UNIVERSITATEA DIN BUCUREŞTI

FACULTATEA DE MATEMATICĂ ŞI INFORMATICĂ

SPECIALIZAREA INFORMATICĂ

Lucrare de licenţă

Pill Watch - Reamintirea administrării medicamentelor

Absolvent

Fritz Raluca-Mihaela

Coordonator ştiinţific

Lect. Univ. Dr. Radu-Ştefan Mincu

Bucureşti, iunie 2023

Rezumat

„Pill Watch” este o aplicație Android, dezvoltată cu ajutorul limbajului Kotlin, ce are ca scop reamintirea administrării medicamentelor prin intermediul unor notificări și alarme. Această lucrare prezintă etapele urmate în procesul de dezvoltare a acestei aplicații. Datorită progresului tehnologic, telefoanele inteligente sunt omniprezente, devenind astfel o platformă ideală pentru o asemenea aplicație.

Aceasta dispune de funcții multiple, printre care regăsindu-se: adăugarea medicamentelor și posibilitatea de setare a unor alarme, arhivarea și sortarea medicamentelor, modalități de personalizare a experienței utilizatorului (nume de utilizator, teme luminoase sau întunecate etc.), înregistrarea rezultatului administrării unui medicament (luat, amânat sau pierdut, în funcție de interacțiunea cu notificarea alarmei) și distribuirea istoricului medicamentelor administrate.

O particularitate importantă a acestei aplicații constă în posibilitatea selectării medicamentelor direct dintr-o bază de date (provenită din lista medicamentelor aprobate în România) și analizarea acestora în vederea existenței unor interacțiuni chimice între acestea, prin comunicarea cu o baza de date internațională. Aplicația are la dispoziție un server implementat în NodeJS Express, personalizat pentru funcțiile de preluare a medicamentelor aprobate, conversia acelui document într-un format accesibil și comunicarea cu baza de date internațională.

Totodată, „Pill Watch” dispune de funcții de autentificare (normală, folosind email și parolă sau prin intermediul contului de Google), notificări transmise tuturor utilizatorilor și stocarea datelor în cloud.

Abstract

"Pill Watch" is an Android application which acts as a reminder feature for medicine administration using notifications and alarms. This paper presents the steps taken in the development process of this application. Due to the technological progress, smartphones are now ubiquitous, being the ideal platform for this type of application.

It has multiple features, such as: adding medication and the possilibity to set up alarms, archiving and sorting the medication, user experience customization (username, light and dark themes etc.), logs for medicine intake status (taken, postponed or missed, based on the interaction with the alarm notification) and the option to share the medication history.

An important functionality of this application consists in the possibility of choosing a medication from a database (derived from a list of approved medication in Romania) and checking for chemical interactions between them, by communicating with an international database. The application has a custom-made server implemented in Node.JS Express, personalized with features which vary from retrieving the approved medicine, the ability to convert the document into an accesible format and communicating with the international database for the purpose metioned previously.

At the same time, "Pill Watch" has authenication features (the classic one, using an email and password, or by using the Google account), push notifications, which are sent to all the users, and storing the data in the cloud.

Cuprins

Cuprins

1 Introducere

1.1 Context şi motivaţie

Alegerea acestei teme are la bază doi factori majori, aprofundarea cunoștințelor în dezvoltarea aplicațiilor Android și implementarea unei soluții care ușurează respectarea tratamentelor medicale. Prin îmbinarea celor doi factori cu elementele tehnice și funcționale ce vor fi descrise în continuare, a fost realizată dezvoltarea aplicației „Pill Watch”.

Datorită activităților de zi cu zi, un studiu arată faptul că este normal să uităm să realizăm o multitudine de acțiuni pe care ni le propunem, iar o cale de combatere a acestor momente de „uitare”, este asocierea unor obiecte sau a altor activități cu acțiunile ce trebuie realizate.[1] Exemplele pot varia de la a uita să răspunzi la un mesaj (nu mai găsești notificarea mesajului la care ai spus că răspunzi ulterior), până la a uita să urmezi un tratament prescris de un specialist (pierzi noțiunea timpului, uiți ora la care trebuia administrat, lucruri care pot fi contracarate prin asocierea momentului cu o notificare pe telefon, o emisiune care începe mereu când este ora administrării etc.). În studiul de caz menționat anterior, trimiterea unei scrisori este asociată cu un buchet de flori aflat pe birou în proximitatea scrisorii. Momente de “uitare” s-au întâmplat cel puțin o data sau de doua ori în viața cuiva, fie prieteni, părinți sau mai ales bunici. În raport cu înaintarea în vârstă, crește numărul medicamentelor prescrise și există riscul ca din motive externe (pierderea noțiunii timpului, diverse ocupații în intervalul orar prestabilit pentru administrarea tratamentului etc.) să nu se respecte tratamentele medicale.

Trăim într-o epocă predominată de tehnologie, în care suntem înconjurați de dispozitive inteligente (telefoane, ceasuri, tablete, laptop-uri, sau chiar sisteme ultra performante), iar posibilitățile de dezvoltare sunt infinite. Având în vedere că, în 2023, aproximativ 86.29% persoane dețin un smartphone [2], am decis că o aplicație Android este platforma ideală pentru „Pill Watch”, fiind disponibilă pentru un număr mai mare de utilizatori prin publicarea aplicației pe magazine precum “Play Store” (Google), “AppGalery” (Huawei) etc.

Un alt studiu recent, realizat pe parcursul a 24 de luni, a urmărit impactul de lungă durată pe care l-au avut aplicațiile medicale de telefon asupra respectării tratamentelor medicale în cadrul unor pacienți care sufereau de diverse boli. Studiul a arătat că după 12 luni, procentul de respectare a tratamentelor a crescut de la 23% la 34%, scăzând apoi la 31% când s-au împlinit 24 de luni. Astfel, acest studiu a dovedit faptul că utilizarea aplicațiilor medicale de telefon influențează pozitiv respectarea tratamentelor, dar nu este susținut pentru toți pacienții.[3]

Reamintirea administrării medicamentelor în contextul societății de astăzi implică

influențarea utilizatorilor să își urmeze corespunzător tratamentele. Aplicația dispune de o funcție prin care statusul administrării unui medicament (luat, amânat, pierdut) este salvat pe baza interacțiunii cu notificările alarmelor setate împreună cu ora și data administrării. Acest istoric reprezintă o modalitate ușoară pentru persoane responsabile (părinți, medici, îngrijitori etc.) de a verifica că utilizatorul respecta tratamentul stabilit.

„Pill Watch" reprezintă un pas important, atât în dezvoltarea profesională, acumulând cunoștințe în domeniul dezvoltării aplicațiilor Android, cât și pe plan personal prin punerea la dispoziție a unui mod accesibil și ușor de personalizat pentru persoanele apropiate, în scopul respectării tratamentelor medicamentoase. Obiectivul final al acestui proces este publicarea aplicației, devenind accesibilă publicului din întreaga lume.

1.2 Aplicații similare

În urma unor căutări realizate până la data de 28 februarie 2023 în cadrul platformei „Play Store", de la Google, a fost realizat tabelul 1.1 unde se regăsesc 4 aplicații similare cu „Pill Watch", împreună cu scorul si recenziile acestora.[4] În următoarele rânduri sunt descrise în câteva cuvinte principalele funcții ale acestora împreună cu o sinteză rezultată în urma analizării recenziilor.

1) "MyTherapy Pill Reminder" este aplicația cu scorul cel mai mare (4.9) și cu un număr de recenzii ridicat (mai mare de 148.000 recenzii).[5]

- Dispune de funcții de setare a alarmelor, setarea personalizată a dozajului în funcție de perioada tratamentului, modalitate de salvare a statusului administrării medicamentului (luat, amânat, pierdut), distribuire raport medical către medicul specialist și opțiunea de înregistrare a parametrilor de greutate, presiunea sângelui, nivelul zahărului din sânge etc.[5]
- În urma verificării recenziilor, majoritatea utilizatorilor au fost plăcut impresionați de faptul că aplicația este foarte simplă și intuitivă, dar și de faptul că dezvoltatorii aplicației au îmbunătățit aplicația periodic. Pe de altă parte, există și recenzii negative din cauza absenței unei opțiuni de personalizare a notificărilor (text, sunetul folosit pentru alarme si volumul acestora sau opțiunea de a vibra, sau nu, în momentul declanșării alarmei). [5]

2) "Pill Log: Medication Reminder" se află pe locul 2 în funcție de scorul acordat de utilizatori, dar cu un număr de recenzii relativ scăzut față de celelalte aplicații (mai mic de 5.000 recenzii).[6]

- Funcțiile aplicației variază de la personalizarea sunetului alarmelor, la opțiunea

de a înregistra dozaje detaliate, urmărirea ușoară a progresului tratamentului și flexibilitatea de a modifica orele administrării medicamentelor.[6]

- După analiza recenziilor, un număr semnificativ de utilizatori sunt mulțumiți de calitatea și funcțiile aplicației. De asemenea, există propuneri de funcții noi, împreună cu raportarea diverselor probleme de funcționare ale programului.[6]

3) "Alarm and Pill Reminder"

- Două funcții particulare ale acestei aplicații sunt adăugarea automată a unei etichete colorate în funcție de următoarea dată când trebuie administrat acel medicament (roșu pentru ziua de astăzi, galben pentru mâine și verde pentru poimâine) și posibilitatea de înregistrare a orelor de somn și a zilelor libere pentru a nu declanșa alarmele în intervalele orare specificate sau în zilele libere introduse.[7]
- În urma verificării recenziilor, mulți utilizatori recomandă această aplicație, dar un număr mare dintre aceștia raportează faptul că absența unei opțiuni de amânare a alarmelor i-a influențat să folosească alte aplicații.[7]

4) "Medisafe Pill & Med Reminder" se află pe primul loc la numărul de recenzii (mai mare de 229.000 recenzii), dar pe ultimul loc în această listă privind scorul acestei aplicații (4.6). [8]

- Particularitățile acestei aplicații sunt: verificarea interacțiunilor chimice dintre medicamente, notificări pentru aprovizionarea de noi medicamente, rapoarte săptămânale și lunare despre statusul administrării medicamentului, posibilitatea de conectare cu un ceas inteligent, opțiunea de comunicare cu medicul direct din interfața aplicației, modalitatea de introducere a detaliilor legate de afecțiunea pentru care este administrat tratamentul și posibilitatea de a crea profile multiple. Un exemplu pentru utilitatea opțiunii profilelor multiple este faptul că un părinte poate separa medicamentele care trebuie administrate copilului față de tratamentele pe care le urmează părinții acestuia.[8]
- Recenziile acestei aplicații sunt majoritar pozitive, fiind cea mai populară aplicație de acest tip, cu peste 5 milioane de instalări. Mulți utilizatori au precizat că opțiunea de a folosi mai multe profile și faptul că nu a fost necesară crearea unui cont în prealabil, a fost principalul motiv pentru care folosesc aplicația. Pe de alta parte, alții au raportat faptul că interfața aplicației nu este intuitivă și că există multe momente în care funcțiile acesteia nu se comportă conform așteptărilor (nu apar notificări, alarmele nu sunt declanșate etc). De asemenea, numărul de notificări pe care le primește un utilizator pe săptămână a influențat multe persoane să dezinstaleze aplicația din telefon.[8]

Număr	Nume aplicație	Scor	Număr recenzii
1	MyTherapy Pill Reminder	4.9 ⭐	>148.000
2	Pill Log: Medication Reminder	4.9 ⭐	<5.000
3	Alarm and pill reminder	4.7 ⭐	>13.000
4	Medisafe Pill & Med Reminder	4.6 ⭐	>229.000

Tabel 1.1 *Top 4 Aplicații de tip 'pill reminder' cu recenzii găsite până la data de 28 februarie 2023 [4]*

1.3 Nevoi satisfăcute, elemente inovatoare şi funcţii principale

„Pill Watch" reprezintă o uniune a mai multor funcții care se regăsesc în celelalte aplicații, satisfăcând principala particularitate și nevoie a acestui tip de aplicație (reamintirea administrării medicamentelor), printre alte nevoi.

La adăugarea unui medicament există un câmp de introducere a unui text care pe baza literelor introduse generează o listă cu elemente selectabile ce reprezintă predicții posibile pentru medicamentul căutat. Aceste predicții sunt generate pe baza medicamentelor aprobate în România, preluate de la „Agenția Națională a Medicamentului din România" [9]. Această aplicație aduce ca element inovator prezența exclusivă a medicamentelor aprobate în România, un obiectiv fiind introducerea în viitor a medicamentelor aprobate în regiunea telefonului pe care este folosită aplicația. Pentru medicamentele selectate din această listă, se realizează un apel API către o bază de date internațională din cadrul „Librăriei Naționale de Medicină" [10] unde se verifică interacțiunea chimică dintre medicamentele utilizatorului și cel introdus. După adăugarea unui medicament se poate configura frecvența alarmelor împreună cu ora de începere a tratamentului. Mai multe detalii despre această etapă se găsesc în subcapitolul 3.7.

Particularitățile secundare ale aplicației variază de la elemente personalizabile la componentele interfeței grafice. Elementele personalizabile constau în posibilitatea setării unui nume de utilizator (ce poate reprezenta numele real sau o porecla aleatoare), schimbarea temei (luminoasă sau întunecată) și posibilitatea de ajustare a textului folosit pentru notificări etc. Componentele interfeței grafice sunt intuitive și ușor de utilizat. Acestea variază de la butoane cu pictograme universal folosite (pentru editarea diverselor elemente, ștergere de elemente, arhivare și filtrare), până la metode de evidențiere a istoricului administrării unui medicament prin prezența unor etichete (culori sugestive pentru fiecare stare posibilă: luat, amânat sau

pierdut) și ora de administrare a următorului medicament. Medicamentele nefolosite, care nu mai necesită administrare și alarme, pot fi șterse complet din cont, sau pot fi arhivate, opțiune ce asigură păstrarea istoricului administrării și posibilitatea refolosirii acestor medicamente în viitor. Aceste funcții sunt disponibile în cadrul paginii medicamentului respectiv.

Aplicația beneficiază și de un server creat de la 0 prin care se realizează diverse operații. Exemple de funcții pe care le îndeplinește server-ul sunt: verificarea interacțiunilor chimice dintre medicamente, preluarea medicamentelor de pe pagina web și realizarea conversiei documentului într-un format ce poate fi folosit de aplicație. Acest server a fost utilizat local până pe 25 mai, moment în care a fost găzduit pe o platformă de cloud, Heroku, pentru a putea fi accesat oricând și de pe orice dispozitiv, cu ușurință.[36]

Dezvoltarea aplicației a avut ca obiectiv secundar integrarea si respectarea mai multor paradigme de programare și modele de design, printre care programarea de obiecte în cadrul arhitecturii aplicației.

1.4 Organizarea lucrării

Lucrarea este structurată pe mai multe capitole, respectiv subcapitole. În cadrul acestora, este prezentat modul în care a fost implementată aplicația „Pill Watch” și oferă o reprezentare completă a evoluției și a versiunii finale a acesteia.

