MINISTERUL EDUCAȚIEI NAȚIONALE

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

OnlineGuide

LUCRARE DE LICENŢĂ

Absolvent: **Ionuț-Adrian MUTHI**

Coordonator ştiinţific: **Asis. Ing. Cosmina IVAN**

2019

MINISTERUL EDUCAȚIEI NAȚIONALE

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

DECAN,
Prof. dr. ing. Liviu MICLEA

DIRECTOR DEPARTAMENT,
Prof. dr. ing. Rodica POTOLEA

Absolvent: **Muthi Ionut Adrian**

OnlineGuide

1. **Enunţul temei:** *Proiectul propune o aplicație Android care funcționează pe post de guide TV în mediul online și ofera informații despre articole, link-uri catre site-uri care contin surse video și un forum care permite utilizatorilor să socializeze.*

2. **Conţinutul lucrării:***Cuprins, Introducere, Obiectivele proiectului, Studiu bibliografic, Analiză și fundamentare teoretică, Proiectare de detaliu și implementare, Testare și validare, Manual de instalare și utilizare, Concluzii, Bibliografie, Anexe.*

3. **Locul documentării**: Universitatea Tehnică din Cluj-Napoca, Departamentul Tehnologia Informatiei

4. **Consultanţi**: Asis. Ing. Cosmina IVAN

5. **Data emiterii temei:** 1 noiembrie 2018

6. **Data predării:** 8 iulie 2019

Absolvent: _____

MINISTERUL EDUCAȚIEI NAȚIONALE

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

Coordonator ştiinţific: _____

Declaraţie pe proprie răspundere privind
autenticitatea lucrării de licenţă

Subsemnatul(a)_____
_____,
legitimat(ă) cu _____ seria _____ nr. _____
CNP _____, autorul lucrării
_____
_____
_____elaborată în vederea susţinerii examenului de finalizare a studiilor de licență la Facultatea de Automatică și Calculatoare, Specializarea _____ din cadrul Universităţii Tehnice din Cluj-Napoca, sesiunea _____ a anului universitar _____, declar pe proprie răspundere, că această lucrare este rezultatul propriei activităţi intelectuale, pe baza cercetărilor mele şi pe baza informaţiilor obţinute din surse care au fost citate, în textul lucrării, şi în bibliografie.

Declar, că această lucrare nu conţine porţiuni plagiate, iar sursele bibliografice au fost folosite cu respectarea legislaţiei române şi a convenţiilor internaţionale privind drepturile de autor.

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

In cazul constatării ulterioare a unor declaraţii false, voi suporta sancţiunile administrative, respectiv, *anularea examenului de licenţă*.

Data

_____

Nume, Prenume

_____

Semnătura

Capitolul 1. Introducere – Contextul proiectului

Proiectul propune crearea și implementarea unei aplicații de tip Android care permite utilizatorului să acceseze informații relevante despre filmele, serialele și anime-urile urmărite de acesta.

1.1. Contextul proiectului

În ziua de azi, din ce în ce mai multe persoane aleg să vizioneze filme, seriale sau anime-uri în mediul online, datorită flexibilității pe care acesta o oferă și a posibilității de vizionare nelimitat de context.

Aplicația își propune să ofere utilizatorului un mediu unde poate accesa în orice moment informații relevante despre filmul sau serialul preferat. Un aspect important este accesul ușor, prin internet, al utilizatorilor. După crearea unui profil, utilizatorul va putea adăuga în lista de preferințe filmele, serialele sau anime-urile urmărite la momentul actual și va primi notificări atunci când episoade noi apar sau dacă este adăugat un film care corespunde cu preferințele sale.

Avantajul acestui tip de aplicație este că utilizatorul nu mai este nevoit reținăsau să caute singur data apariției unui film sau serial.Totodată, aplicația îi va oferi utilizatorului recomandări generate în funcție de articolele din lista acestuia de preferințe din cadrul aplicației.

1.2. Motivația

Flexibilitatea și rapiditatea sunt cuvinte specifice zilelor noastre. Tot mai multă lume își dorește să acceseze rapid și ușor informația dorită, dacă este posibil, chiar printr-un singur click. Această aplicație va oferi informații despre filme sau seriale într-un mod rapid și accesibil, fiind mereu la îndemâna utilizatorului pe dispozitivul mobil.

Informațiile care vor fi prezentate utilizatorului cuprind:

- În cazul serialelor și anime-urilor:
  - Descrierea: informații despre serialul sau anime-ul selectat;
  - Rating-ul: este ilustrat sub formă de stele care reprezintă valori în intervalul 1-5 și creat pe baza feedback-ului de la utilizatori;
  - Numărul de episoade apărute;
  - Data apariției următorului episod: data calendaristica aproximativa când episodul devine disponibil online.
- În cazul filmelor:
  - Descriere;
  - Rating.

1.3. Conținutul lucrării

În cadrul acestui subcapitol va fi expusă structura lucrării și o prezentare succintă a tipului de informație regăsit în fiecare capitol.

Capitolul 1. Introducere - Primul capitol cuprinde o introducere în tema proiectului și a structurii lucrării, pentru a facilita înțelegerea temei alese.

Capitolul 2. Obiectivele Proiectului – Acest capitol cuprinde o descriere a obiectivelor principale și secundare care se doresc atinse în dezvoltarea proiectului.

Capitolul 3. Studiu Bibliografic – În acest capitol se va realiza o analiză comparativă a sistemului propus în cadrul proiectului și a altor sisteme cu funcționalități similare pentru a determina cele mai eficiente metode de implementare.

Capitolul 4. Analiza și Fundamentare Teoretică – În acest capitol vor fi prezentate cerințele funcționale și non-funcționale și câteva de utilizare stabilite pentru fiecare tip de utilizator. De asemenea, vor fi descrise, pe scurt, tehnologiile utilizate în dezvoltarea aplicației.

Capitolul 5. Proiectare de Detaliu și Implementare – Acest capitol va cuprinde o descriere detaliată a arhitecturii conceptuale a sistemului propus. Diagrama bazei de date, diagrama de pachete și clase și diagrama de deployment vor fi prezentate tot în acest capitol. Fiecare modul și componentele sale vor fi descrise în detaliu.

Capitolul 6. Testare și Validare – În acest capitol se vor prezenta modurile în care a fost efectuată testarea diferitelor componente ale aplicației și a sistemului ca întreg.

Capitolul 7. Manual de Instalare și Utilizare – Acest capitol conține cerințele software necesare pentru a putea instala și utiliza aplicația și instrucțiuni de utilizare a aplicației.

Capitolul 8. Concluzii – În acest capitol vor fi expuse concluziile la care s-a ajuns în urma implementării proiectului și se vor descrie viitoarele posibilități de îmbunătățire a aplicației.

Capitolul 2. Obiectivele Proiectului

În acest capitol se va prezenta o descriere a obiectivelor principale și secundare care se doresc atinse în dezvoltarea proiectului și care asigură realizarea unui sistem cu utilizare simplă și eficientă.

2.1. Obiective principale

Obiectivul principal al proiectului este implementarea unui sistem sub forma unei aplicații Android, care să faciliteze căutarea și anunțarea utilizatorului cu privire la filmele, serialele și anime-uri urmărite și a posibilelor date de apariție ale acestora. Utilizatorul va primi o notificare atunci când un nou episod/ film a apărut și, prin accesarea episodului/ filmului, va fi direcționat către o serie de link-uri unde acesta poate fi vizionat. De asemenea utilizatorul poate interacționa cu alți utilizatori prin intermediul unui forum. Pentru a putea viziona articolele dorite utilizatorul o să aibă acces la link-uri către site-urile unde se afla sursa video a articolelor, aplicația oferind mai multe variante din care acesta să poată alege.

2.2. Obiective generale

În procesul de dezvoltarea a unei aplicații de tip Android este necesar să luăm în considerare unele caracteristici cum ar fi:

Eficiența utilizării aplicației: interfața aplicației trebuie să fie simplă, ușor de navigat, cât mai prietenoasă, organizată.

Funcționalitățile sistemului în funcție de tipul de utilizator sunt:

- Utilizatorul standard:
  - Poate căuta filme, seriale sau anime-uri disponibile în aplicație;
  - Poate adăuga un articol în lista de preferințe;
  - Poate comunica cu alți utilizatori prin intermediul unui forum;
  - Poate accesa diverse site-uri pentru vizualizarea articolelor dorite prin link-uri
  - Se poate loga utilizând contul Google;
  - La adăugarea unei intrări în lista de preferințe poate gestiona progresul vizualizării în următoarele moduri:
    - ”Plan to watch” : articolul încă nu a fost vizionat;
    - ”Watching” : utilizatorul este în proces de vizionare(specific serialelor și anime-urilor). Poate selecta și progresul exprimat prin numărul de episoade
    - ”Watched” : articolul selectat a fost deja vizionat de utilizator.

- Utilizatorul administrator:
  - Gestionează procesul de scraping prin intermediul aplicației Web Scraper
  - Poate face popularea bazei de date;
  - Poate adăuga noi site-uri;
  - Poate face actualizare pe :
    - Filme;
    - Seriale;
    - Anime-uri.

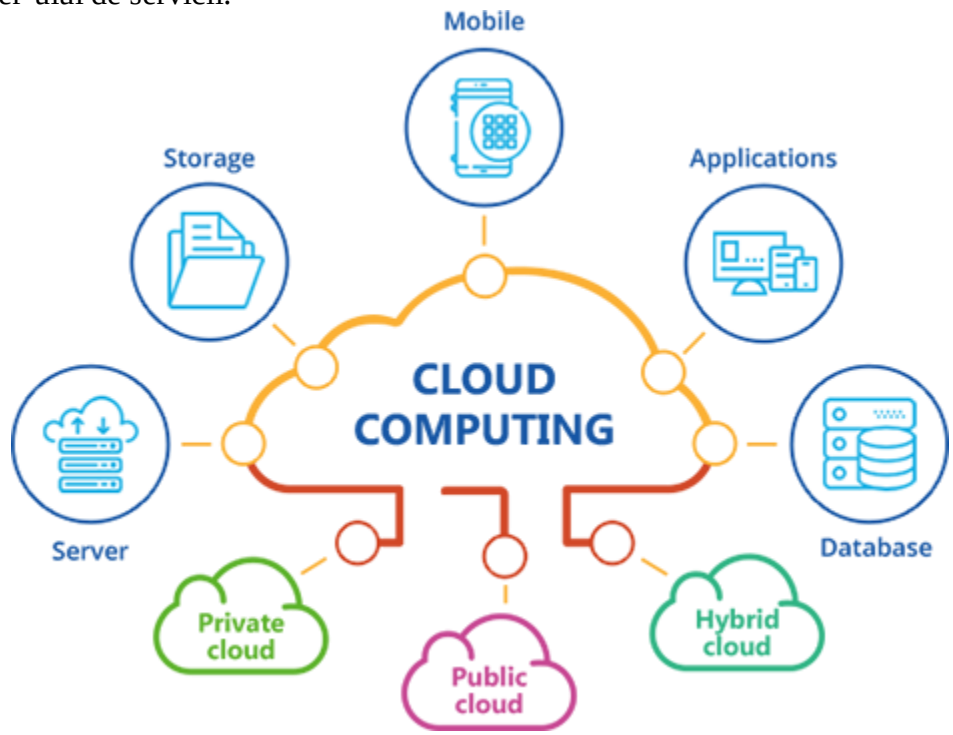
Capitolul 3. Studiu Bibliografic

În acest capitol vor fi prezentate, analizate și evaluate câteva sisteme reprezentative pentru oferirea de informații cu privire la filme/ seriale. Aceste sisteme reprezintă modele pentru realizarea proiectului propus deoarece vizează aceeași temă și cuprind operații specifice.

3.1. Cloud computing

Conform cărții lui George Reese [5] cloud este mai mult decât internet, cloud este locul în care găsești tehnologia de care ai nevoie când ai nevoie și pentru cât timp ai nevoie. Unul din avantajele serviciilor de tip cloud este faptul că plătești doar ce și cât folosești. Cloud poate oferii atât software cât și infrastructura, poate fi o aplicație accesată prin internet sau un server.

Conform NIST [6] (National Institute of Standards and Tehnology) cloud computing reprezintă un model care permite acces omniprezent, convenabil și la cerere la o rețea comună de resurse configurabile (de exemplu: servere, depozitare de date, aplicații, servicii) care pot fi oferite rapid cu un management minim și cu interacțiunea provider-ului de servicii.



Figură 3.1- Diagrama conceptuală a Cloun computing

3.1.1. Caracteristici esențiale

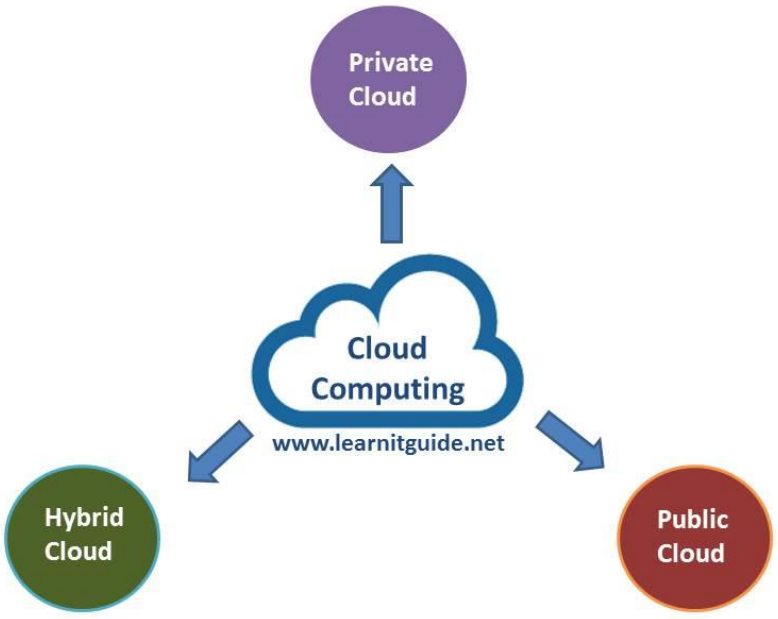
- Aplicațiile sau serviciile software sunt stocate la distanță
- Utilizatorul are accesa la diverse servicii sau aplicații software via Internet
- În majoritatea cazurilor utilizatorul nu trebuie să instaleze nimic pe mașina gazdă, totul se face din browser

- Acces și administrare, bazat pe rețea, la software
- Costuri reduse pentru implementare și întreținere
- Mobilitate crescută pentru forța de lucru la nivel global
- Infrastructuri flexibile și scalabile
- Permite focusare asupra inovării în loc de întreținere și implementare
- Disponibilitate crescută pentru aplicații de calcul de performanță înaltă pentru afaceri medii și mici

3.1.2. Modele de deployment cloud

Cloud computing are câteva modele de deployment, fiecare cu caracteristici specifice pentru agenții care doresc să treacă la servicii de tip cloud.

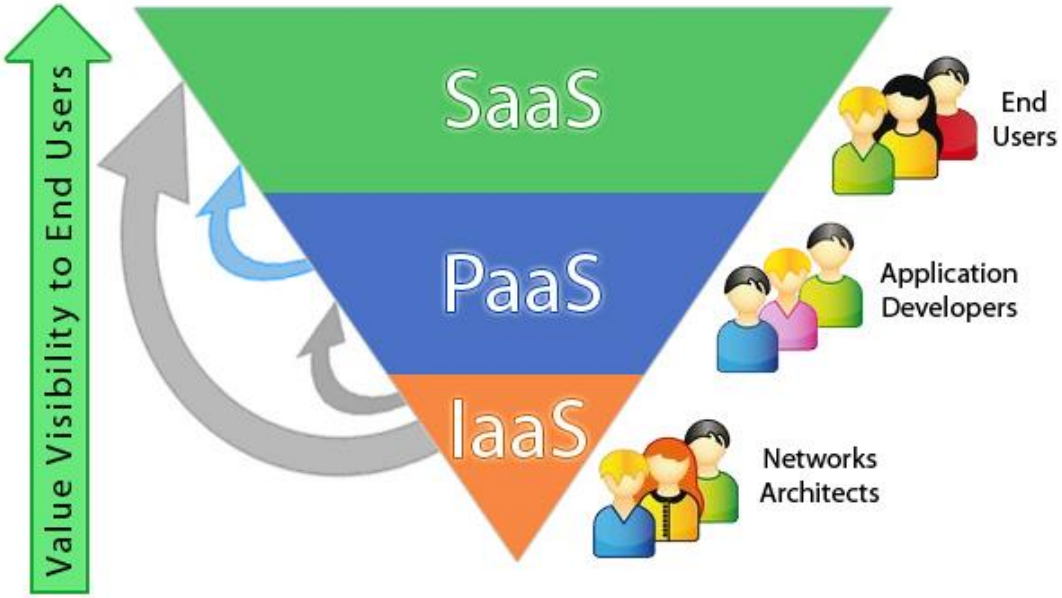
- Cloud public: folosit de organizații care oferă servicii către publicul general sau către un grup mare de industrii
- Cloud privat: folosind de o singură organizație. Poate fi deținut și operat de organizație, o parte terță sau o combinație a celor două.
- Cloud hibrid: reprezintă o îmbinare a două sau mai multe cloud-uri care rămân entități unice dar care comunică prin tehnologii standardizate.



