceselor, a memoriei, a bateriei și așa mai departe. Pentru dezvoltarea de aplicații web, autorii lucrării oferă și o imagine sugestivă legată de modul în care este împărțită arhitectura unui astfel de proces de dezvoltare 3.1.

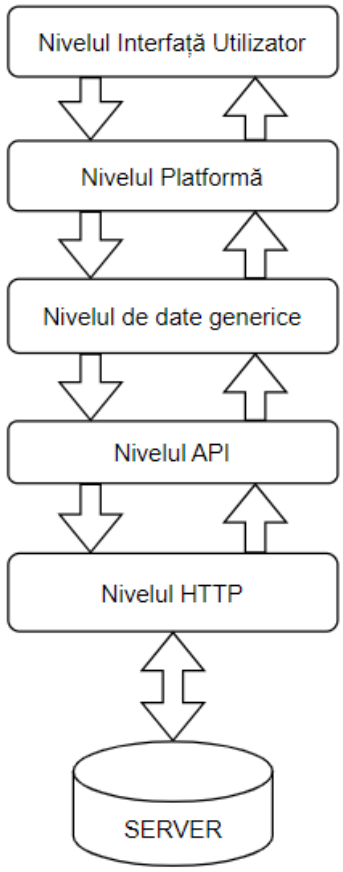

Figura 3.1: Arhitectura unei aplicații Andorid

Nivelul de mai jos, HTTP, este cel care face legătura cu serverul, primind și trimițând cereri acestuia. Aceste răspunsuri de la server sunt trimise mai departe la nivelul API care formatează și formulează răspunsurile sub formă de querry-uri și le trimite mai departe la nivelul cu date generice. În acest nivel sunt declarate și implementate metodele și funcționalitățile necesare, iar apoi datele prelucrate de aceste metode sunt trimise mai departe la nivelul de platformă. Nivelul de platformă este dependent de aceste date pe care acesta le stochează sub diferite moduri precum adaptoare sau listview-uri. Cel mai de sus nivel este desigur interfața cu utilizatorul care este responsabilă de interacțiunile utilizatorului cu aplicația și sincronizarea acestor interacțiuni.

Modelul unei aplicații android este bazat pe o serie de componente: activități, servicii, receptori și furnizori. Activitățile sunt folosite pentru interfața utilizatorului și sunt câte o activitate per fereastră a aplicației. Serviciile se folosesc pentru comunicarea între aplicații și rămân active în background atunci când utilizatorul navighează printre ferestrele aplicațiilor. Un exemplu de serviciul ar fi adăugarea de poză de profil într-un cont. De asemenea, receptorii sunt utilizați pentru a comunica și cu alte aplicații prin

trimitere de mesaje asincrone și sunt folositori în cazul notificărilor sau al apelurilor din timpul utilizării altor aplicații. În final, componentele furnizor sunt cele responsabile pentru stocarea datelor necesare în aplicație precum credențialele pentru autentificare.

3.3. Aplicații existente

Sistemele de recomandare în cadrul aplicațiilor web sau android sunt destul de răspândite, mai ales printre aplicațiile de divertisment și rețele sociale. De la recomandarea unui film pe Netflix, lista de "S-ar putea să îi cunoști" pe Facebook și până la recomandări de diferite produse recent achiziționate pe Emag, toate folosesc un sistem de recomandare propriu bazat pe diferite reguli și diferiți algoritmi. În subcapitolele următoare se vor prezenta o serie de aplicații tip bibliotecă care folosesc un sistem de recomandare personalizat, împreuna cu atributele și caracteristicile specifice lor.