În **capitolul 2** sunt expuse tehnologiile și specificațiile arhitecturale folosite în procesul dezvoltării aplicației „Pill Watch”. Acest capitol este împărțit în 3 subcapitole care corespund specificațiilor arhitecturale și tehnologilor folosite în cadrul dezvoltării aplicației Android, respectiv în cadrul implementării server-ului NodeJS express. În fiecare subcapitol vor fi enumerate rolurile îndeplinite în cadrul aplicației, împreună cu exemple concrete din codul sursă.

În cadrul **capitolului 3** sunt evidențiate funcțiile aplicației prin intermediul parcurgerii elementelor importante (adăugarea unui medicament, pagina unui medicament, pagina de setări, editarea profilului, pagina de conectare etc.) acompaniate de imagini specifice preluate din aplicație. De asemenea, va fi descris raționamentul folosit și explicarea modului de abordare a acestuia în detaliu.

Capitolul 4 își propune să ofere o analiză în retrospectivă a obiectivelor inițiale ale aplicației și a rezultatelor obținute în urma dezvoltării sale. De asemenea, vor fi evidențiate planurile de viitor pentru aplicație, precum și abilitățile și competențele dobândite pe parcursul de dezvoltare.

2 Tehnologii folosite și specificații arhitecturale

În capitolul anterior, s-a menționat dezvoltarea aplicației Android propriu-zise și implementarea unui server personalizat. Combinând aceste două proiecte, obținem aplicația „Pill Watch”. În următoarele subcapitole, vor fi abordate specificațiile arhitecturale, reprezentarea bazei de date și o listă detaliată cu toate tehnologiile folosite în funcție de proiectul din care fac parte, Android sau server-ul NodeJS Express.

2.1 Arhitectură MVVM

Această aplicație folosește un model arhitectural care separă logica fiecărei pagini în Views (fragmente și activități) și ViewModels. Fragmentele și activitățile sunt dedicate ajustării elementelor din interfața utilizatorului. Un ViewModel reprezintă o legătură între datele aplicației (modele) și Views, toate operațiile necesare pentru interfața grafică se realizează în ViewModel și sunt transmise către View. Logica este separată aplicând principiul de „separation of concerns”, rezultând un cod curat, ușor de citit și evitând probleme ce pot apărea din cauza ciclului de viață al acestor Views.[11]

Arhitectura folosită are denumirea oficială de Model-View-ViewModel (MVVM). Acest model de arhitectură are la bază principiul menționat anterior, astfel încât un View are o conexiune unilaterală cu un ViewModel, iar un ViewModel are o conexiune unilaterală cu un Model. Acest mod de organizare în ViewModels este foarte folositor pentru că oferă flexibilitatea mai multor Views să folosească un singur ViewModel.[12]

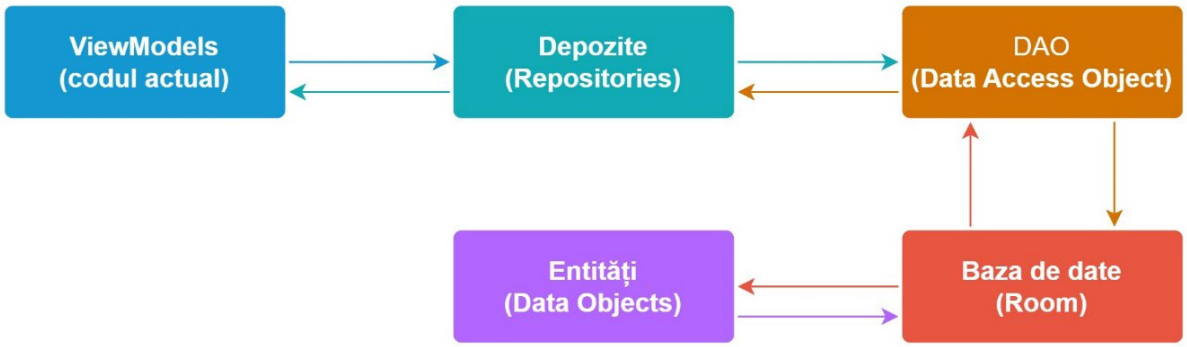


Figura 2.1 *Diagramă de clase pentru clasele din proiectul de Android simplificată*

În figura de mai sus (2.1) este prezentată o diagramă care evidențiază relațiile dintre clase și accesibilitatea disponibilă acestora. Astfel, în ViewModels se regăsește logica de afișare a datelor și sunt folosite depozite (repositories) pentru a obține datele necesare. Depozitele reprezintă modalitatea de comunicare dintre baza de date și ViewModels, având acces doar la obiectele de accesare a datelor (Data Object Access, pe scurt DAO). Un obiect de accesare a datelor (DAO) reprezintă o interfață în care sunt definite metode de interacțiune cu baza de date, împreună cu interogările (queries) specifice pentru a obține răspunsuri necesare de la baza de date. Baza de date este gestionată de Room (librărie specifică aplicațiilor Android care ușurează interacționarea cu SQLite) și folosește operațiile CRUD (Create, Read, Update, Delete) în gestionarea entităților. Entitățile sunt reprezentările datelor, sub formă de obiecte, în baza de date.

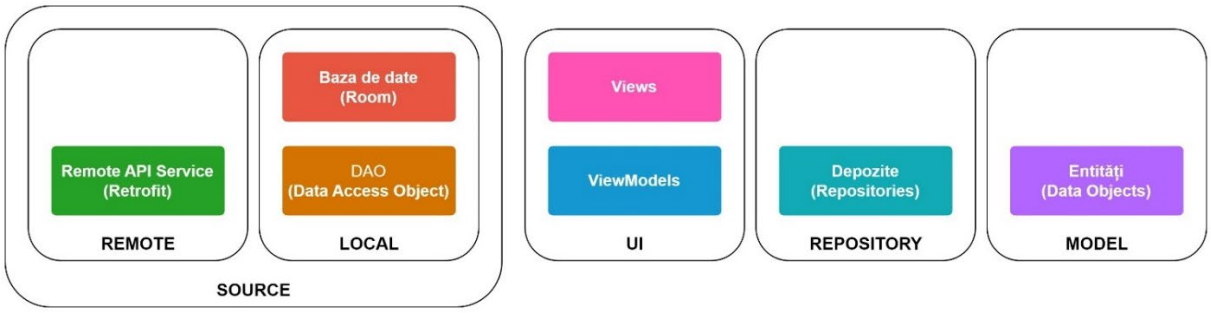


Figura 2.2 *Diagramă reprezentativă pentru localizarea elementelor structurale în proiect*

Pentru implementarea acestui model arhitectural, fișierele proiectului Android au fost împărțite în mai multe directoare și sub-directoare, conform figurii 2.2. Datorită naturii fișierelor Views, a fost creat un director specializat pentru interfața utilizatorului, denumit „ui”. În interiorul acestui director au fost create sub-directoare pentru fiecare fragment și activitate, denumite specific funcțiilor îndeplinite (exemplu: HomeFragment și HomeViewModel se regăsesc în sub-directorul „home”).

Pentru datele aplicației și modul de preluare al acestora, a fost creat un director, „data”. Principalele sub-directoare ale acestui director sunt denumite descriptiv în funcție de fișierele pe care le conțin „model”, pentru entitățile aplicației, „repository”, pentru depozite și „source”, pentru sursa din care este preluată informația. Sub-directorul „source” conține două sub-directoare: „local”, pentru datele stocate în baza de date și „remote”, pentru datele provenite dintr-o locație externă (exemplu: date provenite din server-ul aplicației).

2.2 Baza de date

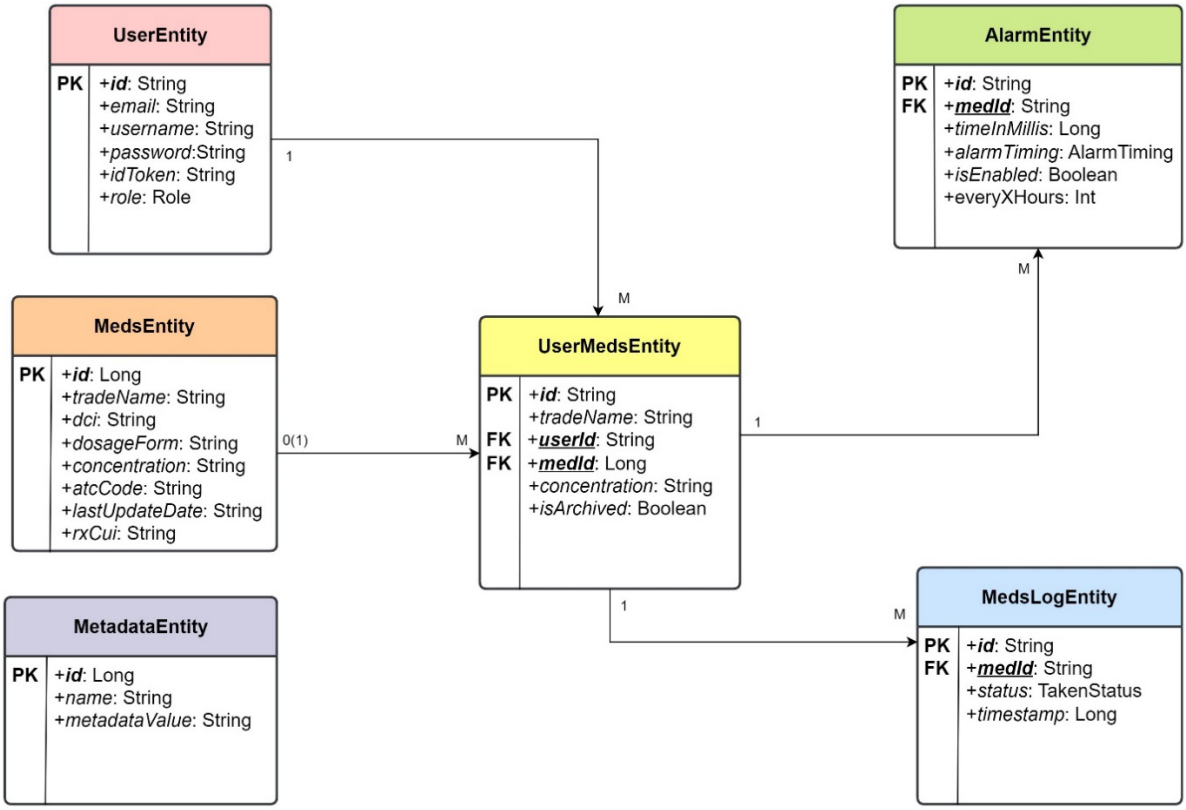


Figura 2.3 *– Diagramă Entitate-Relație*

Se poate observa în figura 2.3 că baza de date a aplicației dispune de 6 entități cu diferite relații între acestea. Urmărind structura relațiilor dintre entități, se poate observa faptul că „UserMeds” este dependent de „User” și „Meds” (valoarea cheii externe provenită din tabelul Meds poate fi nulă în momentul în care medicamentul introdus de utilizator nu este ales din predicțiile preluate din acel tabel). În continuare, este evidențiată și relația de dependență dintre entitățile „MedsLog”, respectiv „Alarm”, față de „UserMeds”.

„Metadata” reprezintă singura entitate care nu prezintă o relație cu alte tabele, din simplul fapt că în acest tabel este stocată cheia de securitate SHA corespunzătoare fișierului JSON transmis de către server. Mai multe detalii despre rolul acestei chei SHA folosite va fi descris in subcapitolul 2.4 dedicat funcțiilor server-ului.

În cadrul tabelului „Meds” sunt stocate medicamentele provenite din lista preluată de la „Asociația Națională a Medicamentului” din România, filtrată după existența acestora în baza de date de la „National Library of Medicine”. După filtrarea realizată prin intermediul implementărilor sever-ului, toate medicamentele salvate în tabelul „Meds” sunt disponibile

pentru verificarea posibilelor interacțiuni chimice.

Entitatea „UserMeds” reprezintă o modalitate de a asocia utilizatorii cu medicamentele alese pe baza predicției sau cu medicamentele introduse manual. În cadrul aplicației, un user poate face diferența între medicamentele preluate (cele care pot fi verificate pentru interacțiuni chimice) și cele introduse manual după adăugarea unui medicament prin verificarea pictogramei aflată lângă numele medicamentului în cadrul paginii de medicație (MedicationFragment). Mai multe detalii pe acest subiect vor fi oferite în capitolul 3.6, unde este prezentată pagina respectivă în amănunt. De asemenea, această entitate se află în strânsă legătură cu entitățile „Alarm” și „MedsLog”. Cele două sunt înlăturate din baza de date în corespondență cu ștergerea medicamentului de care sunt legate.

Entitatea „User” reprezintă obiectul prin care sunt stocate și gestionate informațiile unui utilizator. O particularitate a acestei entități este prezența unui atribut (Role) prin care se configurează rolul utilizatorului în cadrul aplicației. Acest atribut îndeplinește funcția de deosebire dintre un utilizator normal și unul cu acces la funcții administrative (admin). În prezent nu este o modalitate concretă de a adăuga rolul de admin, acest lucru fiind realizabil doar prin metode programatice (se verifică dacă email-ul folosit pentru înregistrare sau conectare există într-o listă predefinită de email-uri care necesită acel rol). De asemenea, cu excepția unui buton denumit „Clean”, ce are funcția de ștergere din baza de date a datelor din tabelele „UserMeds”, „Alarm” și „MedsLog”, nu există alte implementări care sunt disponibile doar pentru utilizatorii cu rolul de admin.

Entitățile „Alarm” și „MedsLog” au o proprietate comună, având unul din atribute un obiect de tip Enum. Pentru „Alarm”, acest atribut este „alarmTiming” care reprezintă frecvența declansării alarmei, exemple de valori posibile pentru acesta sunt: la fiecare X ore, unde X este valoarea atributului „everyXHours”, o data pe zi, de 2 ori pe zi, fără alarme etc. Pe de altă parte, „MedsLog” folosește atributul „status” pentru a înregistra starea administrării medicamentului, cu valorile posibile: luat, amânat și pierdut.

2.3 Aplicație Android

Aplicația a fost dezvoltată pentru dispozitivele mobile datorită vitezei de evoluție a domeniului, precum și flexibilității și confortului oferite de aplicațiile mobile în comparație cu cele web. În continuare vor fi enumerate și explicate tehnologiile folosite în cadrul aplicației.

2.3.1 Dagger

Această tehnologie este un framework prin intermediul căruia se realizează injectarea dependențelor în momentul compilării, pentru Java, Kotlin și Android. În prezent, acest framework este dezvoltat cu ajutorul companiei Google.[13]

Principiul de bază este injectarea inversă a dependențelor (Inversion of Control – IoC) prin care un obiect primește dependențele prin intermediul unui constructor sau ca parametru al unei metode. Prin folosirea acestui framework, este generat automat codul de injectare în momentul compilării aplicației.