Figură 3.2- Tipurile de Cloud coputing

3.1.3. Model de servicii cloud

Principalele modele de servicii cloud declarate de NIST sunt SaaS, PaaS, IaaS.



Figură 3.3 – Structura ierarhică a serviciilor oferite de cloud

- Software as a Service (SaaS)
  - Mai este cunoscut si ca Application-as-a-sevice(AaaS)
  - Aplicația este livrată peste o platformă a Web-ului la utilizatorul final, în mod tipic prezentând aplicația printr-un navigator
  - Model în care o aplicație e găzduită ca serviciu pentru clienții care o accesează via Internet
  - Aplicațiile sunt livrate printr-un navigator la mii de clienți utilizând o arhitectură multi-utilizator
  - Exemple de utilizare a modelului SaaS:
    - Administrarea resurselor clienților
    - Conferințe video
    - Administrarea de servicii IT
    - Administrarea de conținut Web
  - Caracteristici cheie :
    - Aplicațiile sau serviciile sunt stocate la distanță
    - Un utilizator poate accesa aceste servicii sau aplicații software via Internet
    - În majoritatea cazurilor, un utilizator nu trebuie să instaleze nimic pe mașina gazdă
    - Acces și administrare la software bazat pe rețea
- Platform as a Service (PaaS)
  - Platforme de dezvoltare sunt ofere ca servicii
  - Spre deosebire de platformele de dezvoltare tradiționale PaaS:
    - Suportă mai mulți developeri în același timp
    - Asigură load balancing și fail recovery
    - Oferă management integrat al resurselor

- Se plătește doar ce și cât se utilizează

- Serviciile PaaS includ
  - Proiectarea aplicațiilor
  - Dezvoltare
  - Testare
  - Lansare
  - Găzduire
  - Colaborare între echipe
  - Integrarea serviciilor Web
  - Integrare de baze de date
  - Securitate
  - Scalabilitate
  - Stocare
  - Administrarea stărilor și versionare

❖ Infrastructure as a Service (IaaS)
- Spre deosebire de SaaS și PaaS nu oferă aplicații ci hardware
- Resurse oferite:
  - Spațiu server
  - Echipament de rețea
  - Memorie
  - Cicluri CPU
  - Spațiu de stocare

3.2. Dezvoltarea aplicațiilor Android

Conform cursului [1] Android este un sistem de operare mobil bazat pe o versiune modificată de Linux și biblioteci Java. Acesta este un sistem open-source dezvoltat de Google care are ca avantaj abordarea unitară pentru dezvoltarea aplicațiilor. Acest lucru presupune că o aplicație dezvoltată conform API-ului Android va putea rula pe mai multe dispozitive pe care este instalat sistemul de operare respectiv.

Arhitectura sistemului de operare Android cuprinde cinci secțiuni grupate pe patru nivele:

- Kernelul Linux conține driver-ele pentru diferitele componente hardware, fiind responsabil de gestionarea proceselor, memoriei, perifericelor, dispozitivelor de intrare/ ieșire.
- Bibliotecile (user-space) conțin codul care oferă principalele funcționalități ale sistemului de operare, făcând legătura între kernel și aplicații.
- Motorul Android rulează serviciile de platformă, precum și aplicațiile care le utilizează, fiind reprezentat de:
  - API este mașina virtuală Java implementată începând cu versiunea 5.0, folosind un tip de compilare AOH (Ahead of Time) în care bytecode-ul este transpus în cod mașină la momentul instalării, astfel încât acesta este executat direct de mediul dispozitivului mobil.
  - Zygote este procesul care gestionează toate aplicațiile, fiind lansat în

-
  - execuție împreună cu sistemul de operare.
- Cadrul pentru aplicații expune diferitele funcționalități ale sistemului de operare Android către programatori.
- LA nivelul de aplicații se regăsesc atât produsele împreună cu care este livrat dispozitivul mobil, dar și produsele instalate de pe Play Store sau dezvoltate de programatori.

Pentru a dezvolta o aplicație Android este necesar să dispunem de următoarele componente:

1. Kit-ul de dezvoltare pentru limbajul de programare Java.
2. SDK-ul de Android pentru care se descarcă definițiile corespunzătoare unuia sau mai multor nivele API.
3. Un mediu integrat de dezvoltare (IDE).
4. Un dispozitiv pe care să se ruleze aplicațiile. Pentru a rula o aplicație pe un dispozitiv mobil fizic, trebuie să se activeze posibilitatea de depanare prin USB, considerând pașii următori: Setări → Sistem → Opțiuni Dezvoltator.

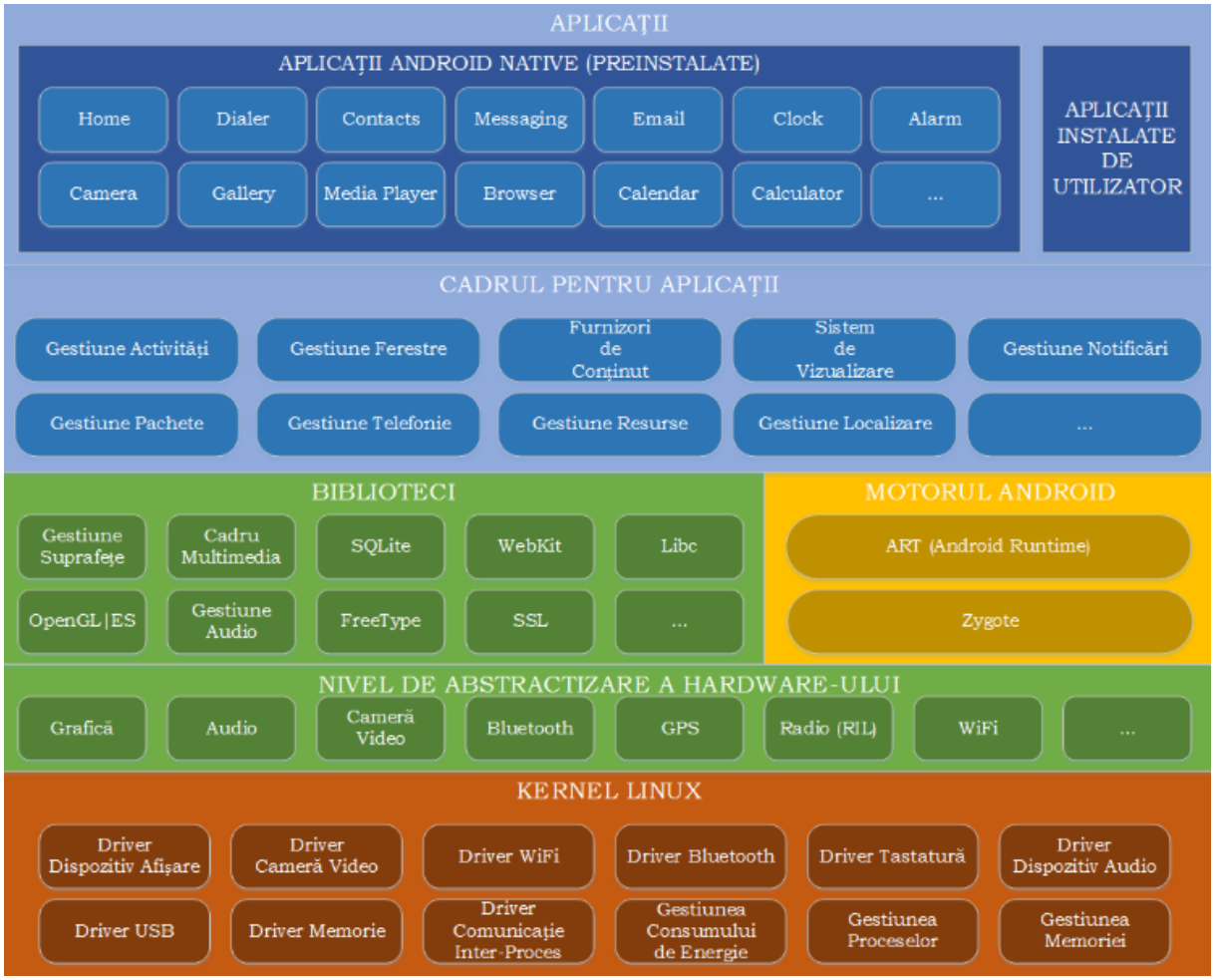


Figura 3. 4 Arhitectura sistemului de operare Android

3.3. Arhitectura sistemului

Arhitectura sistemului reprezentată în Figura 3.2 este formată din:

- Web Scraper: unealta online de scraping
- Server local : folosit de administrator pentru încărcarea datelor în Firebase
- Firebase: serviciu cloud care face
  - Stocarea datelor
  - Autentificarea utilizatorului prin Google
  - Notificarea utilizatorului prin Cloud Messaging
  - Forum folosind In-App Messaging
- Aplicație Android: folosită de utilizatori pentru a beneficia de funcționalitățile aplicației

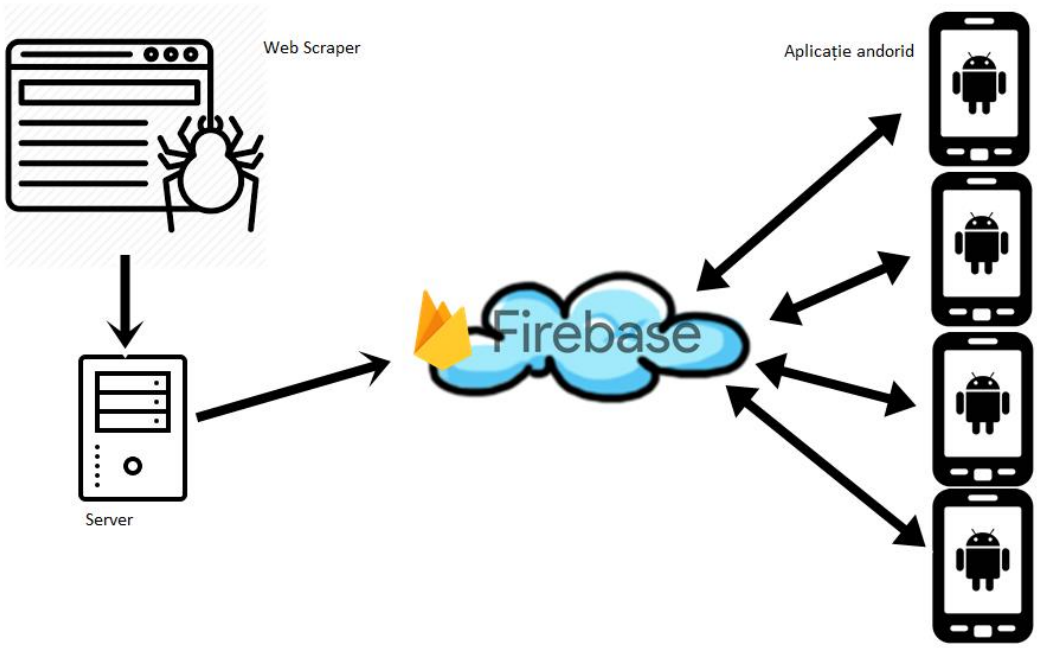


Figura 3. 5 Arhitectura conceptuală generală sistemului

3.4. Sisteme similare

Consumatorul de media din ziua de azi nu mai este limitat la 30-40 de canale TV oferite de abonamentul acestuia la o anume rețea, ci are la dispoziție întreg internetul care îi permite vizionarea de filme, seriale și anime-uri fără a-i impune restricții de tipul oră sau locație. Un prim motiv al migrării în mediul online este diversitatea de site-uri pe care utilizatorul le poate alege pentru a accesa serviciile dorite de la o varietate de furnizori prin intermediul diferitor site-uri. Totodată această diversitate poate fi privită și ca un dezavantaj deoarece atunci când are foarte mult variante, utilizatorul poate deveni confuz, neștiind ce să aleagă.

Deoarece utilizatorii de televiziune migrează spre mediul online diferite platforme care permit vizionarea de filme, seriale și anime-uri au început să domine internetul (Netflix, HBO GO, Crunchyroll). Dezavantajul acestor platforme este că utilizatorul este nevoit să plătească pentru serviciile oferite prin achiziționarea unui pachet care conține doar un număr limitat de articole oferite de platforma respectiva astfel că utilizatorul are

nevoie de mai multe abonamente pentru a-și satisface nevoile și pentru a urmării ceea ce își dorește. Acest lucru se poate dovedi costisitor și atunci utilizatorii vor prefera site-urile online în defavoarea acestor platforme deoarece acestea le oferă posibilitatea de a alege dintr-o gama mai mare de articole.

Pentru a ajuta utilizatorii sa gestioneze filmele, serialele sau anime-urile urmărite au fost create numeroase aplicații care oferă informații cu privire la data apariției acestora, dar care nu oferă și sursa video unde utilizatorii pot urmări articolul dorit.

Aplicația propusă de mine oferă informații despre o diversitate de filme/ seriale/ anime-uri selectate în funcție de cele mai noi apariții și rating-uri.

În crearea aplicației am avut în vedere analiza comparativă prezentată mai jos, a unor aplicații cu funcționalități similare pentru a extrage un set de funcționalități potrivite aplicației.

a. Next Episode[1] este o aplicație Android care permite utilizatorului să vizioneze trailer-ul filmului/ serialului, să adauge în lista de preferințe anumite seriale/ filme. De asemenea, utilizatorul primește notificări atunci când un serial/ film pe care îl urmărește a apărut online și recomandări bazate pe preferințele sale.

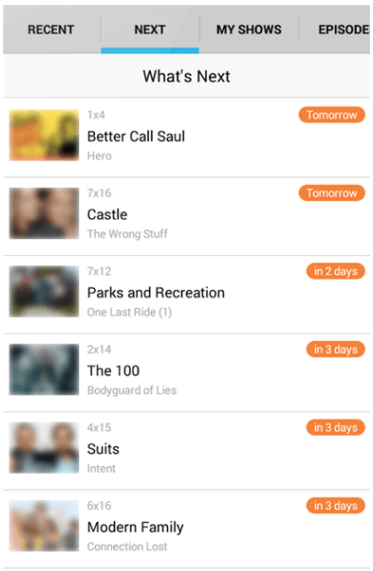


Figura 3. 6- Next Epsiode

b. TV Time[2] Track and Discover Shows este o aplicație android care permite utilizatorilor să urmărească data apariției episoadelor unor seriale/ anime-uri salvate în lista de preferințe. De asemenea, utilizatorii au posibilitatea de a posta comentarii care au legătură cu episodul vizionat și de a marca episoadele ca vizionate sau nevizionate.

[1] https://play.google.com/store/apps/details?id=net.nextepisode.android
[2] https://play.google.com/store/apps/details?id=com.tozelabs.tvshowtime

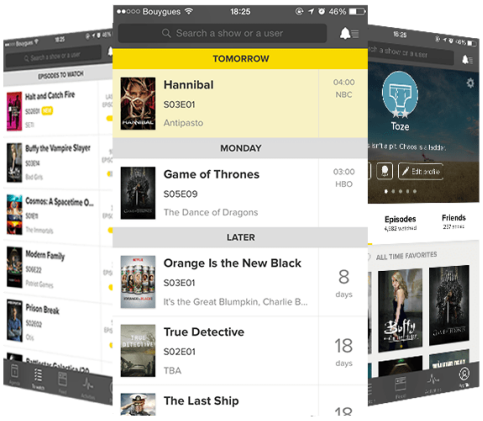


Figura 3. 7- TV Time

c. LiveChart.me[3] este o aplicație care permite utilizatorilor să urmărească când va apărea un nou episod din anime-ul preferat. Această aplicație oferă o scurtă descriere a anime-ului, articole în care acesta a fost menționat și trailer-ele apărute. Această aplicație ajută utilizatorul să își gestioneze anime-urile urmărite și oferă diverse surse alternative de informații externe aplicației.

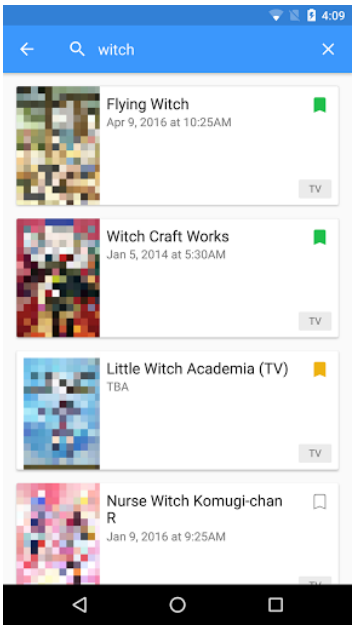


Figura 3. 8- LiveChart.me

[3] https://play.google.com/store/apps/details?id=me.livechart.android

Funcționalitate	OnlineGuide	MALClient	Crunchyroll	LiveChart	Next Episode	TV Time
Logare	Da	Da	Da	Da	Nu	Da
Notificări	Da	Da	Da	Da	Da	Da
Listă preferințe	Da	Da	Da	Da	Da	Da
Video	Nu	Nu	Da	Nu	Nu	Nu
Posibilitate de vizionare	Da	Nu	Da	Nu	Nu	Nu
Recomandări	Da	Da	Nu	Nu	Nu	Nu
Căutare	Da	Da	Da	Da	Nu	Da
Sortare	Da	Da	Da	Da	Da	Da
Selectarea episoadelor vizionate	Da	Da	Nu	Nu	Nu	Nu

Tabelul 3. 1

Descrierea funcționalităților:
- Logare: utilizatorul se poate loga;
- Notificări: utilizatorul primește notificare legate de articolele din aplicație;
- Lista preferințe: utilizatorul poate crea o listă care conține articolele favorite;
- Video: utilizatorul are acces la video în cadrul aplicației;
-Posibilitate de vizionare: utilizatorul are posibilitatea de a accesa video-ul în afara aplicației;
- Recomandări: utilizatorul primește recomandări cu alte articole din aplicație
- Căutare: utilizatorul poate căuta un articol pe baza numelui;
- Sortare: utilizatorul poate sorta articolele după diverse criterii cum sunt genul sau rating-ul;
-Selectarea episoadelor vizionate: utilizatorul poate gestiona serialele sau anime-urile în curs de vizionare selectând câte episoade a vizionat.