3.3.1. Goodreads

Goodreads este o aplicație pentru cititori dezvoltată în 2006, aceasta se găsește atât pe web, cât și ca aplicație pentru mobil și oferă o serie de funcționalități care unesc ideea de a gestiona cărți și primi recomandări cu ideea de comunitate și rețea socială. Ca și majoritatea aplicațiilor, Goodreads oferă un sistem de creare cont și autentificare și, desigur, utilizatorul poate să își modifice datele sau să adauge și alte date personale pentru a-și mări cercul de prieteni din aplicație. Goodreads oferă un sistem ușor de a descoperi cărți noi, atât prin căutarea lor după diferite cuvinte cheie precum titlu, autor, gen, dar și prin recomandările primite care sunt bazate pe genurile de cărți de care utilizatorii sunt interesați, review-urile lăsate, istoricul de cărți citite și cărțile citite de prietenii din aplicație.

Sistemul de recomandare din aplicația Goodreads se bazează pe o abordare hibridă, însă detaliile de implementare și modelul nu sunt publice. Inițial Goodreads nu oferea ideea de recomandări utilizatorilor, era o simplă aplicație de gestiune a cărților și de socializare cu prieteni și grupuri cu aceleași gusturi în lectură, însă, cu timpul, Goodreads adaugă și un astfel de sistem. Un anunț făcut în 2011 pe site-ul aplicației [3] oferă indicii despre modul în care sistemul este conceput. În cadrul acestui anunț, dezvoltatorii prezintă un nou sistem de recomandare personalizat, afirmând că "Este Netflix-ul recomandărilor de cărți". Goodreads a cumpărat compania Discovereads.com, iar cu ajutorul sistemului lor de recomandare bazat pe deep-learning au reușit să își îmbunătățească recomandările semnificativ. Timp de trei ani, dezvoltatorul Discovereads, Brian Percival, împreună cu Kyusik Chung, au continuat să îmbunătățească tehnologia de recomandare care are la bază competiția "The Netflix prize" [4]. Aceasta unește multiplii algoritmi de învățare automat precum Învățarea prin recompensă (Reinforcement Learning) și tehnici de analiză și filtrare a evaluărilor cărților precum Filtrare colaborativă și Filtrare bazată pe conținut pentru a găsi cele mai bune si precise recomandări de cărți, personalizate pentru fiecare utilizator.

Cel mai mare dezavantaj al sistemului de recomandare dezvoltat de Discovereads, este că, acesta bazându-se în principal pe ideea de recomandare pe bază de rating, odată ce sistemul devine din ce în ce mai popular începe să recomande cărți pe care majoritatea cititorilor le preferă. De asemenea, unele evaluări pot să nu fie precise sau irelevante, iar acest lucru poate afecta sistemul de recomandare. De asemenea, unele evaluări pot să nu fie precise sau irelevante, iar acest lucru poate afecta sistemul de recomandare. Însă, odată cu evoluția sistemelor de recomandare și apariția de noi tendințe, Goodreads

îmbunătățește considerabil modul de recomandare, adăugând din ce în ce mai multe filtre de recomandare personalizate și bazate și pe popularitate.

3.3.2. LibraryThing

La fel ca și Goodreads, LibraryThing este o aplicație care se bazează atât pe gestionarea cărților, cât și crearea unei comunități de cititori pasionați. Aplicație se identifică prin utilizarea de cataloage, acestea sunt liste de cărți create de utilizatori, care pot păstra cărțile pe care aceștia au început să le citească, cărțile preferate sau pur și simplu toate cărțile pe care aceștia le dețin în bibliotecă. Într-un articol de pe pagina de web a aplicației se oferă o serie de caracteristici prin care se diferențiază de celelalte aplicații de cărți. Dezvoltată în 2005, aplicația susține că se identifică prin faptul că recomandările pe care aceasta le oferă nu sunt bazate pe idea de marketing, "Recomandările noastre de cărți nu vor fi niciodată influențate de considerente comerciale" [5].