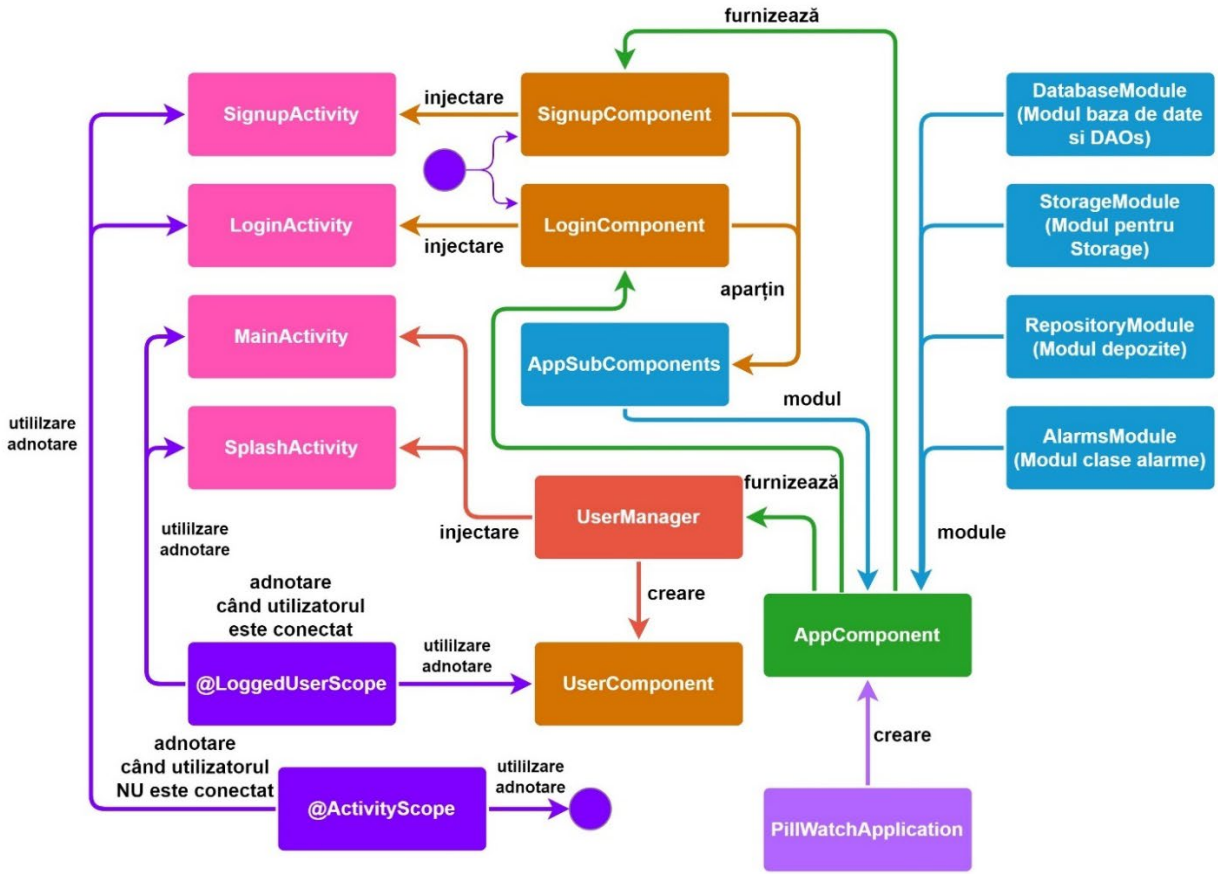


Figura 2.4 *Diagrama procesului de injectare a dependențelor cu ajutorul Dagger în aplicația „Pill Watch”*

Modul de funcționare al întregului proces de injectare este prezentat în figura 2.4. Există mai multe module prin intermediul cărora se detaliază modalitatea în care sunt puse la dispoziție obiectele ce urmează să fie injectate în alte clase. Aceste module variază de la: unul folosit pentru baza de date și DAO-urile respective, pentru interfața care oferă acces la SharedPreferences (spațiul de stocare intern al aplicației), pentru depozite, pentru clase care se

ocupă de generarea și gestionarea alarmelor (AlarmGenerator și AlarmHandler) și un modul care are în definiția sa cele 3 subcomponente ale aplicației.

Vorbind despre subcomponente, acestea au fost create pentru a diferenția ciclurile de viața ale anumitor obiecte injectate. Fiecare subcomponent dispune de o adnotare specifică înainte de definirea subcomponentului și a claselor care corespund acestuia. În proiect am definit două astfel de adnotări: LoggedUserScope și ActivityScope pentru definirea domeniului de aplicare. În continuare vor fi descrise aceste domenii de aplicare, împreună cu rolul utilizării lor.

Când este folosit LoggedUserScope, se presupune că s-a efectuat cu succes conectarea utilizatorului și clasa UserManager, care corespunde subcomponentului UserSubcomponent, urmează să fie un obiect unic pentru toate clasele care îl vor folosi și va fi modalitatea prin care se activează procesul de injectare prin Dagger în clasele respective. Potrivit figurii anterioare, UserManager produce injectarea în MainActivity și SplashActivity. Un lucru care nu este menționat în figura anterioară este faptul că acest proces de injectare este repetat în fragmentele care aparțin acestor activități prin suprascrierea funcții „onAttach” (adică în momentul atașării acestui fragment la o activitate, se realizează acea injectare în mod similar activității).

Având în vedere natura similară dintre subcomponentele și activitățile corespunzătoare conectării și înregistrării unui utilizator, a fost folosită o adnotare pentru ambele cazuri. Un alt factor luat în considerare pentru această alegere este faptul că un utilizator, după conectare sau înregistrare, poate să nu dispună de un nume de utilizator salvat în baza de date. Dacă se întâmplă acest lucru, utilizatorul este redirecționat către un fragment dedicat acestei acțiuni, UsernameChangeFragment. Adnotările dispun de o proprietate prin care doar o adnotare poate fi folosită per clasa, astfel fiind necesară folosirea unei adnotări comune între cele două subcomponente și activități.

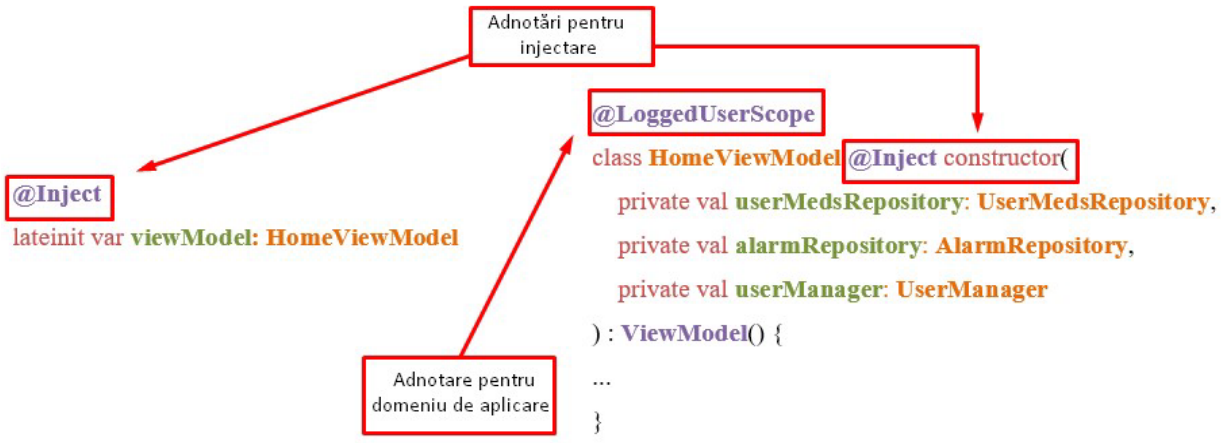


Figura 2.5 *Exemplu folosire adnotări în contextul injectării dependențelor cu Dagger*

În figura de mai sus, 2.5, este redat un exemplu din cadrul proiectului pentru injectarea dependențelor realizată prin folosirea adnotării @Inject. Aceasta trebuie introdusă înaintea declarării unui obiect și în momentul declarării constructorului. Pasul din urmă realizându-se în definiția clasei.

2.3.2 Firebase

Firebase este o platformă dezvoltată de Google pentru crearea aplicațiilor web și de telefon. Platforma oferă mai multe servicii printre care Cloud Firestore, baza de date în timp real, trimitere de notificări și mesaje prin Cloud Messaging, autentificare prin Firebase Authentication, și emulatoare menite pentru testarea aplicațiilor.[14] În continuare vor fi enumerate serviciile utilizate în cadrul aplicației „Pill Watch".

❖ **Cloud Firestore** este o bază de date, tip NoSQL, unde datele sunt stocate în cloud. Principalele funcții reprezintă posibilitatea de sincronizare a datelor utilizatorilor în timp real, integrare perfectă cu celelalte servicii oferite de Firebase și posibilități standard sau personalizate pentru abordarea situațiilor în care nu există conexiune la internet.[15] Pentru „Pill Watch", acest serviciu este folosit pentru stocarea datelor utilizatorilor, împreună cu istoricul tratamentelor sale, într-o platformă cloud.

❖ **Cloud Messaging** reprezintă o modalitate de comunicare prin mesaje sau notificări între mai multe platforme gratuit. Se pot trimite diverse mesaje către toți utilizatorii aplicației sau doar către cei care se află în momentul respectiv în aplicație.[16] În cadrul proiectului, serviciul acesta este folosit pentru trimiterea unei notificări către toți utilizatorii aplicației în momentul efectuării unui anunț.

❖ **Authentication** reprezintă serviciul care oferă servicii de autentificare pentru aplicații. Acesta susține autentificarea clasică, prin email și parolă, sau prin modalități moderne, utilizarea numărului de telefon sau prin intermediul conturilor de Google, Facebook, GitHub sau Twitter. Acest serviciu oferă o modalitate ușoară de a verifica identitatea utilizatorului și securitatea datelor. Serviciul dispune și de posibilitatea integrării unei autentificări care se bazează pe mai mulți factori, de exemplu: autentificarea prin 2 factori (termenul în engleză fiind 2 factor authentication sau 2FA) prin utilizarea unor alte aplicații specializate pentru această funcție sau prin trimiterea unui SMS pe telefon.[17] În cadrul „Pill Watch", autentificarea se realizează prin email și parolă, sau prin utilizarea unui cont de Google.

2.3.3 Room

Tehnologia este o librărie care face parte din Android Jetpack (colecție de librării cu scopul de a reduce cantitatea de cod care se repetă în cadrul unei aplicații și pentru a oferi consistență codului pentru mai multe versiuni de Android și telefoane[19] și este folosit pentru a oferi persistența locală a datelor, împreună cu o modalitate ușoară de stocare a acestora. Room oferă un nivel de abstractizare peste SQLite pentru un acces fluent la baza de date. De asemenea, librăria oferă adnotări intuitive pentru evitarea repetării codului și dispune de modalități de migrare ale bazei de date, precum și verificarea interogărilor SQL în momentul compilării.[18]

2.3.4 Retrofit

Retrofit reprezintă o librărie care realizează cereri HTTP pentru programele Android și Java. Acesta funcționează pe principiul transformării unei interfețe introdusă de programator într-un HTTP API. Fiecare cerere HTTP realizată prin intermediul Retrofit poate sa fie sincronă sau asincronă. Această librărie dispune de adnotări specifice limbajului Java și Kotlin pentru a defini aceste cereri. Răspunsul primit este deserializat și transformat, în funcție de caz, într-un format ce poate fi folosit în codul pentru Android. De asemenea, librăria dispune de posibilități multiple de conversie prin intermediul altor tehnologii: Moshi, GSON, Scalars etc.[20]

În cadrul proiectului, Retrofit este folosit momentul în care se realizează actualizarea medicamentelor de la ANM și verificarea existenței interacțiunilor chimice dintre medicamente în momentul în care este introdus un medicament nou dintre cele de la ANM.

2.3.5 Moshi

Moshi reprezintă o librărie pentru Android, Java și Kotlin prin intermediul căreia se realizează conversia din obiecte din format JSON în obiecte specifice claselor din Java și Kotlin. Procesul de conversie este disponibil și invers, din obiecte Java sau Kotlin în obiecte JSON. Această librărie dispune de posiblitatea folosirii tipurilor clasice (Integer, Long, String etc.), dar și de posibilitatea de creare a unui adaptor pentru tipuri de obiecte personalizate. De asemenea, librăria oferă posibilitatea de ajustare a numelor folosite pentru obiectul JSON în momentul realizării conversiei, prin intermediul folosirii adnotării @JSON (name=”exemplu nume”), în situațiile în care numele din JSON și din clasa Kotlin nu corespund.[21]

În cadrul aplicației, Moshi este folosit pentru a transforma, din JSON în obiecte Kotlin, răspunsurile apelurilor trimise către server pentru actualizarea datelor medicamentelor și rezultatul verificării existenței interacțiunilor chimice dintre acestea. Mai jos este exemplificată clasa în care este folosită librăria Moshi, folosind adnotările menționate anterior, pentru a

transforma un obiect JSON într-un obiect MedsDataProperty.

```
@JsonClass(generateAdapter = true)
data class MedsDataProperty(
    @Json(name = "Trade name") val tradeName: String,
    @Json(name = "DCI") val dci: String,
    @Json(name = "Dosage Form") val dosageForm: String,
    @Json(name = "Concentration") val concentration: String = "",
    @Json(name = "ATC Code") val atcCode: String,
    @Json(name = "Last Update Date") val lastUpdateDate: String,
    @Json(name = "RxCui") val rxCui: String
)
```

2.3.6 AutoCompleteTextView

Acesta reprezintă o clasă în Android folosită pentru a oferi automat sugestii pentru un text, pe măsură ce acesta este introdus. Acesta moștenește atributele și metodele clasei EditText fiind un câmp editabil pentru introducerea unui text. Sugestiile sunt prezentate într-un meniu din care utilizatorul poate alege texul căutat. Modul de funcționare al acestei clase implică definirea unui adaptor pentru a afișa datele corespunzător.[22]

„Pill Watch” folosește un astfel de element în cadrul procesului de adăugare al unui medicament nou. Pentru afișarea acestui element, este implementat un adaptor care preia datele din ViewModel în funcție de textul introdus. Datele sunt reprezentate de numele medicamentului folosit în comerț și de concentrația acestuia. Acest fapt este datorat existenței unor medicamente cu același nume, dar care conțin concentrații diferite.

2.3.7 RecyclerView

RecyclerView reprezintă o librărie din Android care oferă o modalitate de a crea liste dinamice și o cale eficientă de a afișa o listă de elemente cu dimensiuni mari. În cadrul acesteia, pot fi furnizate datele necesare și apoi poate fi personalizat modul de afișare al acestora după bunul plac prin intermediul configurării unui adaptor. Librăria creează elementele în mod dinamic doar când este nevoie de acestea. După cum implică și numele, elementele sunt reciclate în momentul în care nu mai apar pe ecran prin reutilizarea unui View în loc de a îl distruge. Astfel, această librărie crește performanța aplicației și consumă mai puțină energie.[23] În cazul de față, această librărie este folosită în mai multe fragmente pentru a afișa listele de obiecte în mod dinamic și personalizat, datorită caracteristicilor sale.