3.5. Concluzii

În concluzie aplicația propusă de mine OnlineGuide la fel ca LiveChart.me și TV Time ajuta utilizatorul să gestioneze articolele urmărite.

Analizând aplicația LiveChart.me am observat o serie de caracteristici pe care le are în comun cu aplicația propusă spre implementare printre care se numără:

- Informații despre data apariției următorului episod
- Posibilitatea de a selecta progresul vizionării articolului la adăugarea în lista de preferințe
- Oferă notificări la apariția unui nou episod al unui articol din lista de preferințe

Dezavantajele aplicației LiveChart.me sunt că nu oferă o modalitate de acces către video-ul articolului și faptul că tipul articolelor conținute este strict din categoria anime-urilor.

Caracteristicile relevante ale aplicației TV Time sunt:

- Informații despre data apariției următorului episod
- Forum pentru fiecare din articole
- Conține filme, seriale și anime-uri

Dezavantajele aplicației TV Time sunt că nu oferă o modalitate de acces către video-ul articolului, nu permite utilizatorului să gestioneze progresul de vizionare a articolelor din lista sa.

Capitolul 4. Analiză și Fundamentare Teoretică

În acest capitol se vor expune cerințele funcționale și non-funcționale ale aplicației propuse, din punct de vedere funcțional și non-funcțional. Sunt prezentate și principalele cazuri de utilizare și tehnologiile folosite pentru dezvoltarea aplicației.

4.1. Cerințele sistemului

Cerințele sistemului reprezintă atât funcționalitățile acestuia reprezentate de cerințele funcționale, cât și constrângerile care se aplică sistemului care determină funcționarea corectă a acestuia reprezentate de cerințele non-funcționale

4.1.1. Cerințe funcționale ale sistemului

Cerințele funcționale reprezintă descrierea completă a capabilităților sistemului din perspectiva utilizatorilor și acțiunile pe care aplicația le realizează în anumit circumstanțe.

Principalele cerințe funcționale sunt prezentate în Tabelul 4.1

Identificator	Descriere
CF 1	Logare/Delogare utilizator folosind Google
CF 2	Căutare articole
CF 3	Sortare articole în funcție de anumite criterii
CF 3.1	Sortare după gen
CF 3.2	Sortare după tip (film/serial/anime)
CF 3.3	Sortare după rating
CF 3.4	Sortare după cele mai vizionate din ultima luna
CF 3.5	Sortare după cele mai recent adăugate

CF 4	Acțiuni asupra articolelor
CF 4.1	Rating articol
CF 4.2	Accesarea link-ului pentru vizionare
CF 4.3	Primirea de notificări la apariția articolelor din lista de preferințe
CF 4.4	Adăugarea de articole în lista de favorite
CF 4.4.1	Selectarea progresului de vizionare a articolului la adăugarea acestuia in lista de preferințe (watching/plan to watch/watched)
CF 4.4.2	Selecția numărului de episoade văzute la momentul adăugării in lista (pentru articole de tip seriale)
CF 5	Mecanism de recomandare
CF 5.1	Filtrarea genurilor predominante pentru user-ul curent pentru toate tipurile de articole Parcurgerea listei de preferințe a utilizatorului și determinarea pentru fiecare tip de articol (film/serial/anime) genul predominant pentru user-ul curent (peste 65% -70%);
CF 5.2	Căutare articole care se potrivesc criteriilor stabilite
CF 6	Comunicarea cu alți utilizatori prin intermediul unui forum.

Tabelul 4. 1

4.1.2. Cerințe non-funcționale ale sistemului

Cerințele non-funcționale reprezintă constrângeri aduse sistemului, reguli impuse care trebuie respectate.

În cadrul acestei aplicații cerințele non-funcționale sunt:

- **Utilizabilitatea** – reprezintă ușurința cu care sistemul este învățat și utilizat. Utilizatorul va reuși să folosească cât mai eficient sistemul dacă acesta respectă anumite standarde utilizate în majoritatea aplicațiilor Android.
- **Accesibilitatea** – un sistem trebuie să fie ușor accesibil. Accesibilitatea este asigurată prin faptul că aplicația implementată este accesibilă prin internet.
- **Mentenanță** – crearea unor soluții software ușor de ajustat și care pot adopta ușor modificări sau dezvoltări. Sistemul implementat poate fi ușor dezvoltat deoarece are o structură clară, simplă, bine definită astfel făcând modificările necesare pe viitor simplu de implementat.
- **Simplitatea** – opțiunile disponibile trebuie să fie simple și clare. Pentru a asigura această cerință se apelează la structurarea operațiilor în funcție de rolul deținut de utilizatori.
- **Disponibilitatea** – redată în procente, măsoară timpul în care aplicația poate fi utilizată. Sistemul este utilizabil neîntrerupt atât timp cât utilizatorul e conectat la internet, se vor putea aduce modificări persistente asupra datelor. Datorită serviciului de baze de date real-time Firebase, actualizările vor fi salvate în memoria cache și introduse în Cloud în momentul următoarei conectării la internet. Pentru alte funcționalități cum ar fi comunicarea prin intermediul forum-ului sau notificarea utilizatorului, este necesară conexiunea la internet.
- **Performanța** - reprezintă măsurarea criteriilor care analizează eficiența sistemului, cum ar fi timpul de răspuns, rata de procesare sau memoria utilizată de aplicație. În ceea ce privește sistemul propus de mine durata de timp necesară sistemului să pornească este aplroximativ 3 secunde.

- **Scalabilitatea** - capacitatea unui sistem de a suporta corect un volum mare de încărcare sau capacitatea sa de extindere. Un sistem este scalabil dacă se comporta similar chiar și când volumul de date pe care le prelucrează este mare. Prin serviciul de baze de date Firebase sunt permise 100000 de conexiuni concurente și 1000 de modificări pe secundă într-o singură bază de date. Scalarea dincolo de aceste limite necesită divizarea informațiilor în mai multe baze de date.

4.2. Cazuri de utilizare

Această secțiune prezintă rolurile acceptate de aplicație și cazurile de utilizare pentru fiecare tip de utilizatori. Analiza cazurilor de utilizare are rolul de a oferi o imagine globală asupra utilizării aplicației pentru a facilita înțelegerea funcționalităților acestuia.

4.2.1. Tipuri de utilizatori:

Pentru această aplicație sunt definite două tipuri de utilizatori:

- ❖ Utilizatorul administrator: se ocupă de gestionarea scraper-ului și de actualizarea și popularea bazei de date din Firebase pe partea de server local.
- ❖ Utilizatorul standard: are acces la aplicație și funcționalitățile acesteia

4.2.2. Diagrama cazurilor de utilizare

Diagrama cazurilor de utilizare ilustrează acțiunile pe care le poate realiza utilizatorul.

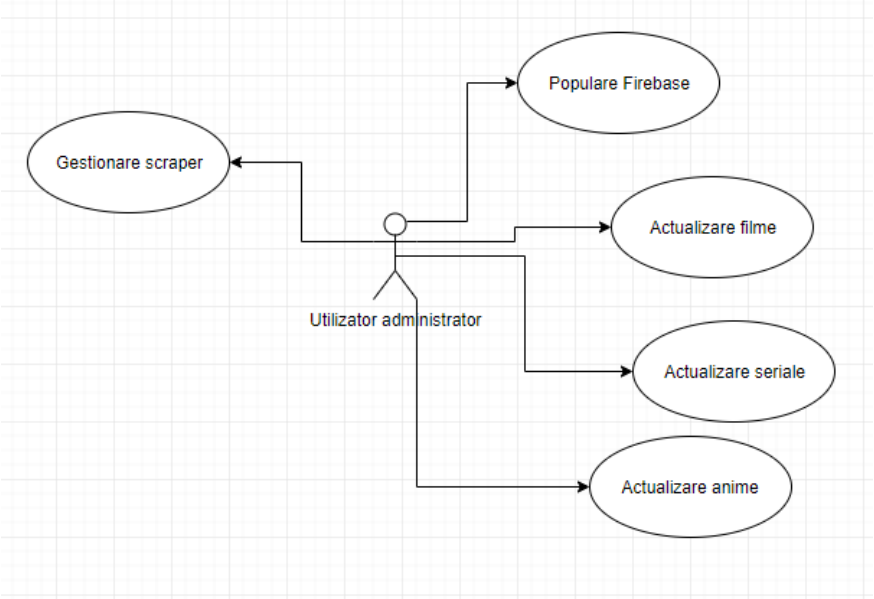


Figura 4. 1- Diagrama cazuri utilizare utilizator administrator

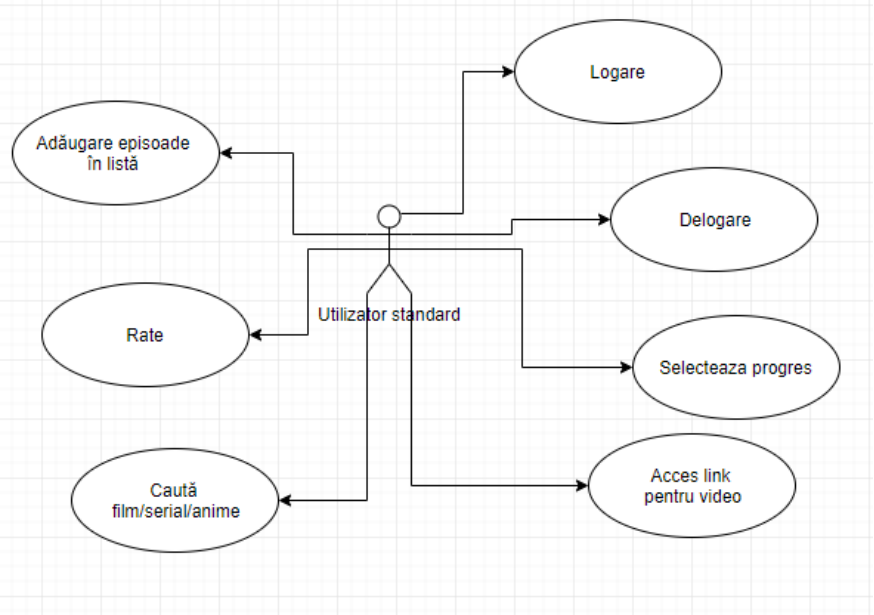


Figura 4. 2- Diagrama cazuri utilizare utilizator standard

4.2.3. Descrierea cazurilor de utilizare

❖ Caz utilizare 1

Nume caz de utilizare: Vizualizare filme de comedie

Actor principal: Utilizator

Descriere: Rolul acestui caz de utilizare este de a oferi utilizatorului o lista cu toate filmele care au ca gen comedie.

Precondiții:

- Utilizatorul este autentificat.

Post condiții:

- Utilizatorul a vizualizat lista și poate determina un curs de acțiune bazat pe informațiile obținute (adăugare în lista de preferințe sau accesare informații).

Scenariu principal:

1. Utilizatorul deschide aplicația
2. Utilizatorul selectează butonul GEN
3. Utilizatorul selectează genul dorit (comedie)
4. Utilizatorul primește o lista cu toate filmele din genul dorit (comedie)

❖ Caz utilizare 2

Nume caz de utilizare: Autentificare

Actor principal: Utilizator

Descriere: Rolul acestui caz de utilizare este de a descrie procesul de autentificare .

Precondiții:

- Utilizatorul trebuie să dețină un cont Google sau să fie dornic de a crea unul.

Post condiții:

- Utilizatorul e autentificat în aplicație

Scenariu principal:

1. Utilizatorul apasă butonul ”Sign in”
2. Utilizatorul alege contul Google dorit
3. Autentificarea a fost efectuată cu succes și utilizatorul e redirecționat către fereastra principală a aplicației

Scenariu alternativ:

1. Utilizatorul nu deține un cont Google
    i. Utilizatorul are posibilitatea de a crea un cont Google

❖ Caz utilizare 3

Nume caz utilizare: Delogare

Actor principal: Utilizator

Descriere: Rolul acestui caz de utilizare este de a descrie procesul de delogare.

Precondiții:

- Utilizatorul e autentificat.

Post condiții:

- Utilizatorul e delogat.

Scenariu principal:

1. Utilizatorul deschide meniul de utilizator
2. Utilizatorul apasă ”Log out”
3. Delogarea a fost efectuată cu succes și utilizatorul e redirecționat către pagina de autentificare

❖ Caz utilizare 4

Nume caz utilizare: Căutare articol
Actor principal: Utilizator
Descriere: Rolul acestui caz de utilizare este de a descrie procesul de căutarea unui articol în aplicație
Precondiții:

- Utilizatorul e autentificat

Post condiții:

- Rezultatul căutării e afișat pe ecranul aplicației

Scenariu principal:

1. Utilizatorul selectează bara de căutare
2. Utilizatorul introduce titlul sau o parte a titlului căutat
3. Aplicația afișează articolele care se potrivesc datelor introduse de utilizator

Scenariu alternativ:

1. Articolul căutat nu există în baza de date
    a. Aplicația afișează o listă goală

❖ Caz utilizare 5

Nume caz utilizare: Afișare listă preferințe
Actor principal: Utilizator
Descriere: Rolul acestui caz de utilizare este de a descrie procesul de accesare a listei de preferințe a utilizatorului
Precondiții:

- Utilizatorul e autentificat

Post condiții:

- Lista de preferințe e afișată pe ecranul aplicației

Scenariu principal:

1. Utilizatorul deschide meniul de utilizator
2. Utilizatorul apasă ”Listă preferințe”
3. Aplicația afișează articolele din lista de preferințe a utilizatorului

Scenariu alternativ:

1. Lista de preferințe e goală
    a. Aplicația afișează o listă goală

❖ Caz utilizare 6

Nume caz utilizare: Filtrare date după tipul de articol
Actor principal: Utilizator
Descriere: Rolul acestui caz de utilizare este de a descrie procesul de filtrare a datelor după categoria din care fac parte: filme, seriale sau anime-uri

Precondiții:
- Utilizatorul e autentificat

Post condiții:
- Lista de articole care se încadrează în tipul selectat de utilizator este afișată

Scenariu principal:
1. Utilizatorul deschide meniul cu opțiunile de tip articol
2. Utilizatorul selectează tipul dorit
3. Aplicația afișează articolele de tipul selectat de utilizator

❖ Caz utilizare 7

Nume caz utilizare: Filtrare date după genul articolului
Actor principal: Utilizator
Descriere: Rolul acestui caz de utilizare este de a descrie procesul de filtrare a datelor după genul din care face parte
Precondiții:
- Utilizatorul e autentificat

Post condiții:
- Lista de articole care se încadrează în genul selectat de utilizator este afișată

Scenariu principal:
1. Utilizatorul deschide meniul cu opțiunile de tip gen
2. Utilizatorul selectează genul dorit
3. Aplicația afișează articolele care se încadrează în genul selectat de utilizator

❖ Caz utilizare 8

Nume caz utilizare: Rate articol
Actor principal: Utilizator
Descriere: Roulul acestui caz de utilizare este de a descrie procesul de rating a unui articol
Precondiții:
- Utilizatorul e autentificat
- Articolul pentru care se dorește realizarea funcției de rate este selectat

Post condiții:
- Rating-ul articolului este actualizat

Scenariu principal:
1. Utilizatorul selectează numărul de stele pe care vrea să le acorde articolului
2. Utilizatorul apasă butonul ”RATE”
3. Aplicația actualizează valoarea rating-ului pe baza valorii introduse de utilizator în baza de date
4. Interfața utilizator e actualizată cu noua valoare

❖ Caz utilizare 9

Nume caz utilizare: Accesarea video-ului
Actor principal: Utilizator
Descriere: Rolul acestui caz de utilizare este de a descrie procesul de accesare la vide-ul articolului
Precondiții:

- Utilizatorul e autentificat
- Articolul pentru care se dorește accesarea link-ului e selectat

Post condiții:

- Utilizatorul e redirecționat către sursa externa unde se afla video-ul

Scenariu principal:

1. Utilizatorul deschide lista cu link-uri
2. Utilizatorul selectează link-ul dorit
3. Utilizatorul este redirecționat către site-ul corespunzător link-ului

❖ Caz utiliziare 10

Nume caz utilizare: Adăugare articol la lista de preferințe
Actor principal: Utilizator
Descriere: Rolul acestui caz de utilizare este de a descrie procesul de adăugare a unui articol în lista de preferințe a utilizatorului
Precondiții:

- Utilizatorul e autentificat
- Articolul care urmează să fie adăugat este selectat
- Articolul nu se află în lista utilizatorului

Post condiții:

- Articolul este adăugat în lista de preferințe utilizatorului

Scenariu principal:

1. Utilizatorul apasă butonul în formă de plus
2. Utilizatorul selectează progresul de vizionare
3. Utilizatorul apasă ”ADD”
4. Articolul este adăugat în lista utilizatorului
5. Butonul de adăugare nu mai este disponibil pentru articolul dorit

Scenariu alternativ:

1. Articolul se află deja în lista utilizatorului
    i. Butonul de adăugare nu mai este disponibil

❖ Caz utilizare 11

Nume caz utilizare: Postare mesaj pe forum
Actor principal: Utilizator
Descriere: Rolul acestui caz de utilizare este de a descrie procesul de adăugare a unui mesaj pe forumul aplicației
Precondiții:

- Utilizatorul e autentificat

Post condiții:

- Un nou mesaj este înregistrat pe forum

Scenariu principal:

1. Utilizatorul deschide meniul de utilizator
2. Utilizatorul selectează opțiunea ”Forum”
3. Utilizatorul introduce mesajul dorit
4. Utilizatorul apasă butonul ”POST”
5. Mesajul este afișat pe forum

4.3. Tehnologii utilizate în dezvoltarea sistemului

4.3.1. Google Sign-In

Modalitatea de logare folosită în aplicație va utiliza Google Sign-in[4] care oferă posibilitatea de a accesa date din contul de Google al utilizatorului cum ar fi nume, email, id. Utilizatorul se poate deloga de pe aplicație fără a fi nevoit sa se delogheze și de pe contul de Google printr-un buton de delogare.