Sistemul de recomandare LibraryThing este diferit pentru că se bazează pe idea de catalog, creând categorii personalizate de cărți pentru fiecare utilizator. Nici aici nu sunt foarte multe detalii despre sistemul de recomandare, însă în articolul [6] se prezină o scurtă descrirere critică asupra recomandărilor de la LibraryThing. Aplicația folosește o abordare hibridă care combină recomandările bazate pe cărțile ce se găsesc în catalogul fiecărui profil de utilizator, cu alte tipuri de filtrare precum filtrarea colaborativă bazată pe obiect sau pe utilizator, căutând similarități între cărțile diferitor utilizatori. De asemenea, LibraryThing oferă recomandări folosindu-se de metadatele, etichetele și evaluările din catalogul de cărți care analizează legăturile dintre cărți precum etichetele distribuite și apariția cărților în multiple cataloage pentru a identifica o serie de cărți pe care utilizatorii le găsesc interesante.

3.3.3. Amazon Kindle

Amazon, spre deosebire de alte aplicații, a venit cu o soluție care să mulțumească utilizatorul oferindu-i posibilitatea de a-și alege modelul de recomandare și de a-l personaliza, dar nu pe gratis. Amazon Personalize [7] permite dezvoltatorilor să construiască și să publice un sistem de recomandare inteligent care scalează folosind machine learning. Acest sistem analizează informațiile despre client, le identifică pe cele mai importante și folosește cei mai buni algoritmi pentru a antrena și optimiza un model personalizat, croit exact pe cerințele utilizatorului. Această modalitate de a accesa recomandări poate fi aplicată în toate domeniile oferite de Amazon, e-commerce, muzică, filme și, desigur, cărți.

Amazon Kindle este o aplicație destinată utilizatorilor care iubesc să citească, aceasta oferă, pe lângă gestiunea de cărți și autentificare de cont, și posibilitatea de a accesa cărțile, de a le cumpăra și citi direct din aplicație. Utilizând Amazon Personalize, utilizatorul poate să deblocheze și recomandările personalizate din cărțile pe care le-a cumpărat, le-a citit sau pe care le are în lista de dorințe.

3.3.4. Libib

Aplicațiile prezentate până acum reușeau să unească ideea de gestiune a cărților cu sistemele de recomandare, Libib, pe de altă parte este o aplicație care se focusează în principal pe gestiunea cărților. Inspirată din cataloagele oferite de LibraryThing, aplicația Libib oferă utilizatorilor posibilitatea de a cataloga până la 5000 de titluri de cărți. Aceste cataloage sunt sincronizate prin Cloud și oferă statistici pentru utilizator pe baza cărților

conținute. De asemenea, aplicația păstrează idea de socializare și permite utilizatorilor să evalueze cărți, să creeze diferite grupuri și să dea import sau export la colecțiile altor utilizatori. Este o aplicație bună pentru utilizatorii care doar își doresc să țină evidența cărților citite sau a cărților publicate, însă, în comparație cu celelalte aplicații Libib nu oferă un sistem de recomandare. [8]

Capitolul 4. Analiză și fundamentare teoretică

4.1. Sisteme de recomandare

4.1.1. Tensorflow Recommenders

4.1.2. Retrievel Model

4.1.3. Ranking Model

4.2. Perspectiva Tehnologică

4.2.1. Android Studio

Android Studio este o platformă de dezvoltare a aplicațiilor de Android în limbajele Java, Kotlin sau C/C++, este un IDE (Integrated Development Evironment) și se bazează pe funcționalitățile de dezvoltare oferite de Intellij IDEA, având și o interfață asemănătoare. Următoarele caracteristici și funcționalități oferite de Android Studio sunt cele care fac ca această platformă de dezvoltare să fie cea mai bună și cea mai aleasă opțiune pentru aplicații de mobil:

- Are un editor de cod intuitiv care, la fel ca și celelalte platformele de dezvoltare oferite de IDEA, ajută programatorul prin completarea automată, evidențierea greșelilor de sintaxă și oferirea de alternative pentru a le repara;
- Oferă un layout potrivit pentru dezvoltarea de aplicații Android, programatorul având posibilitatea de a vedea schimbările pe care le face în interfața utilizator în timp ce scrie codul XML;
- Are un sistem de construire a aplicațiilor bazat pe Gradle care gestionează toate dependințele și scalabilitatea proiectului;
- Oferă posibilitatea de adăugare a unui Emulator care imită un telefon pentru ca utilizatorul să poate să testeze codul și modul cum arata interfața pe diferite versiuni de mobil. De asemenea, oferă posibilitatea de a conecta propriul telefon, de a descărca aplicația creată în telefon și testarea palpabilă a aplicației, fără utilizarea unui emulator;
- Are un sistem de debugging integrat care este ușor de utilizate și are o interfață foarte sugestivă, la fel și pentru sistemul de testare integrat care oferă dezvoltatorului informații folositoare despre statusul testelor rulate.