2.3.8 Coroutines

Coroutines reprezintă o caracteristică din limbajul de programare Kotlin care permite scrierea codului asincron într-o manieră care nu blochează aplicația. Acestea sunt esențiale în

dezvoltarea aplicațiilor Android datorită faptului că permit executarea operațiilor lungi prin intermediul unor fire de execuție din separate pentru a nu bloca firul de execuție principal. Exemple pentru astfel de operații sunt: apelarea unui API de la care se așteaptă un răspuns sau metode care apelează baza de date. Acestea funcționează prin suspendarea firului de execuție pentru așteptarea finalizării unei operații. Un alt beneficiu al coroutines este reducerea numărului de pierderi de memorie și integrarea cu mai multe librării din Android Jetpack.[24] În cadrul aplicației, coroutines sunt folosite extrem de frecvent pentru apelurile către server sau când se interacționează cu baza de date.

2.3.9 AlarmManager

Acesta reprezintă o clasă în Android care oferă acces la serviciile pentru alarme ale sistemului. Prin intermediul acestor servicii, se poate programa executarea unei operațiuni în viitor, însă vor fi dezactivate dacă dispozitivul este închis sau repornit. [25] Aplicația folosește această clasă pentru a programa declanșarea alarmelor.

2.3.10 WorkManager

Această tehnologie reprezintă o librărie care face parte din Android Jetpack și are rolul de a asigura persistența diverselor sarcini chiar și după repornirea dispozitivului, împreună cu metode de repetare a unor sarcini conform unui program prestabilit. Prezintă mai multe opțiuni de utilizare: programarea realizărilor unor sarcini imediate, de lungă sau scurtă durată sau programarea periodică (de exemplu: în fiecare oră). Această librărie poate fi folosită concomitent cu Coroutines pentru a efectua sarcinile pe mai multe fire de execuție.[27]

În cadrul proiectului „Pill Watch”, această librărie este folosită în două moduri, cu scopul de a gestiona programarea și generarea alarmelor. În cadrul clasei PillWatchApplication, este programată o sarcină periodică, cu intervalul de o oră. Pe de altă parte, în interiorul activității MainActivity, este programată o singură sarcină ce trebuie să se realizeze imediat pentru programarea sau regenerarea alarmelor. Această sarcină se realizează pentru alarme care au fost declanșate, sau pentru care timpul declanșării alarmei corespundea cu perioada în care telefonul sau aplicația nu au fost funcționale din diverse motive. În momentul declanșării unei alarme, acest proces de regenerare se declanșează în mod automat când alarma declanșată este ultima alarmă din cadrul setului de alarme, dar dacă aplicația sau telefonul nu sunt funcționale în acel moment, acest proces de regenerare nu se realizează. Scopul acestor sarcini reprezintă evitarea situației în care alarmele nu sunt regenerate automat.

2.4 Server Node.JS Express

În etapa inițiala a procesului de dezvoltare a aplicației, transformarea fișierului din format Excel în format JSON, a reprezentat o provocare fiind necesare mai multe încercări pentru a identifica o soluție optimă. Inițial a fost încercată abordarea în cadrul aplicației Android utilizând Kotlin, apoi prin intermediul limbajului de programare Java. Într-un final, după constatarea limitărilor în celelalte abordări, a fost descoperită soluția optimă: un framework pentru NodeJS, platformă de tip open-source folosită pentru crearea unor aplicații adaptabile prin intermediul limbajului de scriptare JavaScript.[27]

„Express" (cunoscut și ca „Express.js") reprezintă un framework rapid și minimalist pentru NodeJS cu scopul dezvoltării unei aplicații web și implementarea unor interfețe de programare a aplicațiilor, cunoscute ca APIs.[27] Pe lângă scopul inițial de conversie a fișierului EXCEL, pe parcursul timpului s-a dovedit să fie alegerea ideală datorită modului simplu de transmitere a informației prin intermediul API-urilor.

Severul a fost găzduit pe platforma de cloud Heroku la finalul lunii mai.[36] Acest lucru a fost realizat pentru a oferi posibilitatea accesării API-urilor server-ului de pe orice dispozitiv în orice moment. Înainte de găzduirea online, accesarea API-urilor putea fi efectuată doar când server-ul era pornit manual pe un calculator care se afla în aceeași rețea cu dispozitivul de pe care se testa aplicația (emulator).

Funcțiile principale ale server-ului sunt verificarea actualizării medicamentelor de pe paginile oficiale prin verificarea datei calendaristice ultimei actualizări, preluarea documentului cu medicamente actualizate în mod automat, comunicarea cu o bază de date internațională prin intermediul a 2 API-uri, realizarea conversiei din format Excel în format JSON, interpretarea răspunsurilor primite prin API-uri și transmiterea acestora într-un format accesibil, conversie din format XML în format JSON și metode de interpretare a codului HTML pentru preluarea datelor necesare.

```
3   const express = require("express");
4   const excelToJson = require("convert-excel-to-json");
5   const axios = require("axios");
6   const xml2js = require("xml2js");
7   const fs = require("fs");
8   const crypto = require("crypto");
9   const cron = require("node-cron");
10  const HTMLParser = require("node-html-parser");
11  const util = require("./json/util.json");
12
13  const server = express();
```

Figura 2.6 *Secvență de cod în care se regăsesc modulele importate*

Conform figurii de mai sus (2.6), în cadrul proiectului s-au folosit mai multe tehnologii importate prin intermediul modulelor respective. Liniile de cod 3 și 13 regăsite în figura anterioară reprezintă modul de importare, respectiv inițializare, a unui server „Express”. De asemenea, linia 11 corespunde importării unui modul local prin care se salvează ultima dată calendaristică la care s-au actualizate medicamentele preluate de pe ANM [9] într-un fișier local. Această variabilă este importată la pornirea server-ului pentru a avea o evidență clară a acesteia. Aceasta este ulterior utilizată în vederea preluării documentului actualizat în cazul în care această dată este mai mică decât cea preluată de pe pagina web a Agenției Naționale a Medicamentului (ANM) [9]. În continuare, fiecare dintre celelalte tehnologii vor fi explicate cu detalii legate despre utilitatea acestora în codul proiectului.

2.4.1 Convert-excel-to-json

Această tehnologie reprezintă o librărie cu titlu extrem de sugestiv utilității acesteia, realizând conversia unui fișier de tip Excel într-un fișier de tip JSON. Acest lucru se poate realiza prin metode simple menționând distanța dintre 2 coloane din cadrul fișierului sau prin metode mai specifice prin care se menționează manual numele pe care îl dorim în cadrul obiectului JSON corespondent fiecărei coloane. De asemenea, unele coloane pot fi omise în momentul realizării conversiei. O altă particularitate esențială a acestei librării este posibilitatea de a omite antetul fișierului Excel, în vederea obținerii concrete a datelor din fișier.[28]

În cazul de față, se utilizează această librărie doar în momentul în care se realizează conversia unui fișier excel prin apelarea API-ului sau când se descarcă o variantă actualizată a acestuia. Din documentul descărcat, doar 6 din 20 de coloane conțin informații necesare pentru contextul aplicației, rezultând în preluarea coloanelor respective, cu nume sugestive.

2.4.2 Axios

Tehnologia Axios reprezintă o librărie folosită în JavaScript și NodeJS pentru efectuarea unor solicitări HTTP și preluarea anumitor date de pe pagini web. Aceasta are posibilitatea de transformare a datelor înainte de efectuarea solicitărilor, împreună cu anularea acestora. Răspunsurile primite pot fi transformate, deoarece funcționează asincron, pe principiul promisiunilor, așteptând răspunsurile de la sursă.[29]

În cazul proiectului curent, această librărie este folosită în primul rând pentru a prelua data calendaristică care corespunde ultimei actualizări a medicamentelor de pe pagina web a ANM. În al doilea rând, Axios este folosită pentru descărcarea documentului excel ce conține medicamentele actualizate. În etapele ulterioare, librăria este utilizată pentru a accesa 2 API-uri

furnizat de National Institute of Health (NIH din cadrul NLM) [10]. Unul din API-uri are scopul de a prelua codul RxCui care corespunde codului ATC al unui medicament preluat din documentul excel de la ANM[30]. Codul RxCui reprezintă modul de identificare a unui medicament în baza de date a celor de la NIH în vederea verificării ulterioare pentru interacțiuni chimice. Astfel, cel de-al doilea API este folosit pentru testarea acestor interacțiuni.

2.4.3 xml2js

Această tehnologie reprezintă un modul pentru NodeJS prin intermediul căruia elementele din formatul XML sunt transformate în format JSON. De asemenea, modulul dispune de o funcție „builder" prin care un obiect JSON poate fi transformat înapoi în format XML. O particularitate importantă este faptul că pot fi transformate mai multe fișiere în același timp, însă, această funcție nu este folosită în aplicația curentă. În schimb, particularitatea folosită în cadrul proiectului este posibilitatea de utilizare în mod asincron prin utilizarea promisiunilor.[31] Modulul este utilizat în cadrul procesului de preluare căutare a unui medicament în cadrul răspunsului provenit de la NIH[10], prin transformarea din formatul XML în JSON, cu scopul de a prelua doar eticheta corespunzătoare codului RxCui din obiectul JSON obținut.

2.4.4 fs (FileSystem)

Tehnologia „fs" este prescurtarea pentru FileSystem, un modul oficial pentru Node.JS care oferă opțiunea de a interacționa cu fișierele din sistem. În situația actuală, singurele funcții folosite din acest modul sunt pentru citirea și scrierea unui fișier.[32] Documentele pentru care se realizează cele două funcții sunt cel de „util" (pentru ultima dată calendaristică menționată anterior), cel de tip Excel cu lista de medicamente preluate de la ANM și cel de tip JSON care reprezintă conversia și completarea documentului Excel.

2.4.5 crypto

Crypto, precum FileSystem, reprezintă un modul oficial ce se regăsește în documentația Node.JS. Acesta are funcționalități criptografice de criptare, decriptare, semnare și verificare a certificatelor.[33]

În cadrul proiectului, acest modul este folosit pentru a cripta valoarea fișierului în format JSON (ce conține medicamentele preluate de la ANM împreună cu adăugarea codului RxCui) sub forma unei chei SHA. Această cheie este stocata în baza de date a aplicației Android, pentru a putea verifica (la fiecare conectare sau la apăsarea opțiunii de verificare pentru actualizări)

dacă datele provenite de la server și datele stocate în baza de date sunt identice. Aceasta verificare se bazează pe faptul că în momentul în care datele provenite sunt diferite, cheia SHA a fișierului JSON „data.json” se schimbă în funcție de datele conținute. Astfel, dacă cheia SHA aflată în baza de date și cheia SHA primită sunt diferite, este necesară o actualizare a datelor, moment în care pornește procesul de preluare a datelor noi.

2.4.6 cron

Această tehnologie reprezintă un modul pentru Node.JS prin care se programează sarcini ce vor fi realizate în funcție de setările stabilite. Pentru programarea sarcinilor, pot fi stabilite: secunda, minutul, ora, ziua din lună, luna și ziua din săptămână. Valorile irelevante pot fi înlocuite cu „*” în cadrul funcției de programare.[34]

În cadrul proiectului, acest modul este folosit pentru programarea unei verificări a datei calendaristice a ultimei actualizări a medicamentelor de pe platforma ANM. Verificarea constă în compararea datei preluate de pe pagina web și cea stocată local și este programată în fiecare zi de luni, la ora 2 dimineața UTC, corespunzătoare cu ora 5 a României. Dacă în urma verificării se dovedește faptul că documentul a fost actualizat (data de pe pagina web este mai recentă), se pornește procesul descărcării noului document, împreună cu realizarea conversiei în format JSON.

```
2023-06-12T02:52:43.735772+00:00 app[web.1]: No.32231 retrieved successfully.
2023-06-12T02:52:43.735786+00:00 app[web.1]: No.32232 already in the system.
2023-06-12T02:52:43.735806+00:00 app[web.1]: No.32233 already in the system.
2023-06-12T02:52:43.735935+00:00 app[web.1]: No.32234 retrieved successfully.
2023-06-12T02:52:43.736068+00:00 app[web.1]: No.32235 retrieved successfully.
2023-06-12T02:52:43.736078+00:00 app[web.1]: No.32236 already in the system.
2023-06-12T02:52:43.736722+00:00 app[web.1]: No.32237 retrieved successfully.
2023-06-12T02:52:43.736942+00:00 app[web.1]: No.32238 retrieved successfully.
2023-06-12T02:52:43.737104+00:00 app[web.1]: No.32239 retrieved successfully.
2023-06-12T02:52:43.737116+00:00 app[web.1]: No.32240 already in the system.
2023-06-12T02:52:43.747185+00:00 app[web.1]: 25774
2023-06-12T02:52:43.747203+00:00 app[web.1]: Medicine count found in the NIH database: 25774 out of 32240.
2023-06-12T02:52:43.747204+00:00 app[web.1]: Percentage of medicine found in the NIH database: 79.94 %.
```

Figura 2.7 *Ultimele înregistrări din log-ul aplicației găzduite pe Heroku [36] după realizarea unei actualizări automate pe data de 12 iunie 2023*

Figura 2.6 ilustrează cele mai recente operațiuni din cadrul procesului automat de actualizare a datelor. Procesul a fost activat deoarece data calendaristică preluată în ziua de luni, 12 iunie 2023, de pe site pagina ANM [9] a fost mai recentă decât cea salvată local. Acesta a rezultat în descărcarea fișierului actualizat, realizarea conversiei în JSON, împreună cu extragerea codurilor RxCui din baza de date de la NIH [10]. În plus, figura ne oferă o vedere clară asupra procentului de medicamente care au fost identificate în baza de date de la NIH și pentru care s-a efectuat extragerea codului RxCui.

2.4.7 node-html-parser

Node-html-parser reprezintă un modul Node.JS care analizează codul HTML, transformă structura codului HTML analizat într-un format ce poate fi filtrat și din care pot fi selectate elementele necesare.[35] În cadrul proiectului, acest modul este folosit pentru extragerea informațiilor paginii web ANM [9] cu scopul de a prelua data calendaristică corespunzătoare ultimei actualizări.

3 Descrierea Aplicației

„Pill Watch” dispune de mai multe ecrane ce pot fi accesate prin unul sau mai multe moduri. În figura de mai jos (3.1) este prezentat fluxul pe care îl poate urma un utilizator pentru naviga printre ecranele aplicației. În continuare vor fi enumerate elemente notabile și ecranele aplicației împreună cu funcțiile acestora.