Pentru a folosi acest serviciu e necesara o configurarea a proiectului in care ne va furniza un ID client pe care îl putem integra in proiect astfel fiind capabili sa utilizam serviciul de logare.

Motivul pentru care am optat pentru acest tip de logare e siguranța datelor si oportunitatea pe care o oferă utilizatorilor, aceea de a nu fi nevoit sa creeze un cont nou.

4.3.2. Firebase

Firebase[5] reprezintă un serviciu cloud creat pentru a furniza date pentru aplicații colaborative, în timp real. Este o platformă foarte utilizată pentru dezvoltarea aplicațiilor mobile și web deoarece oferă multe servicii care te scutesc de implementarea unui server auxiliar. Firebase devine serverul tău, API-ul tău și locul de stocare a datelor. Scopul Firebase, descris in cartea lui L. Moroney [3], este de a oferi uneltele și infrastructura de care ai nevoie pentru a crea cele mai bune aplicații, de a dezvolta o afacere de succes. Firebase nu reprezintă un înlocuitor al API-urilor existente pentru crearea aplicațiilor iOS, Android sau web. El este mai degrabă o îmbunătățire care îți oferă serviciile comune de care ai putea avea nevoie – bază de date back-end, autentificare sigură, mesagerie și multe altele. Asta este un beneficiu pentru că nu mai este necesară crearea lor astfel este posibil focus pe elementele care diferențiază aplicația de toate celelalte.

[4] https://firebase.google.com/docs/auth/android/google-signin
[5] https://firebase.google.com

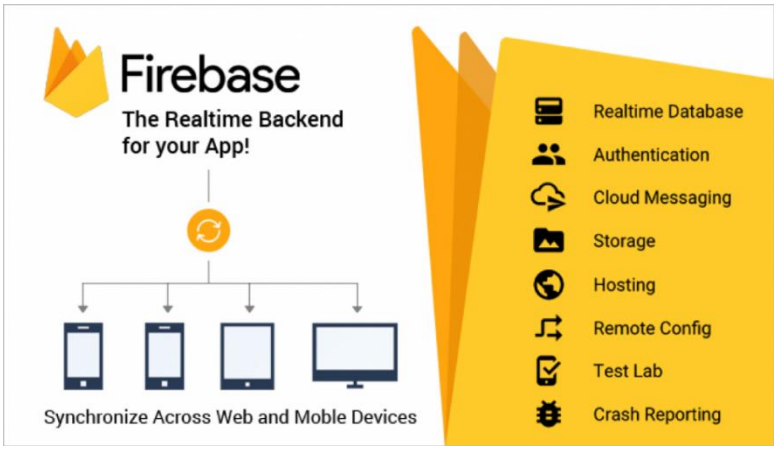


Figura 4. 3-Firebase servicii

Firebase este un Baas (Back-end as a service) care oferă numeroase beneficii, printre care:

- Ușurința de implementare;
- Firebase este deținut și creat în infrastructura Google, lucru care oferă o oarecare asigurare a performanței și calității serviciului oferit.
- Având numeroși utilizatori, este foarte ușor să primești ajutor.
- Firebase oferă atât de multe caracteristici care te ajută să salvezi timp și să te focusezi pe experiența utilizatorului.

Urmează o descriere în detaliu a 4 servicii principale utilizate în dezvoltarea acestei lucrări.

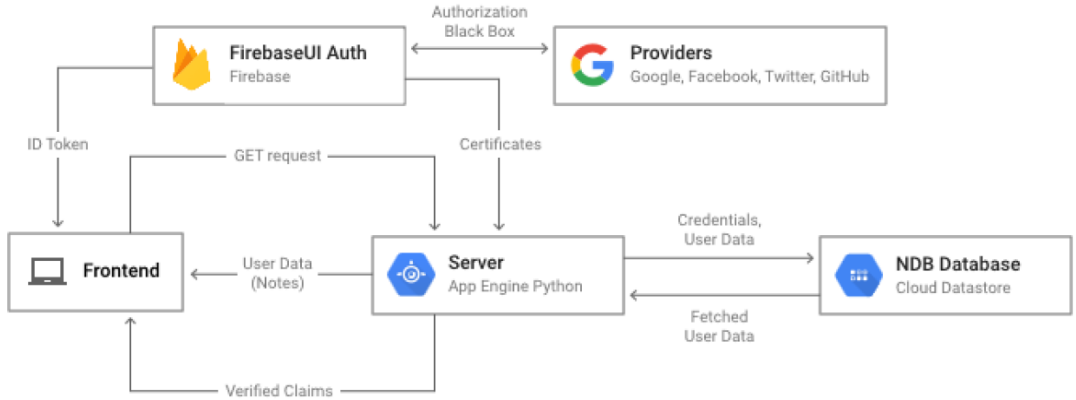
Firebase Authentication

Atunci când un anumit set de informații trebuie oferit unui anumit utilizator, acel utilizator trebuie să se identifice. Cea mai utilizată formă de identificare este autentificarea, care este oferită de Firebase Authentication într-un mod ușor de integrat în aplicație și permite autentificarea cu conturi deja existente cum sun cele de Google sau Facebook.

Majoritatea aplicațiilor au nevoie să cunoască identitatea utilizatorului. Cunoașterea identității utilizatorului permite aplicației să salveze datele acestuia în mod sigur in cloud și să ofere aceeași experiență personalizată pe toate dispozitivele utilizatorului. Firebase Authentication oferă servicii back-end, SDK ușor de utilizat și librării UI gata create pentru a autentifica utilizatorii aplicației. Oferă posibilitatea autentificării folosind parole, numere de telefon sau platformelor Google, Facebook, Twitter.

Pentru autentificarea în această aplicație se va folosi contul Google.

Diagrama din figura 4.4 prezintă modul de migrare a datelor între front-end și back-end folosind Firebase Authentication.

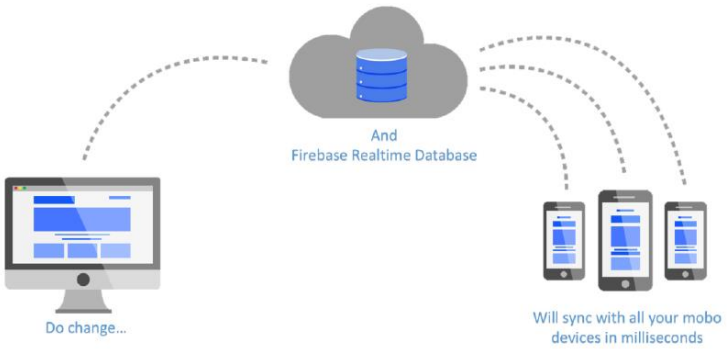


Figură 4.4- Diagrama de migrare a datelor între fornt-end și back-end cu Firebase Authentification

Firebase RealTime Database

Una din cele mai cunoscute facilități pe care Firebase le are este baza de date RealTime care sincronizează în timp real informația stocată ca JSON pentru fiecare client, urmând modelul din figura 4.5.

Aceasta este o bază de date NoSQL stocată în cloud. Aceasta oferă sincronizare între dispozitivele conectate și este utilizabilă chiar și atunci când nu există conectivitate între rețele printr-un cache local. Este o bază de date bazată pe evenimente care funcționează foarte diferit de baza de date SQL tradițională. De câte ori informațiile sunt schimbate în baza de date, evenimente sunt trimise către codul client și mai apoi se pot face modificări la interfața utilizatorului ca răspuns la aceste evenimente. Conține un limbaj bazat pe reguli de expresie, numit Firebase RealTime Database Security Rules, care definesc cum este structurată informația și ce utilizatori au acces la informații.



Figură 4.5- Firebase RealTime Database

Firebase Cloud Storage

În plus față de stocarea datelor, stocarea fișierelor de tip foto sau video poate fi utilă, iar tehnologia care realizează acest lucru este Firebase Cloud Storage. Firebase Cloud Storage este un sistem de stocare simplu si eficient din punct de vedere al costurilor, creat pentru Google. SDK Firebase pentru Cloud Storrage adaugă securitatea Google la fișierele încărcate și descărcate pentru aplicațiile Firebase, fără a ține cont de

calitatea rețelei. SDK se poate folosi pentru a stoca imagini, fișiere audio, video și alte tipuri de conținut generate către utilizator.

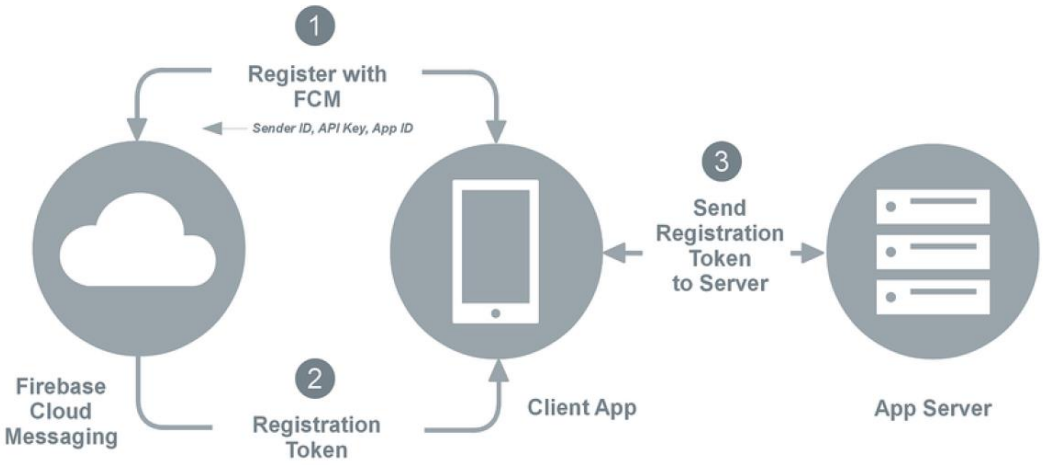
SDK Firebase pentru Cloud Storage se corelează cu Firebase Authentication pentru a identifica utilizatorii și pentru a oferi un limbaj declarativ de securitate care oferă posibilitatea de a seta accesul pentru fișiere individuale sau grupuri de fișiere, pentru a face fișierele publice sau private.

Firebase Cloud Messaging

Firebase Cloud Messaging oferă aplicației posibilitatea de a trimite notificări. Firebase Cloud Messaging este un serviciu interconectat care se ocupă de trimiterea, rutarea și linia mesajelor între aplicațiile server și aplicația de mobil a clientului. FCM este succesorul Google Cloud Messaging și este construit în Google Play Services.

Folosind FCM serverele aplicației pot trimite mesaje către un singur dispozitiv, către un grup de dispozitive sau către un număr de dispozitive abonate unui anumit subiect.

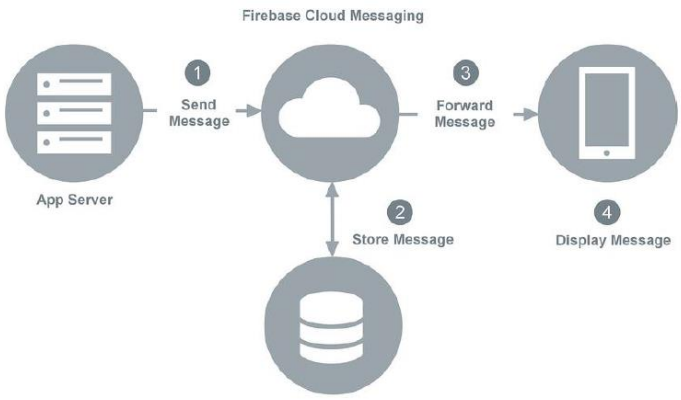
Înainte de a primi notificările aplicația clientului trebuie să fie înregistrată în FCM. Diagrama din figura descrie succesiunea evenimentelor pentru înregistrarea dispozitivului.



Figură 4.6- Firebase Cloud Messaging diagrama de înregistrare

Diagrama din figura descrie procesul prin care mesajele sunt transmise din aplicația server către aplicația client conform documentului oficial:

1. Aplicația server trimite mesajul către FCM.
2. Dacă dispozitivul clientului nu este disponibil, serverul FCM stochează mesajul într-o listă pentru a fi transmis mai târziu. Mesajele sunt reținute in stocarea FCM pentru maxim 4 săptămâni.
3. Când dispozitivul clientului este disponibil FCM trimite mesajul către aplicația client de pe dispozitiv.
4. Aplicația client primește mesajul de la FCM, îl procesează și îl afișează utilizatorului.



Figură 4.7- Firebase Cloud Messaging

Aplicația acestei lucrări folosește abonarea la un subiect pentru a trimite mesaje, deoarece este mai eficient să transmiți mesaje unor utilizatori abonați decât unui grup de dispozitive. Mesajele pot fi transmise folosind Firebase Console Notifications GUI.

4.3.3. Android

Android[6] este un sistem de operare mobil bazat pe o versiune modificată de Linux și biblioteci Java. Acesta este un sistem open-source dezvoltat de Google care are ca avantaj abordarea unitară pentru dezvoltarea aplicațiilor. Acest lucru presupune că o aplicație dezvoltată conform API-ului Android va putea rula pe mai multe dispozitive pe care este instalat sistemul de operare respectiv.

4.3.3.1. Android vs IOS

Conform articolului scris de Asokan M.[4] Android este un sistem de operare bazat pe Linux dezvoltat pentru a fi utilizat pe dispozitive mobile ca telefoanele și tabletele. IOS este sistemul de operare al companiei Apple, a fost dezvoltat pentru iPhone, iPad și Apple TV. Acesta provine din sistemul de operare Mac.

Android oferă următoarele beneficii:

- Este o platformă Open Source folosită de un număr mare de producători de dispozitive mobile
- Permite acces pentru aplicații de tip third pary
- Permite instalarea fișierelor cu extensia apk
- Ușor de personalizat
- Acces rapid la o mulțime de aplicații gratuite sau premium create pentru sistemul de operare Android
- Specificații hardware mai bune

[6] https://www.android.com

IOS oferă următoarele beneficii:

- Structura sistemului este foarte stabilă
- Securitate crescută
- Achiziționarea de aplicații se poate face doar din aplicația AppStore

4.3.4. Web Scraper

Web scraping [7] e procesul prin care se preiau anumite informații de pe o pagina de internet si sunt salvate într-un fișier (ex : csv). În cadrul aplicației prin intermediul web scraping vom prelua informații despre articole precum: nume, descriere, link către episod, număr de episoade.

Web Scraper e un tool care ne permite sa facem selecția parametrilor într-un mod simplu printr-un plug-in atașat browser-ului de internet. Interfața e sugestivă și ușor de utilizat. Prin intermediul acestui tool putem crea selectori specifici acelui site. Selectorii pot fi pentru text, link-uri, imagini, tabele dar si selectori de tip click sau scroll care ne ajuta sa parcurgem site-ul.

O altă variantă pentru Web Scraping ar fi folosind Srapy și MongDB care sunt librării pentru web scraping în Python. Aceste librării permit utilizatorului să facă același lucru ca Web Scraper doar că acesta va fi nevoit să scrie secvențe de cod pentru ce vrea să preia de pe fiecare site într-un director doar că datele vor fi salvate în MongoDB.

Aceste doua metode se diferențiază de multe alte metode prin faptul că sunt capabile să facă web scraping și pe site-uri care nu au interfețele doar în html și css.

4.3.5. RecyclerView

RecyclerView[8] este un widget utilizat pentru afișarea de tip listă pe cantități mari de date. Spre deosebire de LinearLayout care creează toate obiectele din listă un RecyclerView creează doar o parte din datele totale acestea fiind create pe măsură cae utilizatorul face scroll.

4.3.6. Glide

Glide[9] e un framework de gestionare media și încărcare de imagini pentru Android. Glide oferă o modalitate rapidă și eficientă de preluare, decodarea și afișarea imaginilor în special unde se face scroll pe o serie de imagini.