4.2.2. Kotlin

Kotlin este un limbaj de programare dezvoltat de JetBrains, cei care au dezvoltat si Intellij IDEA, și a fost lansat în 2010 pentru a oferi o alternativă limbajului Java. Kotlin, la fel ca și Java, este un limbaj care rulează pe o platformă JVM (Java Virtual Machine) și are următoarele caracteristici:

- Are o legătură strânsă cu Java întrucât platformele de dezvoltare, precum Android Studio, oferă posibilitatea de a converti direct codul din Java în Kotlin;
- Este mult mai concis decât Java, iar unii dezvoltatori îl consideră chiar mai ușor de învățat și îl recomandă pentru cei la început. Ca și particularități sintaxa elimină o mulțime de constrângeri regăsite în Java, precum declararea de variabile fără să

specifice direct tipul variabilei, metode de get și set mai concis declarate, verificare de element null printr-un simplu "?" și faptul că nu necesită ";" la final de linie de cod;
- Problema null-pointer exception nu există, deoarece Kotlin impune verificarea statică prin adăugarea operatorului de verificare null "?";
- Face ca manipularea tipurilor de date să fie mai ușoară și mai sigură prin implementarea unor "smart casts" care schimbă automat tipul de date folosit.

4.2.3. XML

Extensible Markup Language este un limbaj utilizat de Android Studio pentru a gestiona modul cum este implementată interfața cu utilizatorul oferind o alternativă bună pentru dezvoltatorii care erau obișnuiți să lucreze în HTML și CSS. XML este utilizat în layout-urile din aplicație care oferă toate informațiile legate de cum sunt aranjate elementele în pagina respectivă, cum arată acestea și ce particularități au asignând id-uri ce pot fi utilizate în continuare în codul de dezvoltare pentru a genera funcționalități prin interacțiunea cu aceste elemente. XML este ușor de înțeles, acesta utilizează etichete și elemente organizate ierarhic, de asemenea permite implementarea de elemente noi adăugate prin dependințe și elemente personalizate de dezvoltator în cadrul aplicației.

4.2.4. Firebase

Firebase este o platformă dezvoltată de Google care oferă multitudine de extensii și avantaje pentru dezvoltatorii de aplicații web și de mobil. Firebase oferă unelte de dezvoltare care simplifică procesul de creare de aplicații, ajută la îmbunătățirea scalabilității aplicației și oferă un mod de auto evaluare prin extensiile de analizare a activităților din aplicație.

Firebase Authentication

Această extensie oferită de Firebase ajută la implementarea cu ușurință a unui proces de înregistrare și de autentificare pentru utilizatori în aplicație. Principalele avantaje sunt securitatea cu care sunt ținute datele și posibilitatea de creare de conturi utilizând conturile de pe alte platforme precum Google și Facebook. Metodele de securitate urmăresc reguli standard ale protocoalelor de securitate și algoritmi de criptare pentru a proteja datele utilizatorilor. De asemenea, parolele trec prin operații de hashing înainte ca acestea să fie stocate în baza de date, astfel, chiar daca baza de date e compromisă, parolele rămân sigure. În plus, oferă și opțiuni pentru recuperarea contului, în cazul în care a fost uitată parola sau a intervenit ceva și baza de date a fost compromisă, Firebase făcând backup-uri periodice.