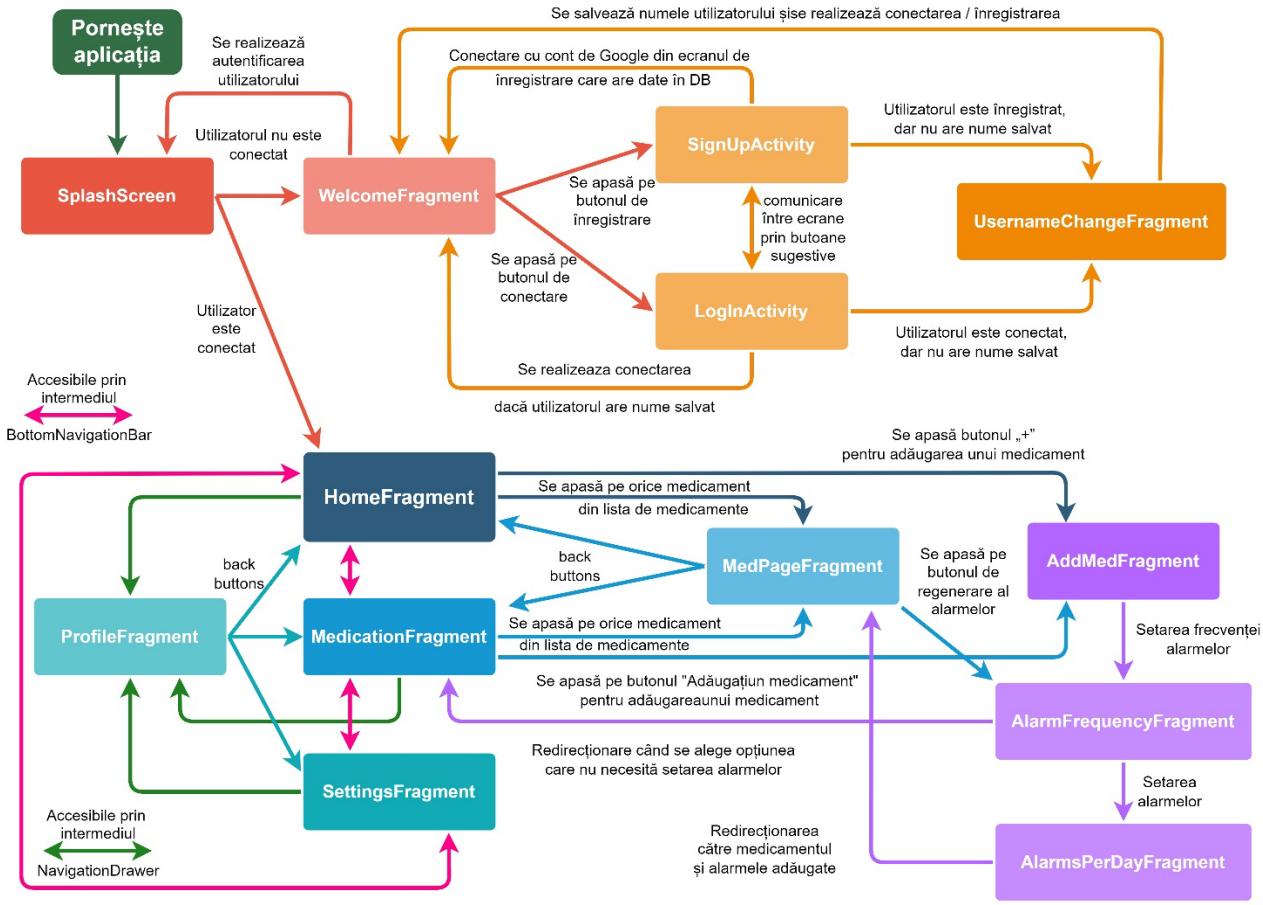


Figura 3.1 *Diagrama fluxului utilizatorului*

3.1 Interfața utilizatorului

„Pill Watch” folosește un logo cu design distinctiv, bine definit, care prezintă similarități cu un ochi (linia genelor și două tipuri de pastile, dispuse alăturat, în locul ochiului propriu-zis). Prin combinarea acestor elemente este transmis mesajul din spatele numelui aplicației, vegherea asupra administrării responsabile a medicamentelor.

Aplicația dispune de mai multe pictograme, 2 fiind reprezentative pentru statusurile verificat și neverificat, altele fiind sugestive funcției pe care o îndeplinesc (editare, ștergere, distribuire, adăugare etc.). Pe lângă acestea, există o pictogramă care se concentrează pe elementul cu medicamente din logo și este folosită pentru fiecare medicament dintr-o listă. În

figura de mai jos (3.2) sunt reprezentate logo-ul, pictograma creată dintr-o porțiune a logo-ului și o selecție din paleta de culori regăsită în aplicația „Pill Watch”.

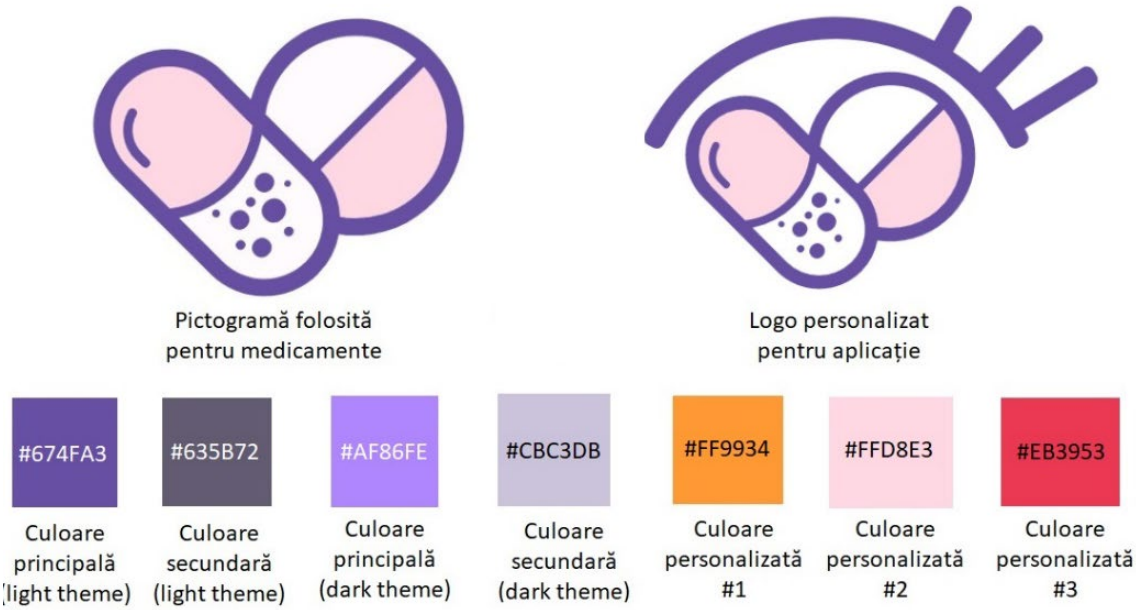


Figura 3.2 *Logo și o selecție a paletei de culori*

Un alt element care reprezintă o proporție din interfața utilizatorului este faptul că există opțiunea de a utiliza una dintre cele două limbi, română și engleză, în funcție de setările telefonului. Prezența limbii engleze în setările telefonului implică automat schimbarea aplicației în limba engleză. Altfel, dacă telefonul are setată limba română ca fiind singura limbă, aplicația va fi prezentată în limba română.

3.2 Ecran de pornire și de întâmpinare

În ilustrația prezentată mai jos (figura 3.3) se poate observa fluxul de navigare pe care îl parcurge un utilizator când deschide aplicația pentru prima dată sau când se deconectează. În principiu, un utilizator poate ajunge la ecranul de întâmpinare doar când acesta nu este autentificat.

În partea stângă, este reprezentat ecranul Splash, care constituie prima interacțiune a utilizatorului cu aplicația „Pill Watch”. Pe acest ecran, sunt afișate elementele cheie, cum ar fi: numele aplicației, logo-ul și un scurt motto care reflectă valorile aplicației. Un element important este reprezentat de un loading spinner care indică utilizatorului că aplicația se află în proces de încărcare. La finalul ecranului există un mesaj personalizat de bun venit, ce se adaptează în funcție de statusul de conectare al utilizatorului, adăugând numele configurat anterior la finalul mesajului când acesta se conectează cu succes.

Pe partea dreaptă se află ecranul de întâmpinare al utilizatorului, unde este prezentat rolul aplicației printr-un text concis. De asemenea, utilizatorul este avertizat de faptul că aplicația nu are scopul de a înlocui sfaturile medicale oferite de specialiști. În această pagină sunt evidențiate și două butoane distincte care direcționează utilizatorul către pagina de înregistrare, respectiv de conectare.

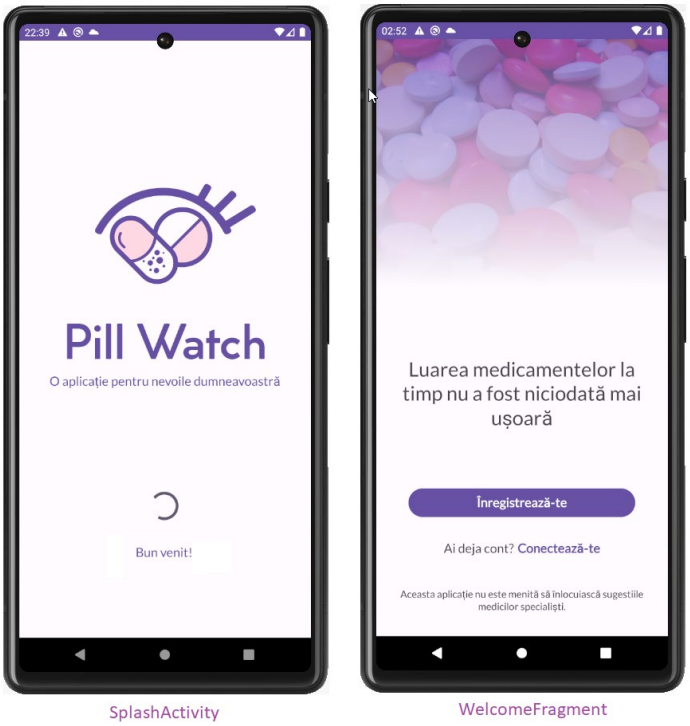


Figura 3.3 *Ecran de pornire și de întâmpinare a utilizatorului*

În ceea ce privește funcționalitățile speciale, în cadrul ecranului Splash se realizează importarea medicamentelor aprobate în România când tabelul din baza de date corespunzător acestor medicamente este gol. Procesul de importare este efectuat, prin intermediul tehnologiei Retrofit, apelând API-urile de pe server-ul personalizat.

3.3 Paginile de înregistrare și conectare

Cele trei capturi de ecran prezentate în figura 3.4 ilustrează trei ecrane diferite: de înregistrare, de conectare și de configurare a numelui de utilizator. În cadrul paginilor de autentificare, câmpurile specifice pentru email, parolă și confirmarea parolei (unde este cazul), sunt supuse unui sistem de validare a datelor. Acest sistem asigură faptul că doar email-urile care respectă tiparul standard, parolele de cel puțin 8 caractere și parolele care se potrivesc sunt acceptate, în timp ce câmpurile goale nu sunt permise. În momentul în care oricare dintre aceste etape de validare nu este respectată, un mesaj sugestiv va fi afișat în partea inferioară a ecranului.

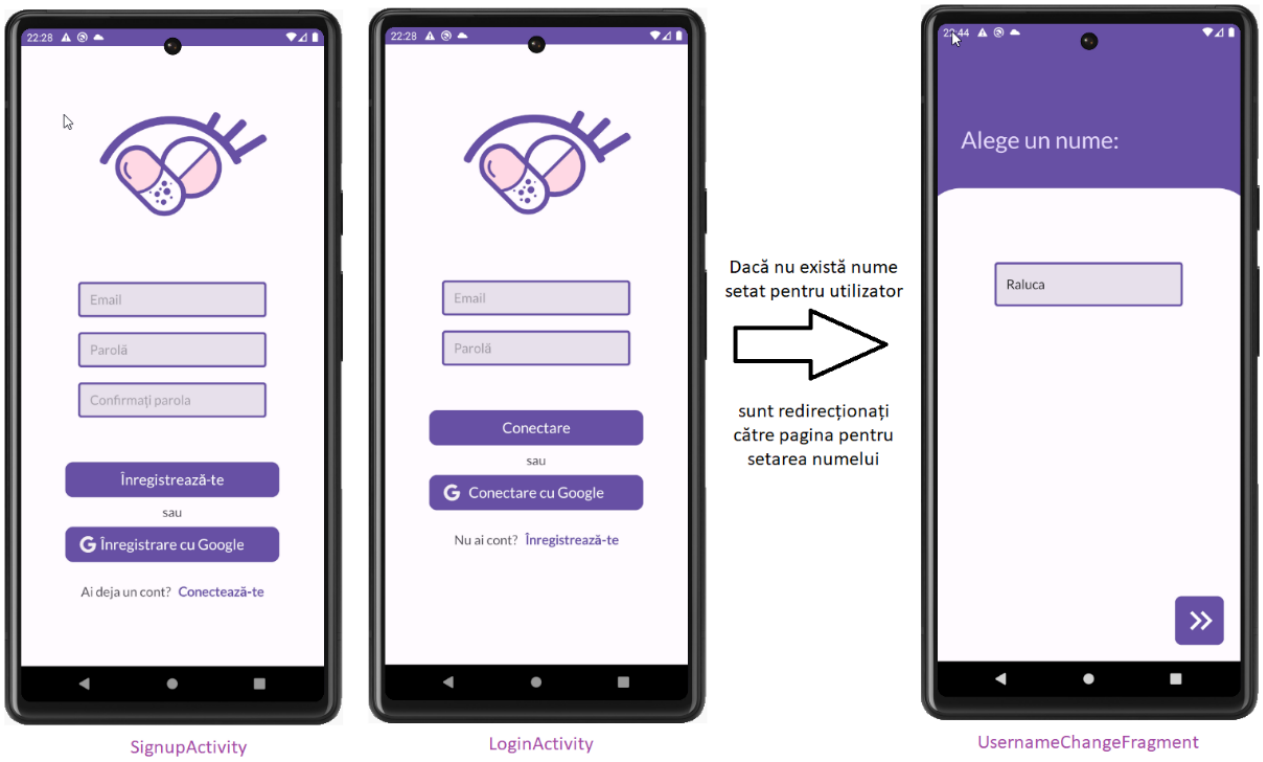


Figura 3.4 *Ecran pentru înregistrare, conectare și setare a numelui utilizatorului*

Între paginile de conectare și înregistrare există o conexiune datorită prezenței unui buton corespunzător în fiecare dintre aceste ecrane. Acest buton permite navigarea ușoară a utilizatorului între cele doua procese de autentificare. Aceste procese pot fi realizate în mod clasic, prin email și parolă, sau prin intermediul contului de Google. Această abordare oferă flexibilitate și comoditate pentru unii utilizatori.

Din punct de vedere al funcționalității, pe lângă procesul esențial de autentificare, aceste două ecrane îndeplinesc și funcția de importare a datelor utilizatorului dacă au fost stocate anterior în Cloud Firestore.