4.3.7. Python

Python[10] este un limbaj de programare creat de Guido van Rossum și făcut public în 1991. Este folosit pentru dezvoltarea web, software și matematică. Conține numeroase

[7] https://www.webscraper.io
[8] https://developer.android.com/reference/android/support/v7/widget/RecyclerView
[9] https://github.com/bumptech/glide

pachete precum Django, Flask și Tornado care ușurează munca utilizatorului. Rezultatele obținute în urma folosirii limbajului sunt rapide datorite varietății de librării existente.

4.3.8. Bootstrap

Bootstrap[11] este un framework de CSS care ajuta la designul site-urilor web. Este construit cu HTML și CSS. Conține o varietate de modele CSS predefinite făcând designul unui site web mult mai rapid și mai ușor pentru dezvoltatori, de asemenea este ușor custumizabil nevoilor fiecărui utilizator.

4.3.9. Libraria csv

CSV[12] este unul din cele mai folosite metode de import și export de date. Datele stocate în fișierele csv sunt separate de virgulă. Librăria csv pentru Python conține funcții de scriere și citire din fișierele csv. Funcția folosită în cadrul aplicației este ”csv.DictReader” care în momentul citirii informațiilor din fișier mapează informațiile de pe rânduri în funcție de numele coloanelor astfel pentru a prelua date de pe o anumita coloana trebuie doar sa specificăm numele coloanei.

4.3.10. Recommandation Engine

Pentru a oferi utilizatorului recomandări s-a optat pentru o soluție bazată pe un sistem propriu care are pașii:

- Parcurge lista de preferințe a utilizatorului și determina atât pentru filme cât și pentru seriale ce tip de gen predomină (care e cel mai întalnit)
- Pentru genurile care se încadrează în parametrii caută primul articol din topul aplicației care nu apare în lista utilizatorului și astfel obținem prima recomandare
- Pentru a doua recomandare caută în lista celor mai recent adăugate articole și îl recomanda pe primul care nu se află în lista de preferințe a utilizatorului

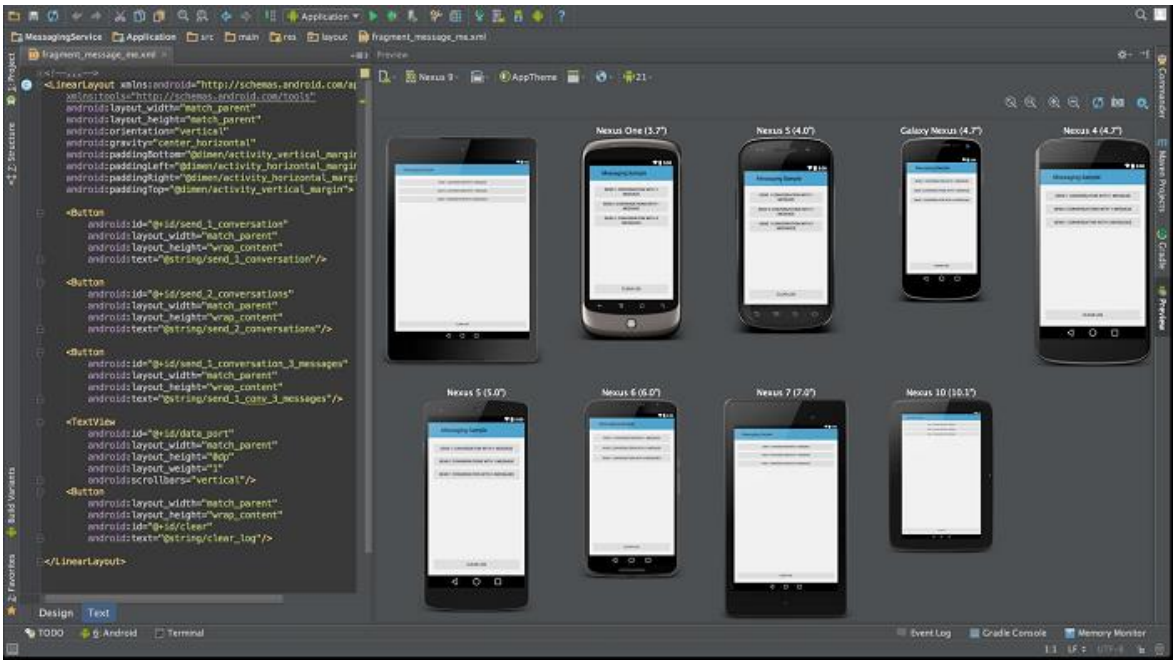
4.3.11. Android Studio

Android Studio[13] este un mediu de dezvoltare pentru aplicații Android bazat pe IntelliJ. Acesta oferă o serie de caracteristici care îmbunătățesc productivitatea la crearea aplicațiilor Android cum ar fi:

- Un emulator rapid și eficient
- Un sistem flexibil bazat pe Gradle pentru build
- Un mediu în care se pot dezvolta aplicații pentru toate tipurile de sisteme cu Android
- Modalitate de a adăuga modificările făcute aplicației fără a crea o noua aplicație de tip apk

[10] https://www.python.org
[11] https://getbootstrap.com
[12] https://docs.python.org/3/library/csv.html
[13] https://developer.android.com/studio/intro/?gclid=Cj0KCQjwjYHpBRC4ARIsAI-3GkG095pV8fvs0pI_OPM69ocCKrzsiwctAIZiKOWL0co5RmRAS3SqdZcaAudhEALw_wcB

- Template-uri care ușurează crearea de aplicații
- Unelte și framework-uri de testare
- Suport pentru platforma Google Cloud pentru a face integrarea aplicațiilor în cloud mai ușoară

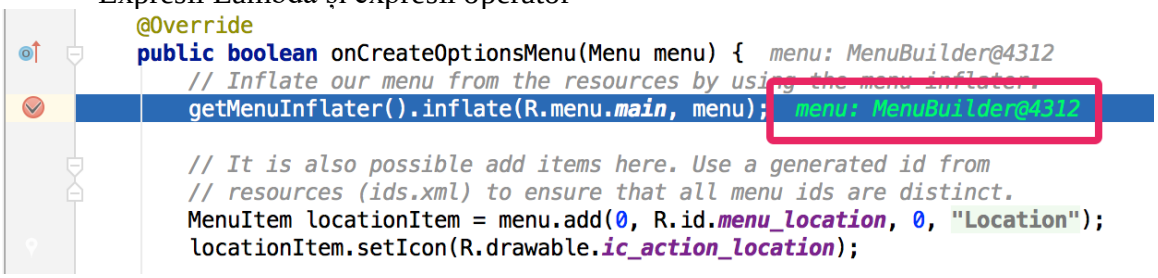


Figură 4.8- Android Studio

4.3.11.1. Debug

Android Studio oferă modalități performante de debug printre care se numără și ”inline debugging ” care face parcurgerea codului în etapa de debug mai ușor de urmărit oferind informații despre variabilele din fiecare linie de cod în linia respectiva. Aceste informații permit dezvoltatorului să vadă informații cum ar fi:

- Valoarea variabilei
- Obiecte referite
- Valoarea returnată de metode
- Expresii Lambda și expresii operator



Figură 4.9- Exemplu de inline debugging

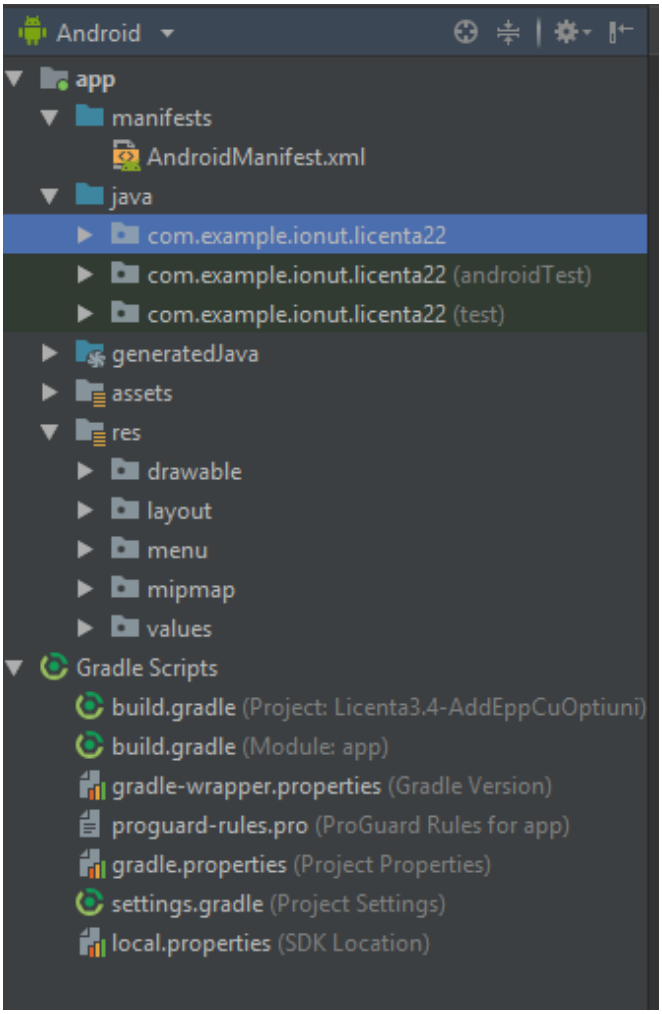
4.3.11.2. Structura proiectelor Android Studio

Proiectele Android conțin unul sau mai multe module cu fișiere de tip cod sursă sau resurse.

Fiecare modul al unei aplicații conține următoarele directoare care se pot observa în figura 4.8:

- **manifest**: conține fișierul AndroidManifest.xml
- **java**: conține fișiere cu surse de cod și codul pentru testele Junit
- **rez**: conține resursele care nu conțin cod cum ar fi:
  - layout-uri xml
  - imagini
  - liste pentru interfața utilizator

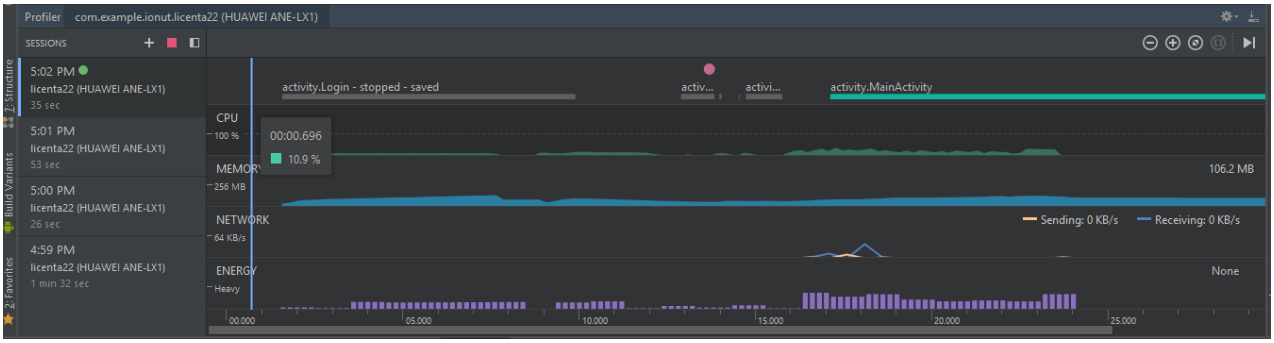
Pentru adăugarea surselor locale pe dispozitiv este necesară crearea unui director cu denumirea ”assets” care să conțină resursele dorite și care sa fie creat în directorul ”app” al aplicației



Figură 4.10 – Structura Proiectului Android

4.3.11.3. Android Profiler

Android profiler este un tool care ne permite să urmărim consumul resurselor în timpul rulării aplicației și ne oferă informații de acțiunile care au avut lor pe timpul rulării cum sunt clasa curent utilizată metoda apelată de utilzator și ne permite să observăm ce impact au aceste acțiuni asupra dispozitivului.



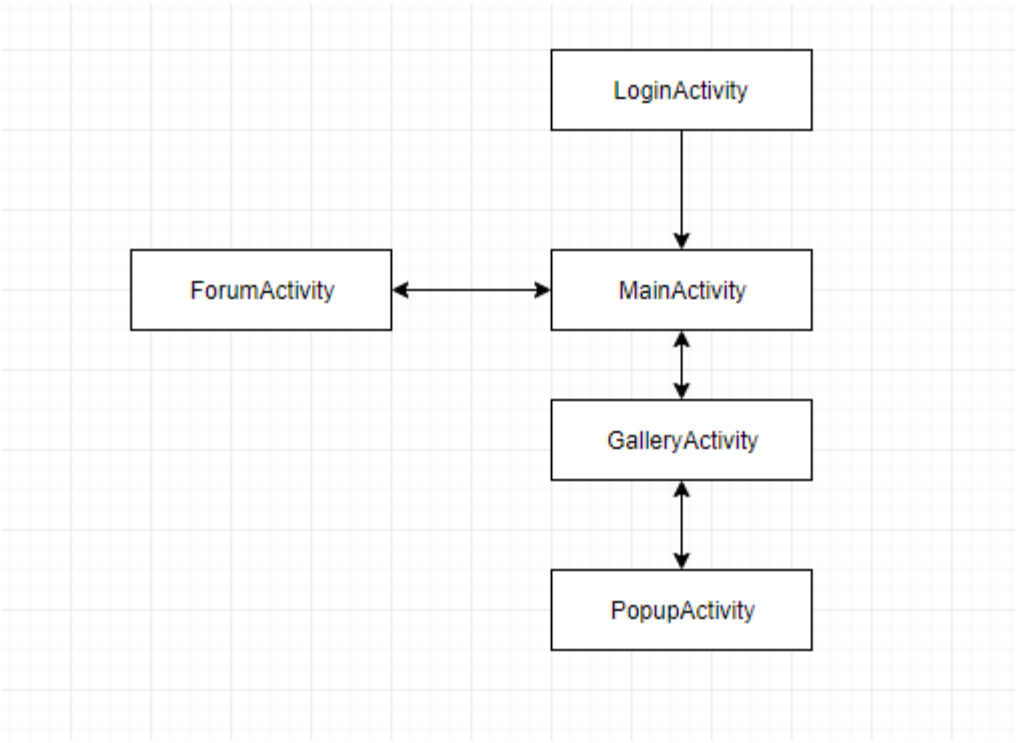
Figură 4.11- Exemplu Utilizare Android Profiler

Capitolul 5. Proiectare de Detaliu si Implementare

Acest capitol cuprinde o descriere detaliată a arhitecturii conceptuale a sistemului propus. Diagrama bazei de date, diagrama de pachete și clase și diagrama de deployment vor fi prezentate tot în acest capitol. Fiecare modul și componentele sale vor fi descrise în detaliu.

5.1. Arhitectura sistemului

În acest subcapitol sunt descrise componentele principale ale sistemului și modul în care acestea interacționează.

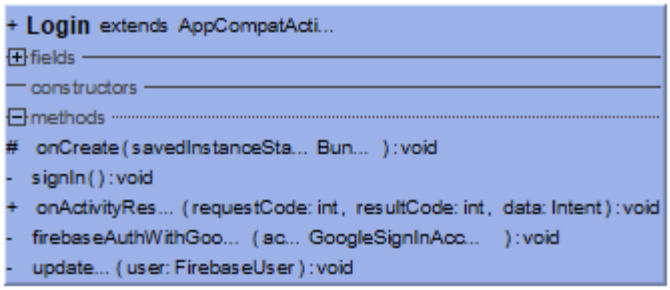


Figură 5.1- Arhitectura sistemului

Componetele principale ale sistemului sunt: autentificare (LoginAcitvity), forum (ForumActivity) și articole (MainActivity, GaleryActivity, PopupActivity), iar funcționalitățile vor fi descrise în continuare.

5.1.1. Autentificare

Se ocupă de partea de autentificare prin Google cu ajutorul Firebase Authentification, este necesar ca utilizatorul să fie logat pentru a putea utiliza funcționalitățile aplicației.



Figură 5.2- Clasa Login

5.1.2. Articole

Se ocupă de prelucrarea datelor despre articole și despre utilizator.