Firebase Authentication pune la dispoziție și autentificarea în doi pași (Two-factor authentication) care adaugă un nivel extra de securitate. De asemenea, implementează și protocoale de tipul OAuth și autentificare cu token pentru verificarea integrității utilizatorilor, iar Firebase, ca și la toate tipurile de stocare oferă posibilitatea dezvoltatorilor să declare anumite reguli de securitate care să permită utilizatorilor să își vadă doar propriile date.

Firebase Realtime Database

Este o altă extensie oferită de Firebase care permite stocarea de date într-o structură de date flexibilă bazată idea unui arbore JSON. Realtime Database este o bază de date NoSQL (Not Only SQL) concepută pentru a depozita și gestiona datele fără a utiliza schema rigidă de tabele a bazelor de date tradiționale. Datele pot fi grupate în colecții și pot fi ușor exportate și importate direct în interfața Firebase sau prin intermediul Aplicației. Particularitatea principală pentru Real-time Database, este chiar în nume, posibilitatea de sincronizare în timp real al datelor între diferite dispozitive și utilizatori. Este o soluție bună pentru aplicațiile de mobil, întrucât datele sunt stocate în baza de date pe cloud și nu în telefon, astfel aplicațiile ocupă mai puțin spațiu și funcționează mai repede.

În ceea ce privește securitatea datelor, la fel ca și Firebase Authentication, dezvoltatorul poate să creeze propriile reguli de securitate pentru a asigura controlul accesului la baza de date. Regulile permit specificarea de permisiuni de scriere și citire pe diferite tabele din baza de date. Scalabilitatea și performanța sunt asigurate prin împărțirea în mai multe servere localizate în diferite țări care oferă o experiență rapidă utilizatorilor.

Firebase Storage

Un alt serviciu oferit de platforma Firebase, este Firebase storage care oferă posibilitatea de stocare sigură în cloud a elementelor care ocupă mai mult spațiu, de exemplu, pentru stocarea de imagini. Într-o aplicație de mobil imaginile, fișiere mari, videoclipuri, documente și multe altele pot fi stocate și gestionate in Firebase Storage. Practic oferă un sistem de gestiune a-l fișierelor multimedia care asigură siguranță și scalabilitate.

Firebase Machine Learning

4.2.5. Tensorflow

Keras

TFLite

4.2.6. Testare

Testarea aplicațiilor mobile este un proces important pentru creare de aplicații. Aceste procese de testare garantează calitatea produsului și faptul că îndeplinește toate funcționalitățile promise. Testarea implică validarea și verificarea tuturor acțiunilor oferite de aplicație și este un mod prin care clientul, dar și dezvoltatorul poate avea siguranța că aplicația funcționează cum trebuie.

Espresso

Espresso este un framework de testare pentru aplicațiile Android și este în principal utilizat pentru crearea de teste automate pe interfața utilizator. Acest framework oferă o sintaxă simplă și ușor de înțeles și implementat, de asemenea, are interacțiune directă cu elementele de pe interfață prin creare de solicitarea diferitelor acțiuni precum click(), isDisplayed(), type() folosindu-se de id-urile pus la fiecare element din XML.

Prin testarea automată, aplicația poate fi testată atât pentru integritate, prin faptul că se asigură o funcționare continuă a tuturor funcționalităților oferite de aplicație,

dar și prin testarea performanței, întrucât testele sunt repetate de multe ori și se poate observa modul în care aplicația răspunde în cazul unu număr mare de date sau în cazul unor serii de acțiuni mai rapid executate pe interfață.

Capitolul 5. Proiectare de detaliu și implementare

5.1. Arhitectura aplicației

5.2. Schema generală

5.3. Cazuri de utilizare

5.4. Baza de date

5.4.1. Model-diagrama

5.4.2. Stocare

5.5. Aplicația de mobil

5.5.1. Activități

5.5.2. Legătura cu baza de date

5.5.3. Biblioteci utilizate

5.6. Modelul de recomandare

5.6.1. Preprocesare date

5.6.2. Crearea modelului

5.6.3. Antrenarea modelului

5.6.4. Rezultate

5.6.5. Postprocesare date si deploy

5.7. Adăugare model în aplicație

5.7.1. Actualizare

Capitolul 6. Testare și validare

6.1. Testare automata a aplicației

6.1.1. Testare creare cont și autentificare

6.1.2. Testare gestionare cărți

6.2. Testare manuală a aplicației

6.3. Testare modelului sistemului de recomandare

6.3.1. Analizare date returnate

Împreună cu cele două capitole **precedente** trebuie să reprezinte aproximativ 70% din total.