La finalizarea procesului de autentificare, în funcție de rezultatul procesului și de configurările utilizatorului, acesta poate fi redirecționat în mai multe moduri. Când procesul s-a finalizat printr-o eroare, acesta este redirecționat către pagina de întâmpinare anterioară sau un mesaj sugestiv este afișat pe ecran (în cazul parolei greșite, email deja utilizat etc.). Pe de altă parte, daca procesul s-a finalizat prin autentificarea utilizatorului cu succes, dacă utilizatorul nu are configurat un nume personalizat, acesta este redirecționat către pagina specializată pentru setarea numelui. Altfel, acesta este redirecționat către ecranul Splash. Pagina specializată pentru configurarea numelui de utilizator este simplă și intuitivă (Figura 3.4) și configurarea cu succes

rezultă în redirecționarea către SplashScreen. Când user-ul navighează înapoi în ecranul de Splash și este autentificat, numele acestuia va fi afișat în mesajul de bun venit.

3.4 Activitatea principală

Această activitate este cea mai importantă fiind responsabilă pentru toate celelalte fragmente din aplicație. Datorită naturii grafului de navigare, implementarea fragmentelor principale în cadrul acestei activități a fost abordarea cea mai simplă și directă. Fragmentul configurat ca fiind fragmentul principal al aplicației, adică primul fragment prezentat după realizarea navigării către această activitate. În cadrul acestei activități sunt implementate meniul de navigare din partea inferioară a ecranului (BottomNavigation) și meniul de navigare prin intermediul unui sertar lateral (NavigationDrawer). O altă funcție importantă ce se regăsește în această activitate este pornirea funcționării unui Worker, cum a fost menționat în capitolul anterior, pentru a verifica dacă este nevoie de o regenerare a unor alarme. Un aspect distinct al acestei activități constă în prezentarea unor mesaje predefinite care oferă informații utile utilizatorului. Acestea mesaje acoperă o număr restrâns de subiecte care variază de la sfaturi legate de interacțiunile găsite sau negăsite și consultarea unui medic, până la precizarea clară că aplicația nu este autorizată să ofere sfaturi medicale.

Meniul de navigare aflat în partea inferioară a ecranului conține modalități de redirecționare către paginile „Acasă”, „Medicație” și „Setări”. Pe de altă parte, meniul de navigare prin intermediul sertarului lateral, dispune de căi de acces către pagina de „Profil” și pagina de „Setări”, precum și două funcționalități importante: deconectare și verificarea actualizării datelor pentru medicamente. Acest meniu poate fi accesat prin apăsarea numelui utilizatorului sau a pictogramelor adiacente, aflate în partea superioară a ecranului, sau prin glisarea degetului de la stânga la dreapta. În acest meniu, pentru utilizatorii cu rolul de admin, există o funcționalitate în plus, pentru eliberarea datelor despre UserMeds, Alarms și MedsLogs din baza de date. Această funcționalitate a fost folosită în cadrul procesului de testare pentru a verifica ușor implementările realizate. Ambele meniuri, împreună cu numele utilizatorului și pictogramele adiacente, sunt accesibile doar în următoarele fragmente: Home, Medication și Settings.

Dacă în cazul paginilor de autentificare se realizează importarea datelor utilizatorului din Cloud Firestore, în această activitate se realizează o importare a medicamentelor, alarmelor și a istoricului administrării medicamentelor din cloud. Acest proces de importare este efectuat în cadrul acestei activități deoarece pot apărea erori datorită faptului că aceste date se bazează pe medicamentele importate de pe server după prima autentificare în cadrul ecranului Splash.

Astfel, acest proces nu putea fi realizat în celelalte activități.

3.5 Pagina pentru Acasă

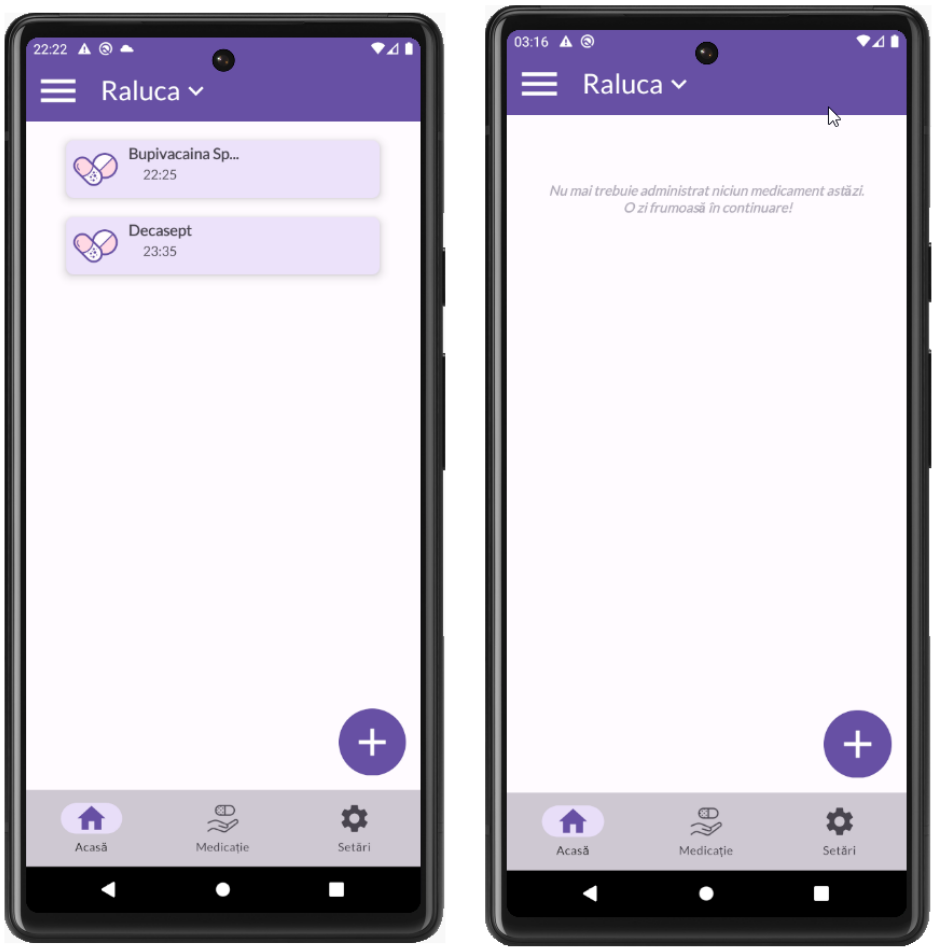


Figura 3.5 *Pagina „Acasă” când lista medicamentelor este goală (dreapta) și când conține elemente (stânga)*

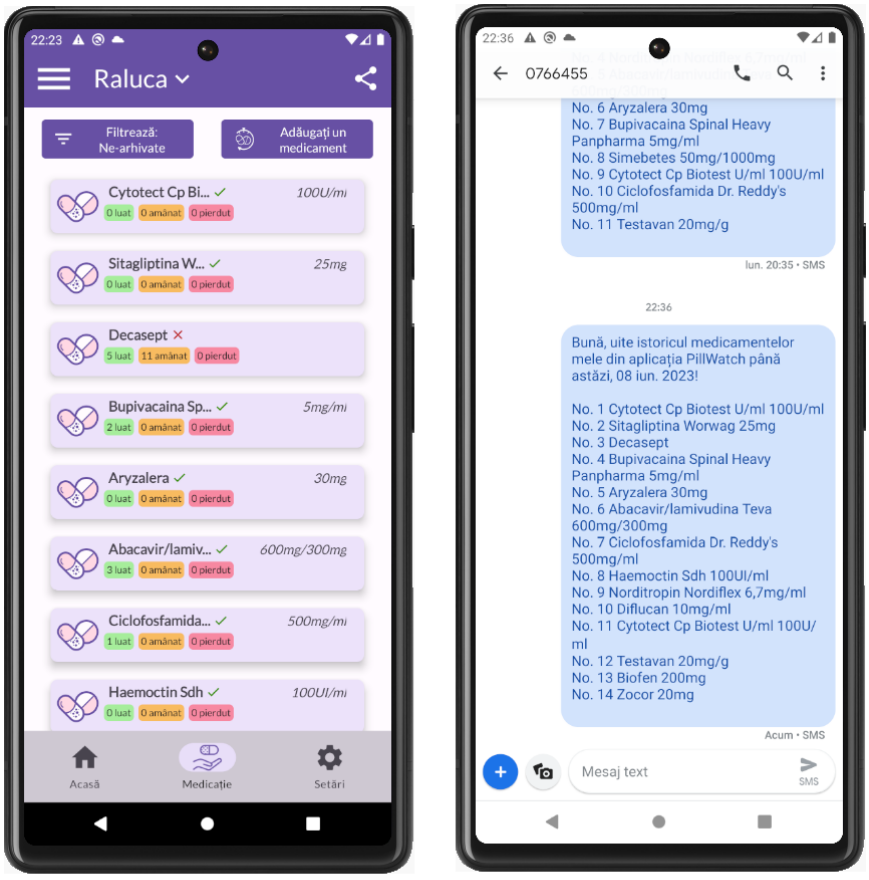
Figura de mai sus (3.5) prezintă interfața simplistă a ecranului „Acasă” în două ipostaze diferite. În prima ipostază, cea din stânga, este ilustrată lista medicamentelor ce trebuie administrate până la miezul nopții, împreună cu ora la care urmează să fie declanșată următoarea alarmă pentru fiecare medicament. Această listă este implementată prin intermediul unui RecyclerView, menționat în capitolul 2. A doua ipostază reprezintă pagina de „Acasă” când nu mai există medicamente ce trebuie administrate în ziua respectivă. Astfel, în centrul ecranului, unde ar trebui sa fie ilustrată lista de medicamente, apare un text sugestiv. Acest fapt este întâlnit în toate cazurile în care este folosit un RecyclerView în aplicația „Pill Watch”.

Pe lângă această listă, există un buton pentru adăugarea unui medicament nou. Mai multe detalii despre procesul de adăugare al unui medicament vor fi oferite în cadrul subcapitolului 3.7, care prezintă toate elementele legate de alarme.

3.6 Pagina listei cu medicamente

Pagina intitulată „Medicație”, conform ilustrației de mai jos (3.6), prezintă o listă detaliată a tuturor medicamentelor utilizatorului. Această listă oferă posibilitatea de a filtra medicamentele în funcție de statusul lor de arhivare (arhivat, ne-arhivat, sau toate medicamentele). Prin apăsarea butonului de filtrare, situat în partea superioară stângă, setările de filtrare sunt ajustate în conformitate cu textul afișat. Similar paginii „Acasă”, această pagină include și un buton de adăugare a unui medicament nou.

Lista de medicamente prezintă mai multe elemente cheie, caracteristice fiecărui medicament. Aceste elemente includ o pictogramă sugestivă pentru medicament, numele medicamentului, o pictogramă sugestivă pentru statusul verificării interacțiunilor (poate fi interpretată și ca un mod de distingere între medicamentele personalizate și cele din lista medicamentelor aprobate), concentrația medicamentului și trei etichete corespunzătoare numărului de administrări a acestuia, în funcție de statusul administrărilor (luat, amânat sau pierdut). Etichetele menționate sunt colorate sugestiv, bazându-se pe statusul administrării, luat – verde, amânat – galben, pierdut – roșu.



MedicationFragment

Opțiunea de distribuire a istoricului medicamentelor

Figura 3.6 *Ecranul pentru „Medicație” cu lista medicamentelor și exemplul mesajului de distribuire al istoricului medicamentelor*

Partea dreaptă a figurii 3.6 exemplifică o funcție implementată în cadrul acestui fragment, și anume, distribuirea listei de medicamente administrate către alte aplicații. Această funcție poate fi accesată prin intermediul unui buton sugestiv, situat în colțul superior drept al ecranului. Utilizarea acestei funcții furnizează informații relevante despre aplicația de proveniență a mesajului (numele aplicației), data înregistrării datelor respective (ziua în care s-a apăsat butonul) și numele medicamentului, împreună cu un index de numărare asociat.

3.7 Adăugarea unui medicament și configurarea alarmelor

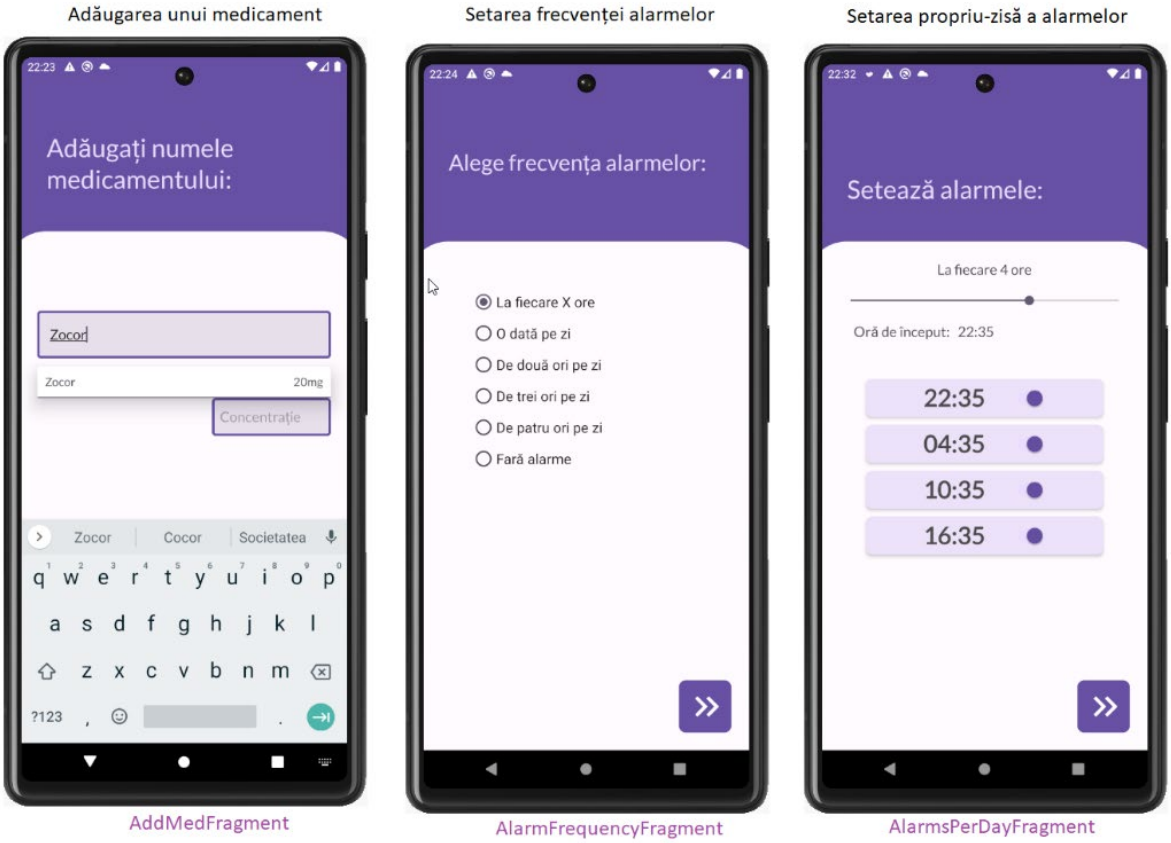


Figura 3.7 *Ecrane pentru adăugarea unui medicament, setarea frecvenței alarmelor și setarea alarmelor*

În figura 3.7 sunt ilustrate trei ecrane distincte care împreună alcătuiesc procesul de adăugare al unui medicament nou. Totodată, cele două ecrane din partea dreaptă a figurii reprezintă și procesul de regenerare al alarmelor care este declanșat din cadrul pagini unui medicament.