MainActivity:

- ❖ preia datele despre user obținute în procesul de logare din contul Google al acestuia pentru a crea meniul de utilizator personalizat pentru fiecare dintre utilizatori cu numele de utilizator si imaginea de profil a contului Google.
- ❖ Populează pagina inițială a aplicației unde avem :
  - o Bara de căutare: permite utilizatorului să caute un articol specific in baza de date
  - o Două recomandări pentru utilizatorul curent bazat pe metodele explicate în 4.3.7
  - o O lista cu ultimele zece articole adăugate în baza de date
- ❖ Articolele afișate sunt elemente într-un RecyclerView, fiecare element este format din imaginea și titlul articolului
- ❖ Prin selectarea unui articol apelăm GaleryActivity

GalleryActivity:

- ❖ Reprezintă pagina pentru un articol și conține:
  - o Titlu
  - o Descriere
  - o Imagine
  - o Genul(genurile)
  - o Rating-ul
  - o Link-uri de acces către video
  - o Buton de adăugare în lista (la articolele care nu se află deja în listă) care apelează PopupActivity
  - o Pentru seriale și anime-uri care sunt în proces de derulare conține și data apariției următorului episod

PopupActivity:

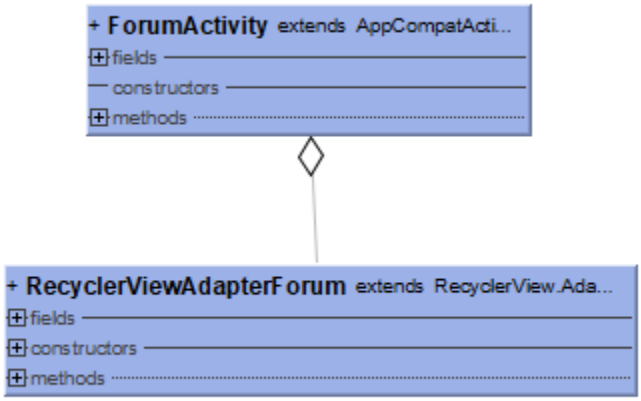
- ❖ Deschide o fereastră la apăsarea butonului de adăugare al unui articol în lista și oferă utilizatorului posibilitatea de a încadra acest articol în una din

categoriile următoare (bazate pe stadiul de vizionare al articolului în momentul adăugării în listă) :

- Watched : pentru articole deja vizionate
- Plan to watch: pentru articole încă nevizionate
- Watching : pentru seriale si anime-uri în curs de vizionare. Aici se poate introduce și nuăarul de episoade deja vizionate

5.1.3. Forum

Această componentă se ocupă de clasele care se ocupă de crearea și popularea forumului. Clasa ForumAcivity reprezintă pagina forumului din aplicație care conține mesajele utilizatorilor. La postarea mesajelor apare numele utilizatorului care a postat și mesajul acestuia. Clasa RecyclerViewAdapterForum conține informații legate de utilizator și mesajul corespunzător acelui utilizator.



Figură 5.3- Forum

5.2. Interfața utilizator

Interfața utilizator este alcatuită dintr-un set de fișiere de tip xml asociate activităților. Principalele fișiere ale aplicației sunt descrise mai jos.

5.2.1. Interfața pentru MainActivity

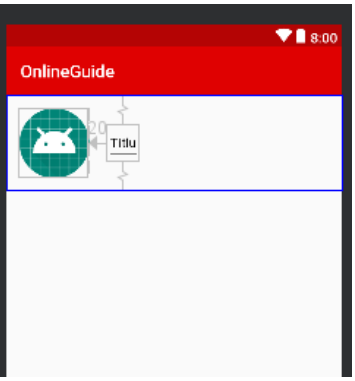
Conține bara de căutare și două RecyclerView, unul pentru recomandări și unul pentru articolele afișate, la crearea elementelor grafice odată cu specificații legate de poziția acestora pe ecran le atribuim și id-uri pentru a le putea identifica și modifica in activități. Un exemplu de RecyclerView este prezentat în figura 5.4. La inițializarea aplicației elementele din RecyclerView sunt completate pe baza unui model creat sa afișeze imaginea si titlul articolelor din listă prezentat în figura 5.5.

```
<TextView
    android:layout_width="match_parent"
    android:layout_height="40dp"
    android:text="Last added:"
    android:layout_marginTop="290dp"
    android:textColor="#000"/>
<android.support.v7.widget.RecyclerView
    android:id="@+id/recycler_view"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_alignParentStart="true"
    android:layout_alignParentTop="true"
    android:layout_marginStart="0dp"
    android:layout_marginTop="330dp"
    android:layout_alignParentLeft="true"
    android:layout_marginLeft="0dp">

</android.support.v7.widget.RecyclerView>
```

Figură 5.4- RecyclerViewer

```
<de.hdodenhof.circleimageview.CircleImageView
    android:layout_width="80dp"
    android:layout_height="80dp"
    android:id="@+id/image"
    android:src="@mipmap/ic_launcher"/>
<android.support.v7.widget.AppCompatAutoCompleteTextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Titlu"
    android:id="@+id/title"
    android:layout_toRightOf="@+id/image"
    android:textColor="#000"
    android:layout_centerVertical="true"
    android:layout_marginLeft="20dp"
    android:textSize="15dp"/>
```



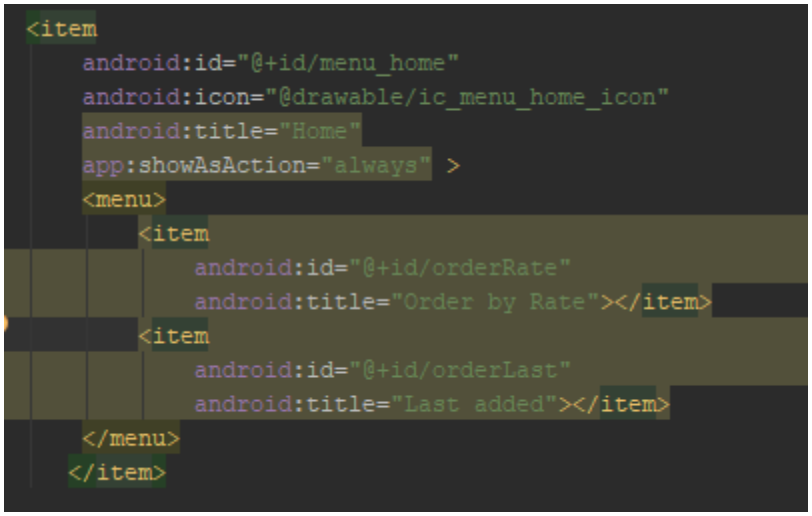
Figură 5.5 – Model element RecyclerView

5.2.2. Interfața pentru GalleryAcivity

Reprezintă pagina unui articol specific și conține imaginea, descriere, titlul, bara de rating, lista de link-uri sau episoade cu link-uri și butonul de adăugare a articolului în lista de preferințe.

5.2.3. Bara de navigare

Conține elemente de tip meniu și este integrată în toate interfețele. În Android Stdio meniurile sunt create în pachetul ”res” în folderul ”menu” la începutul acestor fișiere xml nu se declară tipul de layout si se începe cu ”<menu >”. Pentru a crea un element din meniu acesta e declarat ca ”<item>”. Elementele din meniu pot conține submeniuri ca cel prezentate în figura 5.6.



Figură 5.6- Submeniu cu două elemente

5.3. Structura bazei de date

Acest subcapitol descrie cum datele din baza de date sunt socate și cum e structurată baza de date a proiectului. Firebase e o bază de date RealTime (baza de date noSQL cloud-hosted)

5.3.1. Stocarea datelor

Toate datele din Firebase RealTime Database sunt stocate ca obiecte de tip JSON. Baza de date rezultată e asemănătoare unui arbore JSON cloud-hosted. Spre deosebire de bazele de date SQL nu exista tabele, când adăugam date în arborele JSON acestea devin un nod în structura JSON existentă cu o cheie asociată. Poți introduce chei personalizate sau acestea pot fi generate automat folosind push().[2]

Cheile introduse trebuie să respecte următoarele constrângeri:

- Trebuie să fie codificate folosind UTF-8
- Trebuie să nu depășească 768 biți
- Nu pot conține ”.” ,” $”, ”#”, ”[”,”]”,” /”
- Nu pot conține caractere de control ASCII 0-31 sau 127

În documentul oficial [11] se menționeaza faptul că deoarece Firebase RealTime Database permite date îmbricate până la 32 nivele există impresia că așa ar trebui să fie structura standard. Totuși când preluăm date de la o locație din baza de date primim și nodurile copil. Pe lângă asta de câte ori permitem acces de scriere sau citire la un nod din baza de date permitem accesul și la tot ce se afla în acel nod. Din acest motiv e indicat să structurăm datele cât mai simplu posibil.

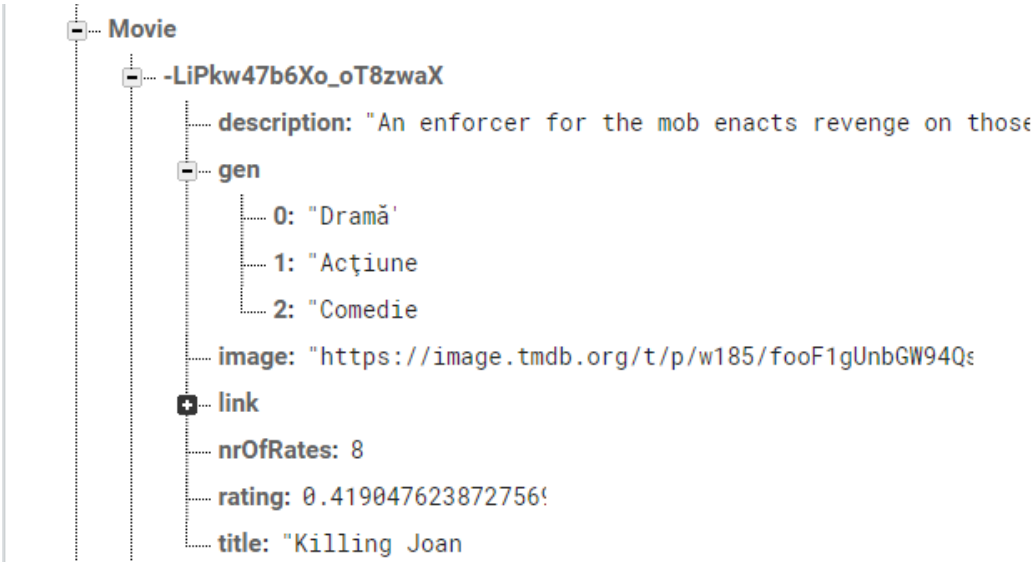
5.3.2. Structura bazei de date a proiectului

Considerând modul de stocare a datelor am creat structura bazei de date astfel încât să se potrivească cât mai bine necesităților aplicației și am obținut patru noduri principale: Movie, Series,Anime,User.

- Movie : conține informații despre filmele introduse în baza de date, fiecare nod fiind un film

- Series: conține informații despre serialele introduse în baza de data, fiecare nod fiind un serial
- Anime: conține informații despre anime-urile introduse în baza de date, fiecare nod fiind un anime
- User: conține informații despre utilizatorii aplicației, fiecare nod fiind un user

Pentru o mai buna înțelegere a bazei de date și cum e structurată voi prezenta un exemplu mai detaliat legat de cum arată un film și cum e adăugat acesta în lista utilizatorului.



Figură 5.7- Strucura film în Firebase

Figură 5.8- Structura utilizator în Firebase

Observăm în Figura 5.7 toate detaliile pe care le stocam despre un film în baza de date la secțiunea Movie (titlu, descriere, imagine, gen, link-uri, rating-ul și de câte ori a fost făcut rate), iar în secțiunea User pentru fiecare articol adăugat reținem cheia și date despre progresul utilizatorului: progresul și unde e cazul si numărul de episoade vizionate. Astfel când afișăm lista cu preferințe a utilizatorului putem căuta articolele pe baza cheii și de asemenea așa stabilim pentru ce articole să afișăm butonul de adăugare în lista de preferințe.

5.3.3. Preluarea datelor

Toate funcțiile care se ocupa de preluare de date sunt create ca inner class și extind AyncTask[14]. AsyncTask este o clasă creată pentru a ușura gestionarea treadurilor din background. Aceasta clasă se ocupă de gestionarea treadurilor și oferă posibilitatea utilizatorului sa modifice ce se întâmplă în treadul respectiv folosind metode predefinite pentru a modifica datele înainte de rulare, în timpul sau după rulare. Apelarea acestor clase se face folosind ”new NumeClasa().execute(parametrii)”, numărul și tipul parametrilor variază de la o clasă la alta și este declarat când adăugam extensia clasei astfel: ” class NumeClasa extends AsyncTask<Parametru1, Parametru2, ... >” parametrii specificați reprezintă doar tipul de date primite ca parametru spre exemplu „AsyncTask<Integer, String, URL>”.

[14] https://developer.android.com/reference/android/os/AsyncTask#usage

Pentru a accesa datele stocate în Firebase RealTime Database în cadrul aplicației Android trebuie să creăm o referință către ramura arborelui din baza de date pe care se afla datele dorite apoi putem sa parcurgem acea ramura cât de adândc dorim prin folosirea funcției ”child()” care returnează o listă cu toate subnodurile nodului curent.

De exemplu daca am dori sa extragem un film cu genul de acțiune ca cel din figura 5.6 prima data trebuie să creăm o referință la Movies apoi pe baza acelei referințe ordonăm datele rezultate după nodul ”gen”, datele returnate de Firebase sunt de tip DataSnapsot aceste date se pot mula pe clasele aplicației folosind funcția getValue() și specificând ca parametru pentru aceasta funcție clasa pe care trebuie să se muleze datele în cazul nostru ”Movie.class” apoi pentru a determina dacă obiectul creat se potrivește parametrilor stabiliți (are ca gen acțiune) folosim funcția contains pe setul de genuri ale filmului obținut. În figura 5.9 putem observa un exemplu unde ”strings[0]” reprezintă genul pentru care se face căutarea.

```
DatabaseReference movieReference = database.getReference( s: "Movie");

movieReference.orderByChild("gen").addValueEventListener(new ValueEventListener() {

    @Override
    public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
        for (DataSnapshot snapshot : dataSnapshot.getChildren()) {

            Movie m= snapshot.getValue(Movie.class);
            m.setKey(snapshot.getKey());
            if(m.getGen().contains(strings[0])){
                addMovie(m);
```

Figură 5.9- Extragerea datelor din Firebase care conțin un gen specificat

5.4. Server și Scraper

Partea de server a aplicației se ocupă de popularea și actualizarea bazei de date folosind informațiile obținute în urma procesului de scraping.

5.4.1. Web Scraper

Web Scraper[15] este o extensie de browser ușor de utilizat care permite extragerea datelor de pe un site printr-o interfață point and click care creează un scraper pentru site-ul selectat și oferă posibilitatea de a descărca datele în format CSV și a programa rulări automate în cloud-ul celor de la Web Scraper la intervale de timp stabilite.

5.4.2. Server

Utilizator administrator este singurul care are acces la acest server. Serverul aplicației este local și utilizează datele preluate prin procesul de scraping pentru a introduce date în Firebase .

[15] https://www.webscraper.io

Acesta a fost implementat în Python folosind Flask[16], Pyrebase[17] și librăria CSV.

Flask este folosit pentru a crea o interfață web ușor de utilizat pentru administrator. Pyrebase e utilizat pentru a face conexiunea între server și baza de date Firebase. Pentru a configura Pyrebase odată pachetul importat trebuie să adăugăm configurarea corespunzătoare bazei de date oferită de Firebase ca în figura 5.10.

```
config={
    "apiKey": "AIzaSyC_O07zr72OAS5F87OKJOS2-Rbhn0iesFg",
    "authDomain": "licenta-7db62.firebaseapp.com",
    "databaseURL": "https://licenta-7db62.firebaseio.com",
    "projectId": "licenta-7db62",
    "storageBucket": "licenta-7db62.appspot.com",
    "messagingSenderId": "1081778156698",
    "appId": "1:1081778156698:web:334ffe6406f0f266"
}

firebase = pyrebase.initialize_app(config)

db=firebase.database()
```

Figură 5.10 – Configurarea Pyrebase

Pentru a putea prelucra datele din fișierul csv indiferent de ordinea coloanelor folosim Dictionary care ne permite sa preluăm coloanele din fișierul csv după numele lor astfel pentru toate site-urile de pe care se preiau date se respectă următorul model:

- Coloana care conține titlurile articolelor se va numi ”title”
- Coloana care conține descrierea articolelor se va numi ”description”
- Coloana care conține imaginea articolelor se va numi ”image-src”
- Coloana care conține genurile articolelor se va numi ”gen”
- Coloana care conține link-urile pentru filme se va numi ”link-href”
- Coloana care conține episoadele pentru seriale și anime-uri se va numi ”episodes” iar cea cu link-urile catre episoadele respective ”episode-href”

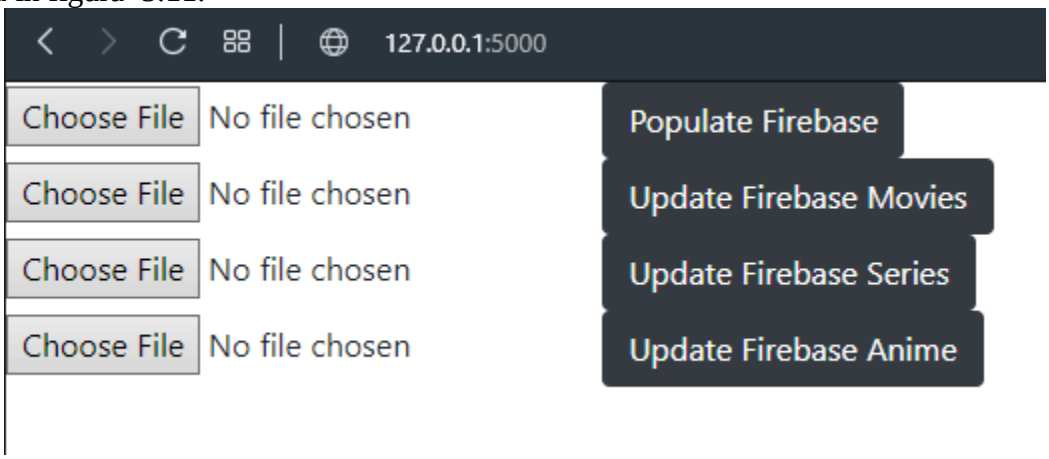
Datele preluate din fișierul csv se introduc în Firebase ca populare sau actualizare de către administrator.