Capitolul 7. Manual de instalare și utilizare

7.1. Resurse necesare

7.2. Instalare aplicație

7.3. Manual de utilizare al aplicației

În secțiunea de Instalare trebuie să detaliați resursele software și hardware necesare pentru instalarea și rularea aplicației, precum și o descriere pas cu pas a procesului de instalare.

Instalarea aplicației trebuie să fie posibilă pe baza a ceea ce se scrie aici.

În acest capitol trebuie să descrieți cum se utilizează aplicația din punct de vedere al utilizatorului, fără a menționa aspecte tehnice interne.

Folosiți capturi ale ecranului și explicații pas cu pas ale interacțiunii.

Folosind acest manual, o persoană ar trebui să poată utiliza produsul vostru.
Între 1 și 5 pagini.

Capitolul 8. Concluzii

8.1. Contribuții personale

8.2. Analiza rezultatelor

8.3. Dezvoltări și îmbunătățiri ulterioare

Între 1 și 2 pagini.

Capitolul ar trebui sa conțină (nu se rezumă neapărat la):

- un rezumat al contribuțiilor voastre
- analiză critică a rezultatelor obținute
- descriere a posibilelor dezvoltări și îmbunătățiri ulterioare

Bibliografie

[1] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, “Using collaborative filtering to weave an information tapestry,” *Communications of the ACM*, vol. 35, no. 12, pp. 61–70, Dec. 1992. [Online]. Available: https://doi.org/10.1145/138859.138867

[2] A. Sarkar, A. Goyal, D. Hicks, D. Sarkar, and S. Hazra, “Android application development: A brief overview of android platforms and evolution of security systems,” in *2019 Third International conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*. IEEE, Dec. 2019. [Online]. Available: https://doi.org/10.1109/i-smac47947.2019.9032440

[3] Kyusik Chung. ”Announcing Goodreads Personalized Recommendations,” Sep. 15, 2011. [Online]. Available: https://www.goodreads.com/blog/show/303-announcing-goodreads-personalized-recommendations

[4] Otis Chandler. ”Recommendations And Discovering Good Reads,” Mar. 10, 2011. [Online]. Available: https://www.goodreads.com/blog/show/271-recommendations-and-discovering-good-reads

[5] LibraryThing members. ”What makes LibraryThing LibraryThing?,” Apr. 3, 2012. [Online]. Available: https://blog.librarything.com/2013/04/what-makes-librarything-librarything/

[6] C. Rana and S. Jain, “Building a book recommender system using time based content filtering,” vol. 11, pp. 27–33, 02 2012.

[7] Amazon. ”Amazon Personalize”. [Online]. Available: https://aws.amazon.com/personalize/

[8] K. Zwaaf, “Libib,” *Technical Services Quarterly*, vol. 36, no. 3, pp. 323–324, Jul. 2019. [Online]. Available: https://doi.org/10.1080/07317131.2019.1621573

Anexa A. Secțiuni relevante din cod

```
 /** Maps are easy to use in Scala. */
object Maps {
  val colors = Map("red" -> 0xFF0000,
                   "turquoise" -> 0x00FFFF,
                   "black" -> 0x000000,
                   "orange" -> 0xFF8040,
                   "brown" -> 0x804000)
  def main(args: Array[String]) {
    for (name <- args) println(
      colors.get(name) match {
        case Some(code) =>
          name + " has code: " + code
        case None =>
          "Unknown color: " + name
      }
    )
  }
}
```

Anexa B. Alte informații relevante (demonstrații etc.)

Se va elimina dacă nu există

Anexa C. Lucrări publicate (dacă există)

Se va elimina dacă nu există