Ecranul din stânga reprezintă procesul de adăugare propriu-zisă a unui medicament prin introducerea numelui medicamentului în câmpul aflat în mijlocul ecranului. Acest proces funcționează pe baza unui component numit AutoCompleteTextView, menționat în capitolul 2.

Prin intermediul acestui component, este realizată predicția medicamentului căutat.

Acest ecran prezintă încă o funcție importantă prin intermediul căreia se verifică interacțiunile chimice dintre medicamentul introdus și celelalte medicamente din contul utilizatorului. Această verificare este declanșată, pe baza unui criteriu, în momentul în care este apăsat butonul pentru continuarea procesului de adăugare a medicamentului. Acest criteriu constă în modul de introducere al medicamentului. Mai precis, dacă acesta a fost introdus manual sau a fost selectat din meniul predicțiilor. Verificarea este declanșată doar pentru medicamentele selectate din acest meniu, deoarece acestea dețin în baza de date un cod folosit pentru găsirea acestora în baza de date internațională de la NIH [10]. În momentul în care se efectuează această verificare, pe ecranul utilizatorului este afișat o fereastră cu text care sugerează faptul că se realizează verificarea interacțiunilor. Aceasta constă în trimiterea către server a codurilor menționate anterior, în vederea apelării API-urilor de la NIH [10] pentru găsirea unei interacțiuni. Pe baza existenței unei interacțiuni există două scenarii posibile. Când nu sunt găsite interacțiuni se realizează navigarea către următorul fragment. Altfel, este afișată o fereastră care furnizează informații despre severitatea interacțiunii găsite (dacă este slabă, moderată sau ridicată) și sunt oferite 2 opțiuni: continuarea procesului de adăugare sau anularea acestuia. În cadrul figurii de mai jos (3.8) este ilustrată o astfel de alertă, declanșată datorită interacțiunii dintre medicamentele Zocor și Diflucan [30]. În funcție de opțiunea aleasă, se realizează, sau nu, navigarea către următorul fragment.

Navigarea către fragmentul pentru configurarea frecvenței administrării unui medicament când nu s-au găsit interacțiuni sau când nu s-a putut efectua verificarea (datorită faptului că medicamentul a fost introdus manual) implică declanșarea unor mesaje de avertizare a utilizatorului cu mesaje care sugerează că pot exista totuși interacțiuni între medicamente.

În partea centrală a figurii 3.7 este ilustrat ecranul pentru setarea frecvenței alarmelor și cuprinde șase opțiuni posibile. Acestea sunt reprezentate de o opțiune de a seta dinamic la fiecare X număr de ore, una pentru a nu configura alarme și opțiuni pentru o dată pe zi, respectiv de două, trei sau patru ori pe zi. Opțiunea pentru a nu configura alarme este finalizată cu redirecționarea utilizatorului către pagina medicamentului creat. Celelalte opțiuni declanșează navigarea către fragmentul pentru setarea alarmelor.

Partea dreaptă a figurii 3.7 reprezintă ecranul pentru setarea alarmelor când este selectată opțiunea dinamică a frecvenței pentru alarme la fiecare X ore. Acest ecran prezintă un slider ce poate avea valorile divizorilor lui 24 (excluzând 24) și reprezintă intervalul pentru administrarea medicamentelor. Acest slider este absent când sunt alese celelalte opțiuni de frecvență. Pe baza valorii selectate prin intermediul slider-ului, numărul de alarme generat variază pentru a

corespunde intervalului configurat. De asemenea, ecranul dispune de o opțiune pentru configurarea orei de început a administrării medicamentului. Pe lângă aceste funcții, alarmele generate în partea inferioară a ecranului pot fi dezactivate, reactivate și chiar modificate. După modificarea unei alarme, lista alarmelor afișate este modificată pentru a fi ordonate în ordinea în care vor fi declanșate. Generarea alarmelor este realizată prin intermediul clasei AlarmGenerator.

Apăsarea butonului pentru continuare implică salvarea alarmelor în baza de date și pe cloud, împreună cu programarea acestora prin intermediul unei clase personalizate numită AlarmHandler, iar utilizatorul este redirecționat către pagina medicamentului. Această clasă este folosită pentru programarea, anularea sau chiar reprogramarea alarmelor (amânate), împreună cu înregistrarea interacțiunii cu notificarea alarmei (statusul de luat, amânat, pierdut).

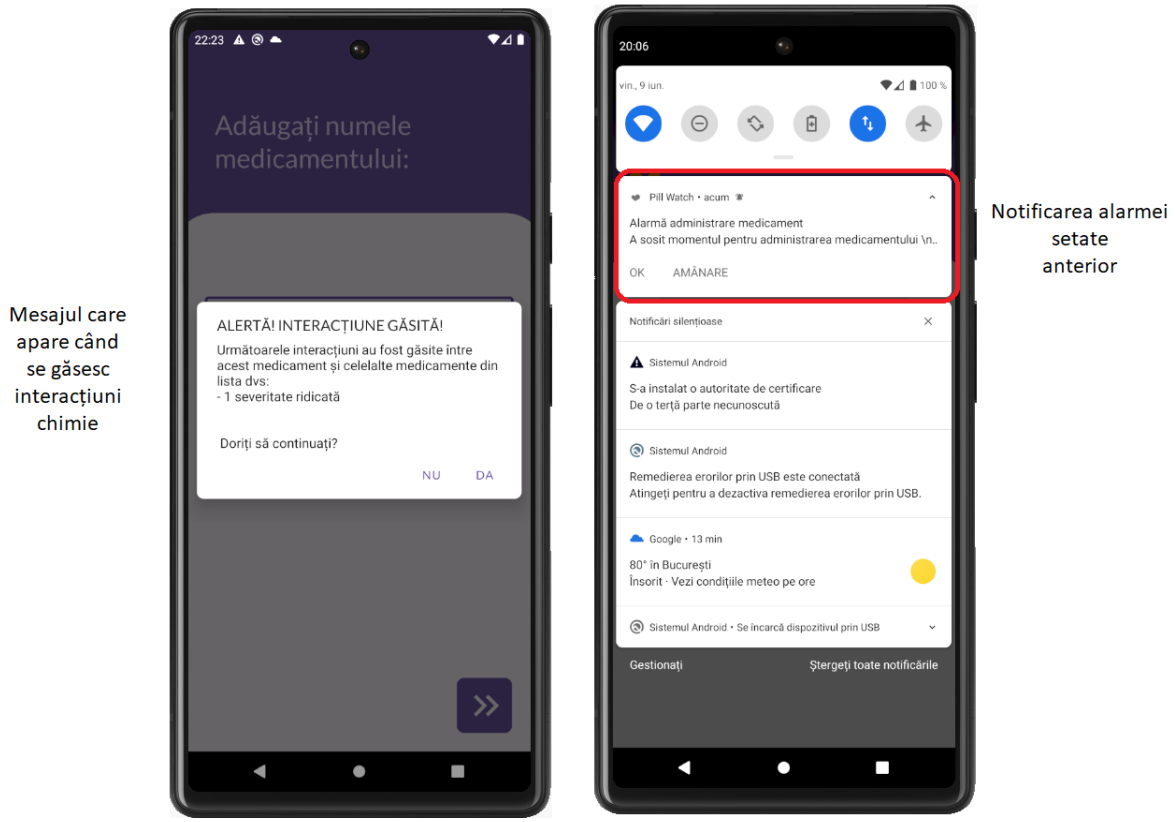


Figura 3.8 *Exemple pentru alerta interacțiunii și notificarea unei alarme*

Declanșarea alarmelor implică nevoia afișării unei notificări personalizate. Această notificare este afișată prin intermediul clasei AlarmReceiver. Această clasă înregistrează id-ul alarmei, împreună cu scopul utilizării acestuia și programează afișarea notificării. Scopul menționat poate să nu fie afișarea notificării, ci înregistrarea statusului administrării. Un exemplu pentru notificarea alarmei în privința administrării unui medicament este ilustrat în

figura 3.8 și implică 2 opțiuni posibile: apăsarea butonului „OK” sau „AMÂNARE”. AlarmReceiver interpretează opțiunea aleasă și apelează AlarmHandler pentru a înregistra statusul administrării medicamentului în funcție de răspuns. Statusul de „pierdut” este înregistrat în momentul în care utilizatorul nu a interacționat cu notificarea afișată în timpul prestabilit de o oră.

Un aspect important de menționat este faptul că în momentul declanșării unei alarme, se verifică automat dacă alarma curentă este ultima alarmă înregistrată pentru medicamentul respectiv. În urma acestei verificări, dacă alarma este cu adevărat ultima, este declanșat un proces de regenerare a alarmelor conform frecvenței stabilite. Ora de început a acestui set de alarme este reprezentată de ora calculată pe baza alarmei curente, la care se adaugă o valoare corespunzătoare, astfel încât se respectă frecvența administrării.

Pe lângă notificarea clasică ce apare în momentul declanșării alarmei, prin intermediul clasei AlarmService, este implementată activarea sunetului prestabilit al telefonului pentru alarme împreună cu utilizarea unui model de vibrații dinamic. Acest model de vibrații constă în scăderea perioadei dintre vibrații în funcție de numărul de amânări ale unei alarme, simulând astfel o alarmă mai insistentă. Conform codului prezentat mai jos, când alarma este amânată de 2 sau mai multe ori, modelul de alarma va fi: vibrează timp de 800 de milisecunde, apoi va exista o pauză de 200 milisecunde. Similar se realizează și celelalte două situații, vibrațiile devenind mai frecvente și mai puțin ignorabile în proporție cu numărul amânărilor.

```
val vibrationPattern = when {
        postponedTimes > 2 -> longArrayOf(0, 800, 200)
        postponedTimes > 1 -> longArrayOf(0, 1000, 500)
        else -> longArrayOf(0, 1500, 1000)
    }
```

3.8 Pagina unui medicament

Figura 3.9 de mai jos reprezintă ilustrarea paginii unui medicament unde pot fi observate pictogramele sugestive pentru funcțiile de ștergere, arhivare, editare și vizualizare a istoricului administrării acestuia, pictograma corespunzătoare statusului de verificare, numele medicamentului, butonul pentru regenerarea alarmelor și lista alarmelor stabilite.

În afară de funcțiile intuitive (ștergere, editare, arhivare), există funcția de vizualizare a istoricului administrării medicamentelor. Această funcție afișează o fereastră în care este prezentată lista administrărilor, prin intermediul unui RecyclerView. Modul de afișare arată ora, ziua din săptămână și data la care a fost înregistrat acea informație, împreună cu o etichetă corespunzătoare statusului administrării (luat, amânat, pierdut). O altă funcție notabilă a acestei

ferestre reprezintă funcția de distribuire a celor mai recente 10 înregistrări de administrare prin intermediul altor aplicații. În cadrul figurii 3.9, este redat un exemplu pentru această funcție de distribuire.

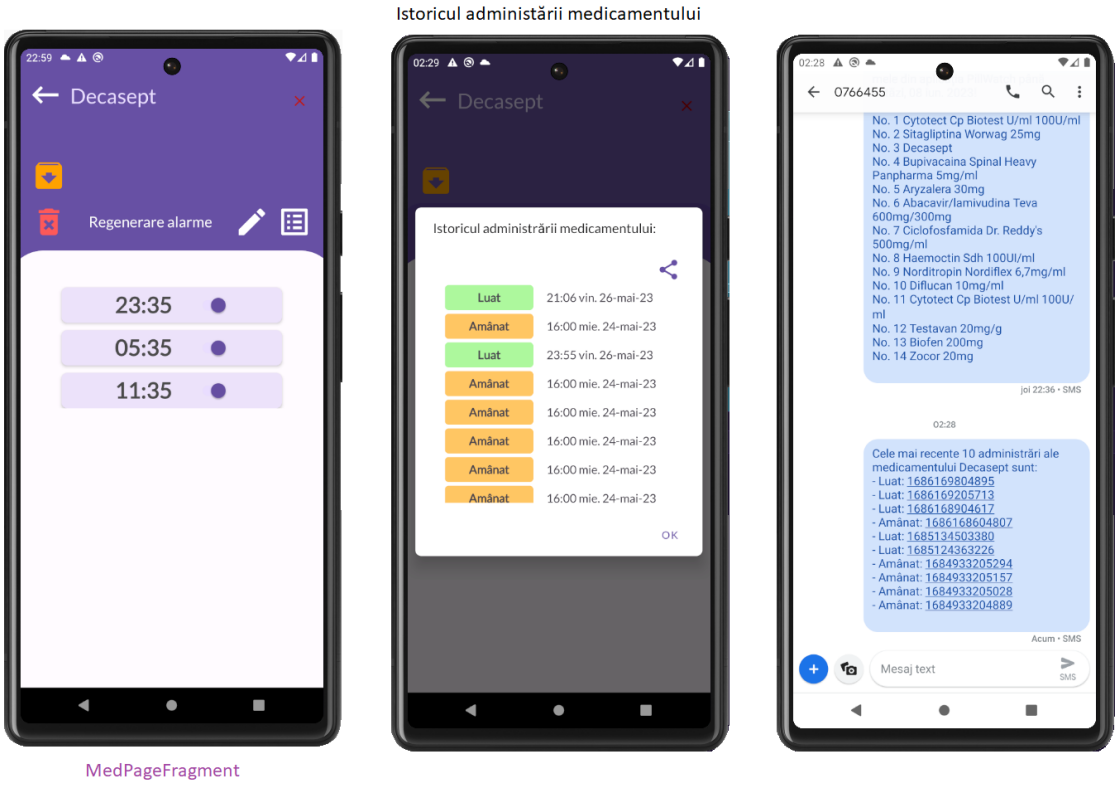


Figura 3.9 *Pagina unui medicament, fereastra corespunzătoare istoricului administrării medicamentului și o exemplificare a funcției de distribuie din cadrul acestei ferestre*

3.9 Pagina pentru setări

Pagina de setări prezintă trei opțiuni pentru utilizator, acestea fiind: schimbarea temei aplicației, personalizarea mesajului corespunzător notificării alarmei și un buton pentru deconectare. În figura de mai jos (3.10) este reprezentat ecranul corespunzător paginii pentru setări, împreună cu exemplificări ale setărilor disponibile.