Datele introduse ca populare sunt cele extrase când se adaugă un nou site la aplicație. Primul scrape pe un site preia informații despre toate articolele existente pe acel site apoi datele din fișierul csv rezultat sunt introduse în baza de date. Următoarele scrape-uri pe același site vor prelua doar ultimele articole introduse de site-ul respectiv.

Pentru că aplicația oferă multiple surse video la adăugarea datelor, prima data se verifică dacă articolul pe care încercăm să îl adăugăm în Firebase deja există caz în care actualizăm lista cu link-uri de acces către sursa video. Dacă articolul nu există vom crea un nou articol cu specificațiile preluate din fișierul csv la care adăugăm câmpurile de rating și numărul persoanelor care au făcut rate inițializate cu 0, iar la seriale și anime-uri pe lângă aceste câmpuri adăugăm și numărul de episoade care se incrementează automat la introducerea unui nou episod.

[16] http://flask.pocoo.org
[17] https://github.com/thisbejim/Pyrebase

Interfața aplicației server a fost creata folosind Flask si Bootstrap și se poate observa în figura 5.11.



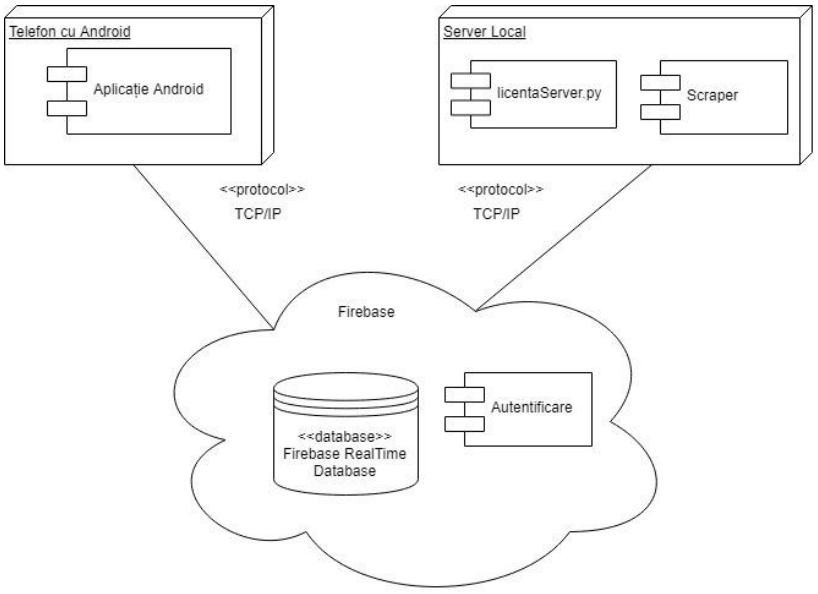
Figură 5.11 – Interfață server local

5.5. Diagrama de deployment

Diagramele de deployment sunt utilizate pentru a descrie componentele hardware care conțin componentele software și a oferi o vizualizare asupra topologiei sistemului.

Sistemul conține trei componente hardware principale:

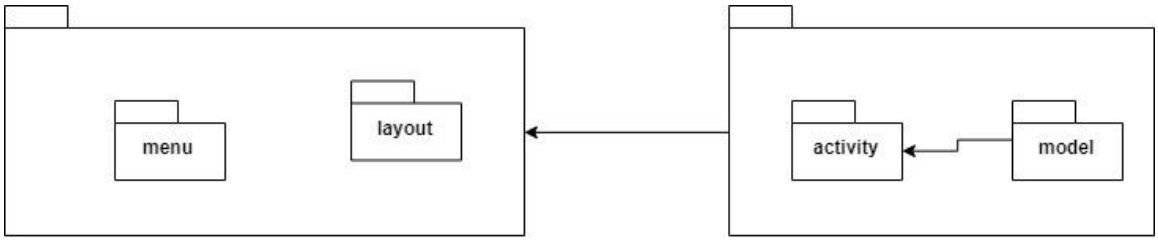
- Serverul local unde este gestionat procesul de scraping, iar datele rezultate în urma acestui proces, reprezentate de fișiere csv, sunt interpretate și încărcate în Firebase
- Telefon cu Android care permite utilizatorului accesul la serviciile oferite de aplicație
- Firebase folosit pentru autentificare și baza de date în timp real



Figură 5.12- Diagrama de deployment

5.6. Diagrama de pachete

Diagrama de pachete ilustrează modul în care sunt grupate clasele pentru o mai bună organizare și structurare a proiectului. Aplicația conține două mari pachete cel de resurse (care conține menu și layout) unde se găsesesc elementele de grafică și cel de activități si clase Java (activity și model) reprezentate in Figura 5.12.



Figură 5.12- Diagrama pachete

Capitolul 6. Testare şi Validare

Acest capitol arată metodele de testare utilizate pentru a asigura funcționarea corectă a funcționalităților aplicației.

În perioada de dezvoltare a sistemului testarea s-a făcut pentru fiecare nouă componentă adăugată în proiect astfel descoperind bug-uri și compatibilitatea noilor componente cu sistemul testând ca acestea să nu afecteze negativ componentele deja existente.

Procesul de validare a sistemului a avut loc de la începutul până la finalul procesului de implementare după fiecare etapă a implementării.

6.1. Testare Manuală

Majoritatea testării pe perioada de dezvoltare a fost făcută manual datorită faptului că aplicația este una mobile, domeniu în care cel mai întâlnit tip de testare e cea manuală testând pas cu pas cerințele funcționale și fiind dependentă de datele primite din Firebase am creat scenarii de test ca cele prezentate mai jos bazate pe cerințele sistemului.

Caz de testare 1
Adăugare articol în lista de preferințe
Actor: Utilizator
Precondiții:

- Utilizatorul este autentificat
- Utilizatorul a selectat un articol

Acțiune	Rezultat așteptat
Apasă butonul de adăugare	Apare fereastra de popup cu progresul
Selectează meniul dropdown	Apar cele trei variante de progres
Selectează progresul	Meniul se închide și rămâne valoarea selectată de utilizator
Apasă ”ADD”	Fereastra de popup se închide și baza de date este actualizată
Apasă ”CANCEL”	Fereastra de popup se închide și nimic nu se modifică în baza de date

Tabel 6.1

Caz de testare 2
Accesare link articol
Actor: Utilizator
Precondiții:

- Utilizatorul e autentificat
- Utilizatorul a selectat un articol

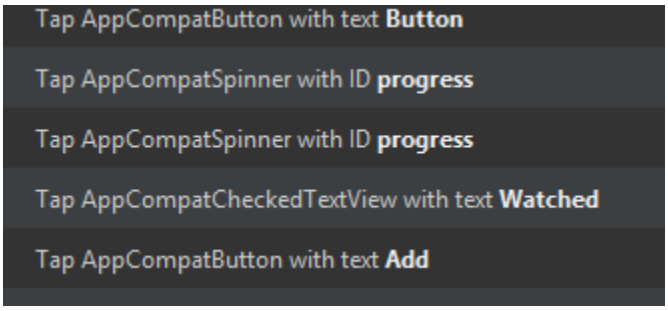
Acțiune	Rezultat așteptat
Deschide meniul de link-uri	Lista de link-uri este afișată
Selectează link-ul dorit	Este redirecționat către site-ul corespunzător link-ului selectat

Tabel 6.2

6.2. Testare automată

Pentru procesul de testare automată am folosit Espresso Test Recorder[18] care permite crearea testelor UI fără a fi nevoit să scrii cod. Espresso Test Recorder înregistrează un scenariu de test făcut interactiv de către dezvoltatorul aplicației care permite adăugarea de posibile inserturi pe care un utilizator le face în interfața grafică apoi pe baza acțiunilor înregistrate generează automat un test pe care îl putem rula pentru a testa aplicația pe viitor.

În figura 6.1 putem observa pașii făcuți pentru a adăuga un articol la lista de preferințe înregistrați cu Espresso Test Recorder.



Figură 6.1- Espresso Test Recorder adăugare articol în lista de preferințe

[18] https://developer.android.com/studio/test/espresso-test-recorder

Capitolul 7. Manual de Instalare si Utilizare

Acest capitol va descrie resursele minime necesare pentru ca aplicația să funcționeze corect și un set de instrucțiuni pas cu pas pentru instalarea si utilizarea aplicației.

7.1. Manual de instalare

Componente hardware și software necesare pentru a instala cu succes aplicația și a o putea utiliza:

- Un smartphone cu sistem de operare Android cu versiunea minima 8.0.1 sau mai nou
- Minim 1GB de ram
- Frecvență minimă 1.5 GHz
- Google Play disponibil pe telefon

Aplicația trebuie încarcată pe Google Play pentru a fi disponibila pentru utilizatori.

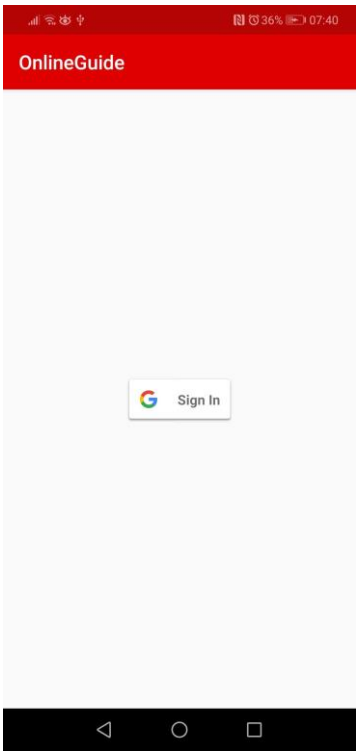
7.2. Manual de utilizare

Acest subcapitol prezinta pas cu pas cum se pot efectua acțiuni pe aplicație

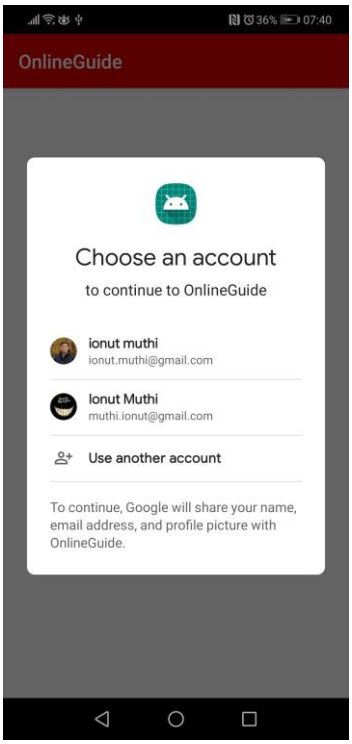
7.2.1. Autentificare

Pentru a putea folosi aplicația utilizatorul trebuie să fie autentificat. Autentificarea se face pe baza contului de Google.

Prin apăsarea butonului ”Sign in” utilizatorul va avea opțiunea de a alege între conturile sale Google sau introduce un nou cont, Figura 7.2



Figură 7.1- Pagina de logare



Figură 7.2- Opțuni autentificare Google

7.2.2. Căutarea de articole

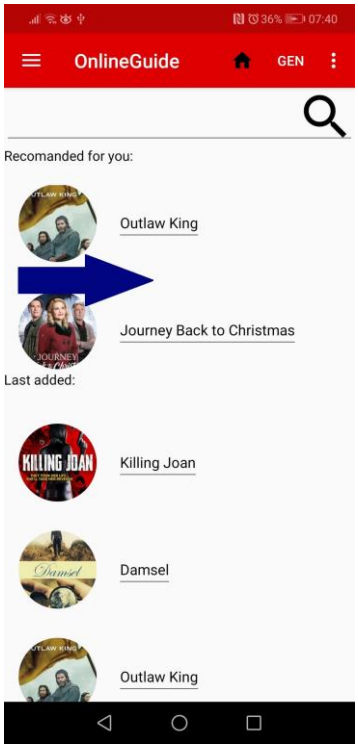
Folosind bara de căutare utilizatorul introduce titlul sau o parte din titlul unui articol iar aplicația va oferi o listă cu articolele care se potrivesc.

7.2.3. Meniu

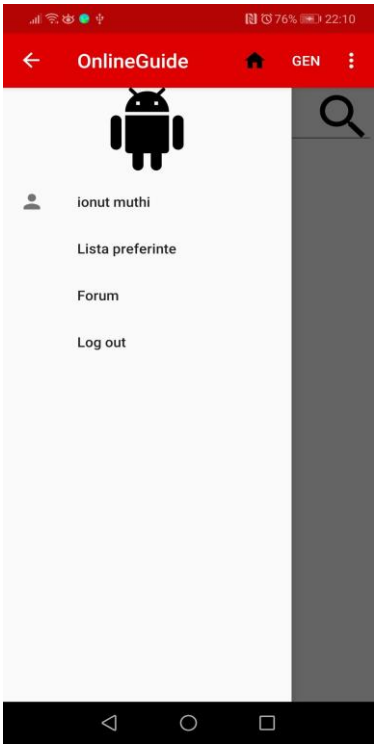
Aplicația are două meniuri pentru a ajuta utilizatorul sa navigheze ușor în aplicație și să își gestioneze contul.

7.2.3.1. Meniu utilizator

Permite utilizatorului să acceseze lista de preferințe și să se delogheze. Conține informații preluate din contul de Google al utilizatorului (numele acestuia de utilizator si imaginea de profil), Figura 7.4. Accesul la meniul utilizatorului se face prin glisare spre dreapta, Figura 7.3.



Figură 7.3- Accesarea meniu utilizator

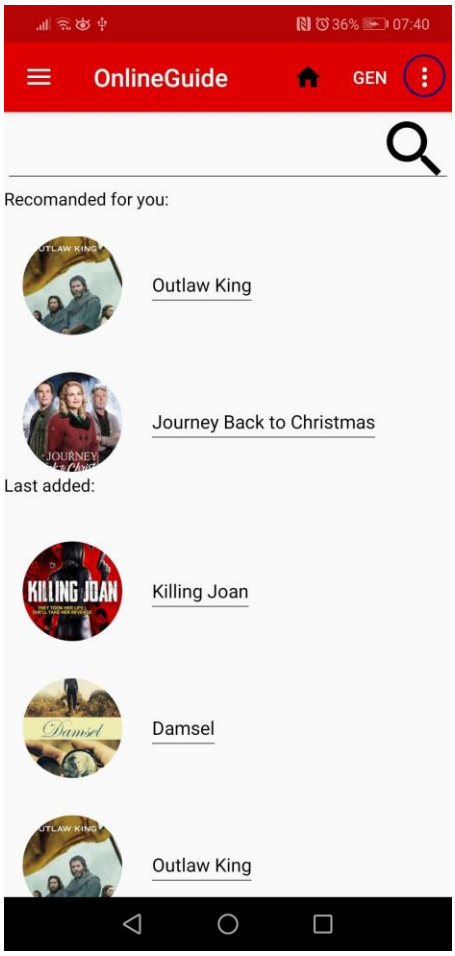


Figură 7.4-Meniu utilizator

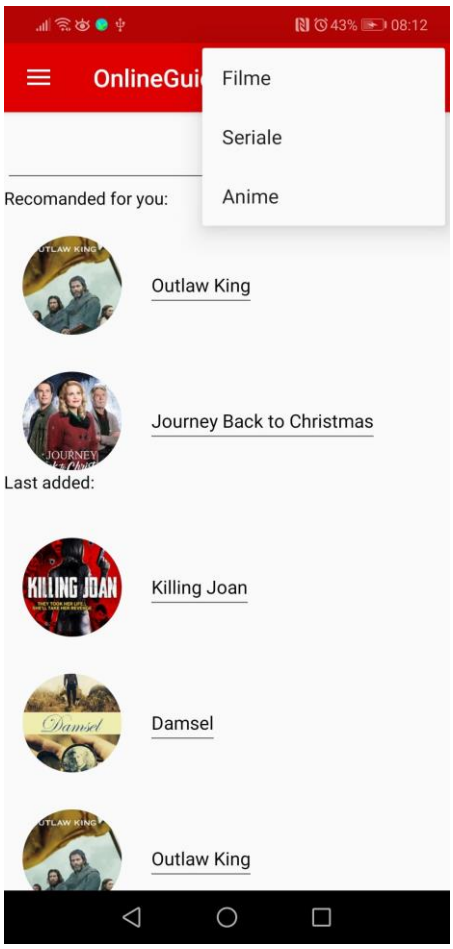
7.2.3.2. Meniu aplicație

Permite utilizatorului să navigheze ușor în aplicație și sa filtreze datele afișate în funcție de tipul de articol, în funcție de gen, în funcție de rating sau ultimele articole afișate.

- Sortarea după tipul de articol împarte articolele în trei mari categorii Figura 7.6. Accesul la meniu se face folosind cele trei puncte din colțul dreapta sus, Figura 7.5.