Setarea temei aplicației dispune la rândul său de 3 opțiuni distincte: luminoasă (fundal alb), întunecată (fundal gri) și setarea implicită (potrivit setărilor telefonului). În urma schimbării temei, pentru ca modificările să fie implementate, este necesară repornirea aplicației. Această repornire poate fi realizată manual de către utilizator, sau forțat de către aplicație în momentul respectiv. Pentru o experiență mai bună a utilizatorului, în momentul în care se alege o temă, va fi afișată o fereastră pentru a informa utilizatorul de această repornire și îi oferă acestuia alegerea între opțiunile de repornire manuală sau forțată.

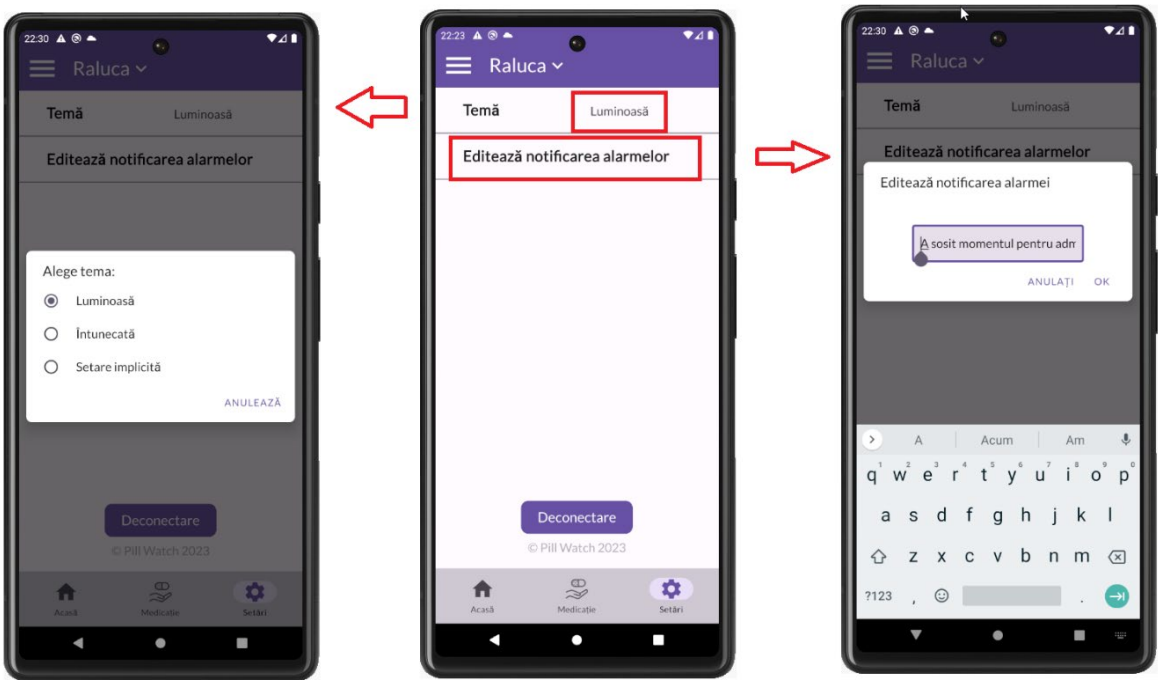


Figura 3.10 *Pagina corespunzătoare pentru pagina „Setări” și imagini cu 2 ferestre corespunzătoare setărilor disponibile*

Celelalte două opțiuni disponibile în cadrul acestui ecran sunt mai simple, opțiunea pentru deconectare fiind evidentă. Opțiunea pentru editare a mesajului notificării alarmelor declanșează afișarea unei ferestre în cadrul căreia este introdus automat mesajul curent. Această fereastră oferă posibilitatea salvării modificărilor realizate, sau anularea procesului.

Figura 3.11 *Ecran corespunzător paginii „Profil”*

3.10 Pagina de profil

Figura 3.11 ilustrează pagina de profil a unui utilizator ce dispune de o interfață simplă și intuitivă. Pe această pagină, utilizatorul poate verifica numărul total de medicamente active (excluzând medicamentele arhivate), adresa de email folosită la înregistrare și numele asociat contului. Acest nume beneficiază și de opțiunea de editare, accesibilă prin apăsarea unei pictograme sugestive. Acest ecran are rol pur informativ și permite personalizarea contului după preferințe.

4 Concluzii și perspective pentru aplicația „Pill Watch”

4.1 Funcțiile aplicației „Pill Watch”

Prin intermediul acestei lucrări, am prezentat în detaliu importanța, contextul, procesul de dezvoltare și tehnologiile utilizate care au contribuit la crearea aplicației „Pill Watch”. Aceasta este dedicată dispozitivelor de tip Android și are rolul de a ajuta utilizatorii să respecte tratamentele medicamentoase prin reamintirea administrării acestora.

Un aspect cheie al aplicației constă în posibilitatea adăugării unor medicamente cu posibilitatea realizării unei verificări pentru existența unor interacțiuni chimice cu celelalte medicamente. O altă funcție importantă disponibilă unui utilizator este capacitatea de a configura alarme pentru administrarea medicamentelor, cu opțiuni de personalizare a acestora. Alarma este reprezentată printr-o notificare, sunetul implicit al telefonului pentru alarme, și un model de vibrații dinamic. Statusul administrărilor este înregistrat pe baza interacțiunii utilizatorului cu notificarea alarmei. Mai multe statusuri alcătuiesc un istoric ce poate fi distribuit prin intermediul altor aplicații. De asemenea, lista medicamentelor din contul unui utilizator poate fi distribuită în același mod.

4.2 Obiective de viitor și provocări întâmpinate

Unul dintre obiectivele principale pentru viitorul aplicației constă în lansarea pe piața din România, ulterior pe plan internațional. De asemenea, se urmărește dezvoltarea unei interfețe care permite o comunicare eficientă și intuitivă între doctor, pacient și farmacist. Această interfață va include posibilitatea ca un doctor să trimită un medicament, sub formă de cod QR către un utilizator, cu parametrii prestabiliți pentru nume, concentrație, frecvență etc. Codul QR va fi scanat, fie online, fie fizic, de către un farmacist în scopul achiziționării medicamentului, respectiv de către pacient. Ulterior va fi necesară doar configurarea orei de începere a tratamentului și respectarea acestuia. Acest mecanism de comunicare are scopul de a minimiza erorile umane realizate prin transmiterea incorectă a numele unui medicament către un farmacist sau înlocuirea rețetelor pe hârtie în viitor.

Dezvoltarea acestei aplicații s-a dovedit a fi o provocare datorită faptului că nu am avut experiență anterioară cu dezvoltarea unei aplicații Android și nu am mai creat până acum o

aplicație completă, de obicei fiind cerut fie frontend, fie backend pentru realizarea proiectelor pentru facultate. Procesul de dezvoltare a fost complet nou, având, iar pentru că a utiliza limbajul de programare Kotlin pentru prima dată, a fost necesară o perioadă de adaptare, deși aveam deja experiență cu Java. Am întâmpinat provocări și în implementarea server-ului, datorită faptului că nu am interacționat foarte mult cu Node.JS și JavaScript în cei 3 ani de facultate. În plus, accesarea unui server local dintr-o aplicație Android s-a dovedit a fi dificilă, fiind necesară implementarea unor certificate autorizate sau ajustarea permisiunilor aplicației pentru a permite apelarea paginilor HTTP (fără SSH).

În ansamblu, în cadrul realizării acestei lucrări, respectiv aplicații și server, am dobândit cunoștințe în dezvoltarea aplicațiilor Android, utilizarea limbajelor Kotlin, JavaScript și Node.JS, manipularea și accesarea API-urilor, utilizarea serviciilor Firebase, găzduirea unei aplicații într-o platformă cloud și crearea unei aplicații de la zero.

5 Bibliografie

[1] Rogers, T., & MilkMan, K. L., Reminders Through Association, Mai 2016. Preluat de pe Sage Journals: https://journals.sagepub.com/doi/abs/10.1177/0956797616643071 Accesat ultima dată pe 31 Mai 2023.

[2] Flynn, J., 20 Vital SMARTPHONE USAGE STATISTICS [2023]: FACTS, DATA, AND TRENDS ON MOBILE USE IN THE U.S, Aprilie 2023. Preluat de pe Zippia: https://www.zippia.com/advice/smartphone-usage-statistics/ Accesat ultima dată pe 30 Mai 2023.

[3] Yoon CH, N. I., Long-Term Impact of a Smartphone App on Prescriber Adherence to Antibiotic Guidelines for Adult Patients With Community-Acquired Pneumonia: Interrupted Time-Series Study. Mai 2023. Preluat de pe JMIR Publications: https://www.jmir.org/2023/1/e42978 Accesat ultima dată pe 30 Mai 2023.

[4] Google. Google Play, 2023, Preluat de pe Google Play: https://play.google.com/store/apps Accesat ultima dată pe 28 Februarie 2023.

[5] MyTherapy, MyTherapy Pill Reminder, 2023, Preluat de pe Google Play: https://play.google.com/store/apps/details?id=eu.smartpatient.mytherapy Accesat ultima dată pe 20 Februarie 2023.

[6] Wachanga, Pill Log: Medication Reminder, 2022. Preluat de pe Google Play: https://play.google.com/store/apps/details?id=com.wachanga.pills Accesat ultima dată pe 22 Februarie 2023.

[7] caiocrol, Alarm and pill reminder, 2023. Preluat de pe Google Play: https://play.google.com/store/apps/details?id=br.com.caiocrol.alarmandpillreminder Accesat ultima dată pe 22 Februarie 2023.

[8] Medisafe, Medisafe Pill & Med Reminder, 2023. Preluat de pe Google Play: https://play.google.com/store/apps/details?id=com.medisafe.android.client&hl=en Accesat ultima dată pe 28 Februarie 2023.

[9] ANM. Lista medicamentelor din NOMENCLATOR, 2023. Preluat de pe Agenția Națională a Medicamentului și a Dispozitivelor Medicale din România: https://nomenclator.anm.ro/medicamente?page=1 Accesat ultima dată pe 20 Martie 2023.

[10] NLM, APIs, 2023. Preluat de pe National Library of Medicine: https://lhncbc.nlm.nih.gov/RxNav/APIs/index.html Accesat ultima dată pe 25 Martie

2023.

[11] Google, Guide to app architecture, Martie 2023. Preluat de pe Developers Android: https://developer.android.com/topic/architecture#separation-of-concerns Accesat ultima dată pe 25 Martie 2023.

[12] Dumbravan, A, Clean Android Architecture: Take a layered approach to writing clean, testable, and decoupled Android applications, June 2022. Publicat de Packt Publishing, pagina 366.

[13] Google, Dagger, 2023. Preluat de pe Dagger: https://dagger.dev/ Accesat ultima dată pe 4 Iunie 2023.

[14] Documentația oficială Firebase, Learn about Build products, 2023. Preluat de pe Firebase: https://firebase.google.com/docs/build Accesat ultima dată pe 2 Iunie 2023.

[15] Documentația oficială Firebase, Cloud Firestore, 2023. Preluat de pe Firebase: https://firebase.google.com/docs/firestore Accesat ultima dată pe 3 Iunie 2023.

[16] Documentația oficială Firebase, Firebase Cloud Messaging, 2023. Preluat de pe Firebase: https://firebase.google.com/docs/cloud-messaging Accesat ultima dată pe 3 Iunie 2023.

[17] Documentația oficială Firebase, Firebase Authentication, Iunie 2023. Preluat de pe Firebase: https://firebase.google.com/docs/auth Accesat ultima dată pe 4 Iunie 2023.

[18] Documentația oficială Android, Save data in a local database using Room. Preluat de pe developers.android: https://developer.android.com/training/data-storage/room Accesat ultima dată pe 2 Iunie 2023.

[19] Documentația oficială Android, Android Jetpack. Preluat de pe developers.android: https://developer.android.com/jetpack Accesat ultima dată pe 1 Iunie 2023.

[20] Square, retrofit. Preluat de pe https://square.github.io/retrofit/ Accesat ultima dată pe 4 Iunie 2023.

[21] Square, moshi, Mai 2023. Preluat de pe GitHub: https://github.com/square/moshi#kotlin-codegen Accesat ultima dată pe 6 Iunie 2023.

[22] Documentația oficială Android, AutoCompleteTextView, Iunie 2023. Preluat de pe developers.android: https://developer.android.com/reference/android/widget/AutoCompleteTextView Accesat ultima dată pe 8 Iunie 2023.

[23 Documentația oficială Android, Create dynamic lists with RecyclerView, Iunie 2023. Preluat de pe developers.android: https://developer.android.com/develop/ui/views/layout/recyclerview Accesat ultima dată pe 4 Iunie 2023.

[24] Documentația oficială Android, Kotlin coroutines on Android, Martie 2023. Preluat de pe developers.android: https://developer.android.com/kotlin/coroutines Accesat ultima dată pe 4 Iunie 2023.

[25] Documentația oficială Android, AlarmManager, Iunie 2023. Preluat de pe developers.android: https://developer.android.com/reference/android/app/AlarmManager Accesat ultima dată pe 9 Iunie 2023.

[26] Documentația oficială Android, Schedule tasks with WorkManager, Martie 2023. Preluat de pe developers.android: https://developer.android.com/topic/libraries/architecture/workmanager Accesat ultima dată pe 5 Iunie 2023.

[27] Documentația oficială ExpressJS, Express Glossary, 2023. Preluat de pe Expressjs: https://expressjs.com/en/resources/glossary.html Accesat ultima dată pe 2 Iunie 2023.

[28] ZoracKy, D, convert-excel-to-json, Februarie 2018. Preluat de pe GitHub: https://github.com/DiegoZoracKy/convert-excel-to-json/ Accesat ultima dată pe 2 Iunie 2023.

[29] Axios, Aprilie 2023. Preluat de pe GitHub: https://github.com/axios/axios#features Accesat ultima dată pe 5 Iunie 2023.

[30] National Library of Medicine. Drug Interaction API. Preluat de pe National Library of Medicine: https://lhncbc.nlm.nih.gov/RxNav/APIs/api-Interaction.findInteractionsFromList.html Accesat ultima dată pe 5 Aprilie 2023.

[31] Leonidas-from-XIV, node-xml2js, Mai 2023. Preluat de pe GitHub: https://github.com/Leonidas-from-XIV/node-xml2js Accesat ultima dată pe 4 Iunie 2023.

[32] Documentația oficială Node.js, FileSystem. Preluat de pe Node.js: https://nodejs.org/api/fs.html Accesat ultima dată pe 5 Iunie 2023.

[33] Documentația oficială Node.js, Crypto. Preluat de pe Node.js: https://nodejs.org/api/crypto.html Accesat ultima dată pe 4 Iunie 2023.

[34] node-cron, Mai 2021. Preluat de pe GitHub: https://github.com/node-cron/node-cron Accesat ultima dată pe 5 Iunie 2023.

[35] taoqf, node-html-parser, Februarie 2023. Preluat de pe GitHub: https://github.com/taoqf/node-html-parser Accesat ultima dată pe 4 Iunie 2023.

[36] Server-ul găzduit pe platforma Heroku: https://pill-watch-server.herokuapp.com/ Accesat ultima dată pe 12 iunie 2023