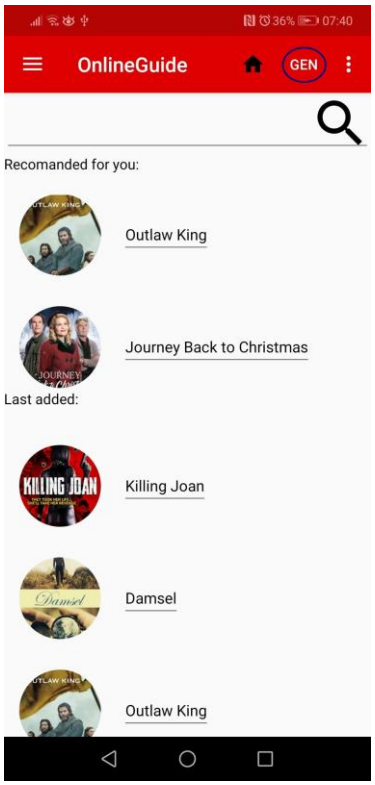


Figură 7.5- Buton acces meniu tip articol

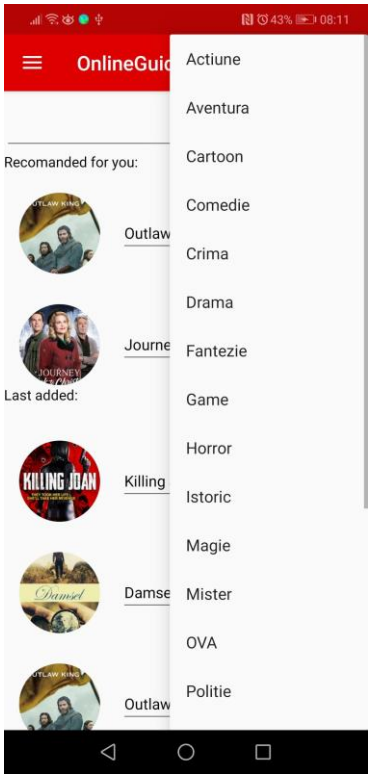


Figură 7.6- Meniu tip articol

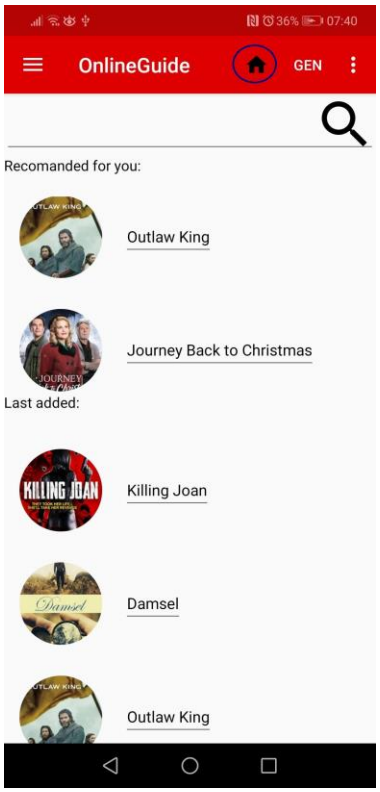
- Sortarea în funcție de gen îi oferă utilizatorului posibilitatea de a vedea toate articolele dintr-un gen specific, Figura 7.8. Accesul la meniu se face prin texul GEN din bara de navigare, Figura 7.7.
- Sortarea după Rating afișează articolele în funcție de rating-ul obținut in urma rate-urilor oferite de utilizatori, Figura 7.10
- Sortarea după ultimul adăugat afișează articolele în ordinea în care au fost adăugate în baza de date, Figura 7.10
- Accesul la ultimele două se face folosind simbolul căsuței, Figura 7.9



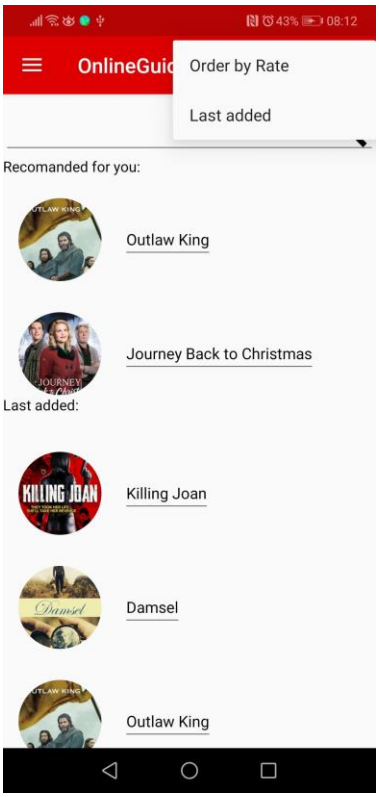
Figură 7.7 -Buton acces meniu gen



Figură 7.8- Meniu gen



Figură 7.9- Buton acces



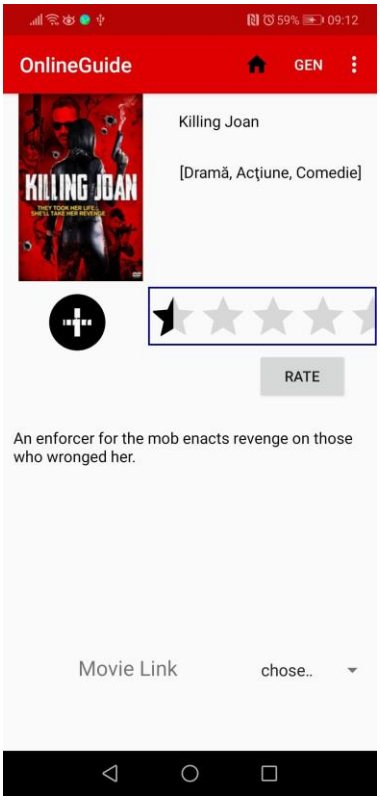
Figură 7.10- Meniu Rate/Ultimul adăugat

7.2.4. Articole

Odată selectat un articol utilizatorul e redirecționat catre pagina lui unde primește informații adiționale despre acesta și unde poate efectua o serie de acțiuni: rating, adăugare articol in lista, accesare video prin intermediul unui link.

7.2.4.1. Rating

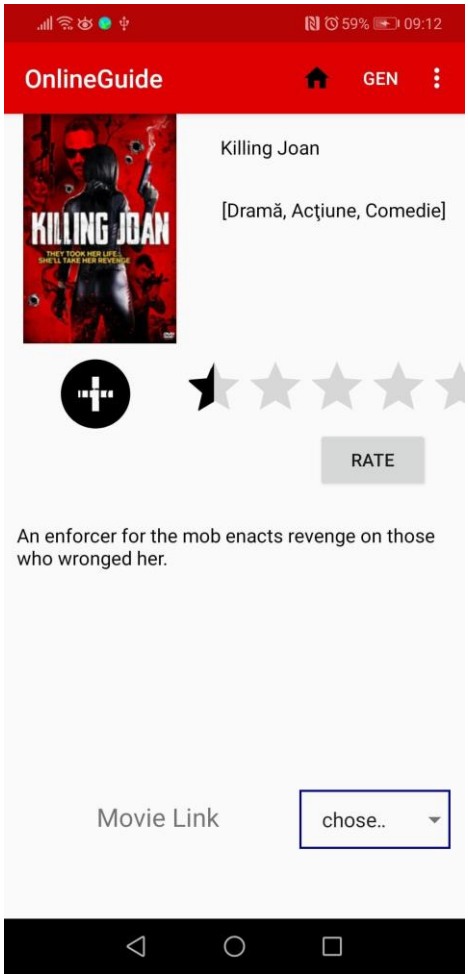
Utilizatorul are opțiunea de a acorda intre una și cinci stele articolului. Pentru a face rate trebuie prima data să selecteze numărul de stele dorite prin glisarea peste stele (Figura 7.11) apoi prin apăsarea butonului ”RATE” valoarea introdusă de acesta e adăugată și valoarea totala a ratingului se modifică.



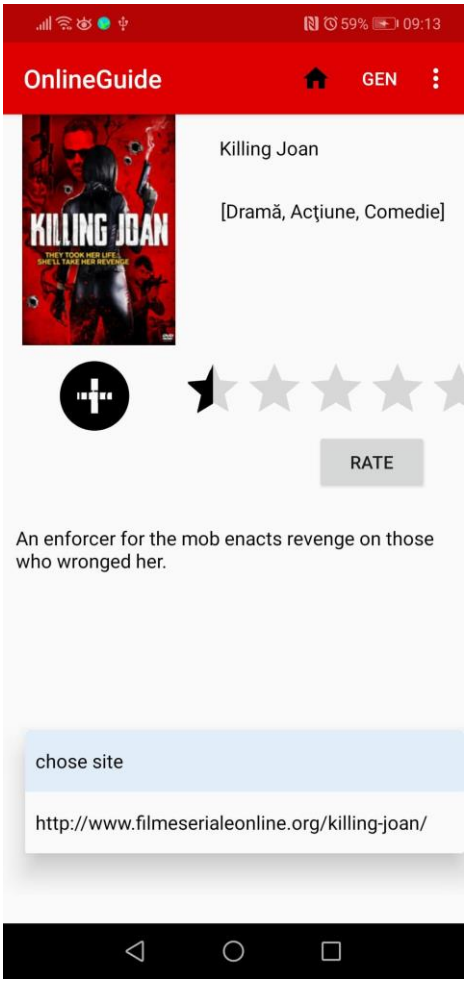
Figură 7.11-Bară rating

7.2.4.2. Accesare link către sursa video

Aplicația nu oferă video pentru articole, dar oferă link-uri către surse externe unde se poate viziona articolul dorit. Accesul la aceste link-uri se face prin apăsarea pe ”chose” (Figura 7.12) in colțul dreapta jos a paginii care va afișa o lista de link-uri (Figura 7.13) unde utilizatorul are acces la video. Prin selectarea unuia din aceste link-uri utilizatorul e redirecționat către site-ul respectiv.



Figură 7.12- Acces catre lista link-uri

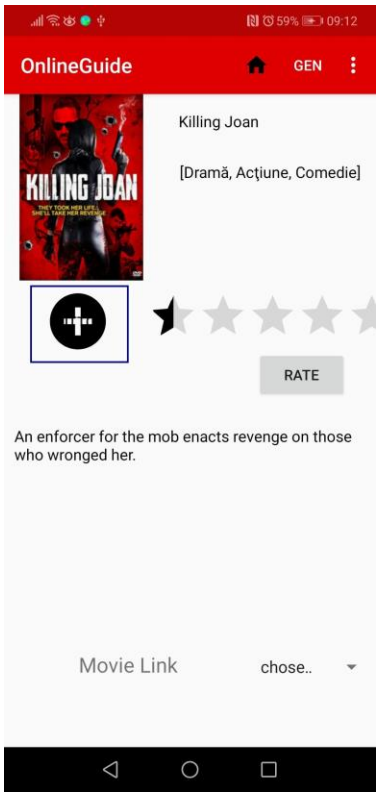


Figură 7.13- Lista link-uri

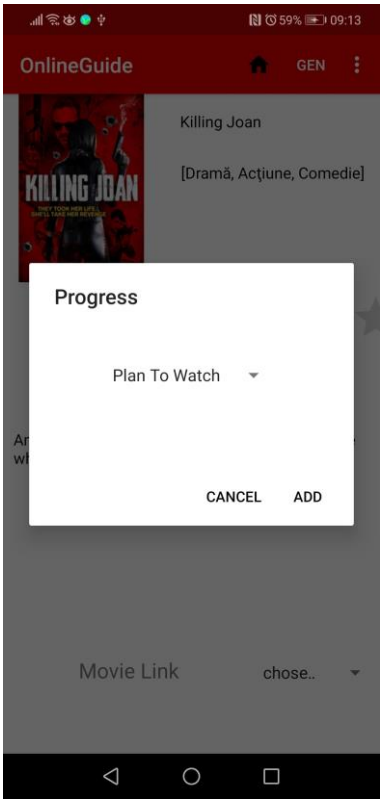
7.2.4.3. Adăugare articol în lista de favorite

Fiecare utilizator are o listă de preferințe care conține articole din cadrul aplicației pe care le poate accesa din meniul de utilizator (7.2.3.1.). Pentru a adăuga un nou articol la această lista utilizatorul trebuie sa deschidă articolul dorit și prin apăsarea butonului în formă de plus (Figura 7.14). Odată butonul apăsat utilizatorul poate selecta progresul de vizualizare (Figura 7.15).

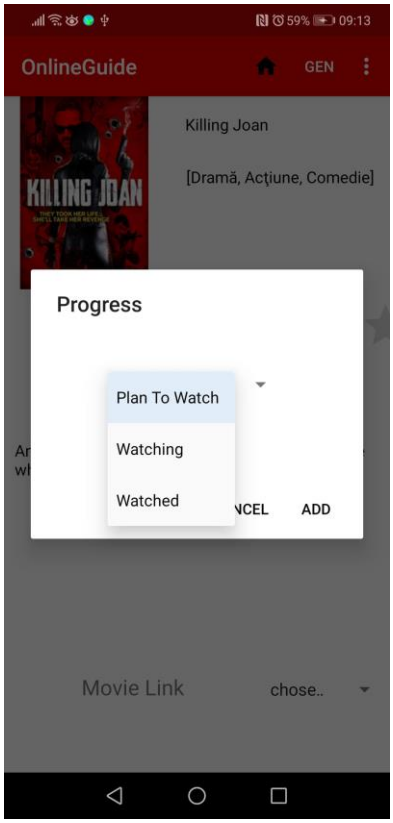
Dacă articolul selectat e un film opțiunile prezentate utilizatorului sunt: Plan to Watch și Watched. În cazul serialeor și anime-urilor pe lângă opțiunile Plan to Watch și Watched utilizatorul mai are varianta Watching (Figura 7.17) care îi permite să selecteze și numărul de episoade vizionate (Figura 7.16).



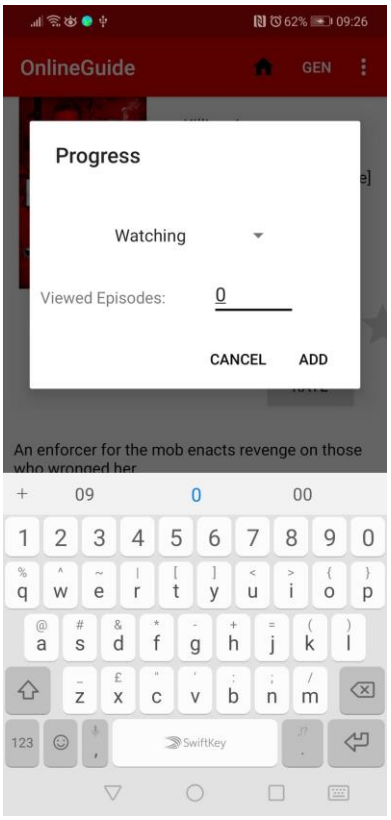
Figură 7.14- Buton adăugare în lista de preferințe



Figură 7.15- Selectarea progresului



Figură 7.16- Opțiuni progres



Figură 7.17- Watchin

7.2.5. Forum

Utilizatorii au acces la forum unde pot comunica cu alți utilizatori. Accesul la forum se face din meniul de utilizator (7.2.3.1.) prin selectarea opțiunii de forum. Pentru a adăuga un mesaj în forum utilizatorul o să introducă mesajul apoi apasă butonul ”POST”.

Figură 7.18-Forum

Capitolul 8. Concluzii

În acest capitol se vor prezenta rezultatele obținute comparativ cu setul de obiective stabilit la începutul proiectului și posibile îmbunătățiri aduse următoarelor versiuni ale aplicației.

8.1. Rezultate

Sistemul îndeplinește majoritatea obiectivelor stabilite în faza de proiectare care reprezintă crearea unei aplicații unde se regăsește conținut de filme, seriale și anime-uri de la o gamă de site-uri online pe care utilizatorul le poate accesa prin intermediul link-urilor și permite utilizatorului să gestioneze cu ușurință articolele pe care acesta alege să le urmărească adăugându-le în lista sa de preferințe.

Forumul din cadrul aplicației permite utilizatorilor să comunice și să creeze o comunitate.

8.2. Dezvoltări ulterioare

În perioada de dezvoltare s-au observat câteva posibile îmbunătățiri ce pot fi aduse aplicației cum ar fi:

- Mai multe variante de autentificare pentru utilizator
  - Folosind contul de Facebook
  - Crearea unui cont folosind email și parolă
- Posibilitatea de a avea un utilizator de tip Guest cu un număr limitat de benefici dar care poate verifica aplicația fără a crea un cont
- Transformarea forumului curent în mai multe forumuri specifice fiecărui articol
- Adăugarea unui sistem de monetizare cum ar fi AdMob oferit de Firebase

Capitolul 9. Bibliografie

[1] D. Niculescu, „Introducere în Programarea Android,” [Interactiv]. Available: https://ocw.cs.pub.ro/courses/eim/laboratoare/laborator01)#dokuwiki__top.

[2] ”Database Structure” 26 May 2018 [Online] https://firebase.google.com/docs/database/android/structure-data

[3] L. Moroney, The Definitive Guide to Firebase, Apress, 2017

[4] Asokan M. (2013). ANDROID Vs iOS – AN ANALYSIS. *INTERNATIONAL JOURNAL OF COMPUTER ENGINEERING & TEHNOLOGY(IJCET)*.

[5] G. Reese, “Cloud Computing Architectures”, O'Reilly Media, 2009

[6] P. Mell, T. Grance, “The NIST Definition of Cloud Computing”, 2011

[7] J. Friesen, “Learn Java for Android Development”, Apress, 2010

[8] Dawn&David Griffths Head First Android Developemnt 2nd Edition

[9] Michael Burton Android App Development for Dummies 3rd Edition

[10] R. Meier, “Professional Android 4 Application Development 3rd Edition”, Wrox, 2012

Anexa 1

Anexa 2

Anexa 3

Termen	Definiție
Anime	Animație produsă sau cu originea în Japonia
NIST	National Institute of Standards and Technology
SDK	Software Development Kit
CSV	Comma Separated Values
UI	User Interface