



UNIVERSITATEA DIN CRAIOVA  
FACULTATEA DE AUTOMATICĂ, CALCULATOARE ȘI  
ELECTRONICĂ  
DEPARTAMENTUL DE CALCULATOARE ȘI TEHNOLOGIA  
INFORMAȚIEI



## PROIECT DE DIPLOMĂ

Gabriela-Andreea PĂTRU

COORDONATOR ȘTIINȚIFIC

Prof.univ.dr.ing. Eugen GANEA

SEPTEMBRIE 2023

CRAIOVA



UNIVERSITATEA DIN CRAIOVA  
FACULTATEA DE AUTOMATICĂ, CALCULATOARE ȘI  
ELECTRONICĂ  
DEPARTAMENTUL DE CALCULATOARE ȘI TEHNOLOGIA  
INFORMAȚIEI



Aplicație mobilă (Android/IOS) pentru gestionarea notificărilor transmise de  
către portalul facultății

Gabriela-Andreea PĂTRU

COORDONATOR ȘTIINȚIFIC

Prof.univ.dr.ing. Eugen GANEA

SEPTEMBRIE 2023

CRAIOVA

*„Învățătura este o comoară care își urmează stăpânul pretutindeni.”*

Proverb popular

## DECLARAȚIE DE ORIGINALITATE

Subsemnatul GABRIELA-ANDREEA PĂTRU, student la specializarea CALCULATOARE din cadrul Facultății de Automatică, Calculatoare și Electronică a Universității din Craiova, certific prin prezenta că am luat la cunoștință de cele prezentate mai jos și că îmi asum, în acest context, originalitatea proiectului meu de licență:

- cu titlul Aplicație mobilă (Android/IOS) pentru gestionarea notificărilor transmise de către portalul facultății,
- coordonată de Prof.univ.dr.ing. EUGEN GANEA,
- prezentată în sesiunea SEPTEMBRIE 2023

La elaborarea proiectului de licență, se consideră plagiat una dintre următoarele acțiuni:

- reproducerea exactă a cuvintelor unui alt autor, dintr-o altă lucrare, în limba română sau prin traducere dintr-o altă limbă, dacă se omit ghilimele și referința precisă,
- redarea cu alte cuvinte, reformularea prin cuvinte proprii sau rezumarea ideilor din alte lucrări, dacă nu se indică sursa bibliografică,
- prezentarea unor date experimentale obținute sau a unor aplicații realizate de alți autori fără menționarea corectă a acestor surse,
- însușirea totală sau parțială a unei lucrări în care regulile de mai sus sunt respectate, dar care are alt autor.

Pentru evitarea acestor situații neplăcute se recomandă:

- plasarea între ghilimele a citatelor directe și indicarea referinței într-o listă corespunzătoare la sfârșitul lucrării,
- indicarea în text a reformulării unei idei, opinii sau teorii și corespunzător în lista de referințe a sursei originale de la care s-a făcut preluarea,
- precizarea sursei de la care s-au preluat date experimentale, descrieri tehnice, figuri, imagini, statistici, tabele et caetera,
- precizarea referințelor poate fi omisă dacă se folosesc informații sau teorii arhicunoscute, a căror paternitate este unanim cunoscută și acceptată.

Data,

06.09.2023

Semnătura candidatului,








UNIVERSITATEA DIN CRAIOVA  
Facultatea de Automatică, Calculatoare și Electronică  
Departamentul de Calculatoare și Tehnologia Informației

Aprobat la data de .....  
Șef de departament,  
Prof. dr. ing.  
**Nicolae Enescu/  
Comin IONETE/  
Dorian COJOCARU**

## PROIECTUL DE DIPLOMĂ

Numele și prenumele studentului/-ei:	Pătru Gabriela-Andreea
Enunțul temei:	Aplicație mobilă (Android/iOS) pentru gestionarea notificărilor transmise de către portalul facultății
Datele de pornire:	Dezvoltarea unei aplicații mobile (Android/iOS) pentru gestiunea notificărilor transmise de către portalului facultății; notificările se pot referi la actualizarea orarului, oportunități de practică/internship
Conținutul proiectului:	<ol style="list-style-type: none"><li>1. Introducere</li><li>2. Concepte teoretice</li><li>3. Despre aplicație</li><li>4. Concluzii</li></ol>
Material grafic obligatoriu:	Power Point care conține cod, capturi de ecran, diagrame, explicații
Consultații:	Periodice
Conducătorul științific (titlul, nume și prenume, semnătura):	Prof.univ.dr.ing. Eugen Ganea
Data eliberării temei:	15.10.2022
Termenul estimat de predare a proiectului:	06.09.2023
Data predării proiectului de către student și semnătura acestuia:	





UNIVERSITATEA DIN CRAIOVA  
Facultatea de Automatică, Calculatoare și Electronică  
Departamentul de Calculatoare și Tehnologia Informației

## REFERATUL CONDUCĂTORULUI ȘTIINȚIFIC

Numele și prenumele candidatului/-ei: Pătru Gabriela-Andreea  
Specializarea: Calculatoare și Tehnologia Informației  
Titlul proiectului: Aplicație mobilă (Android/IOS) pentru gestionarea notificărilor transmise de către portalul facultății  
Locația în care s-a realizat practica de documentare (se bifează una sau mai multe din opțiunile din dreapta):  
În facultate ☐\*  
În producție ☐  
În cercetare ☐  
Altă locație: *[se detaliază]*

În urma analizei lucrării candidatului au fost constatate următoarele:

Nivelul documentării		Insuficient <input type="checkbox"/>	Satisfăcător <input type="checkbox"/>	Bine <input type="checkbox"/>	Foarte bine <input type="checkbox"/>
Tipul proiectului		Cercetare <input type="checkbox"/>	Proiectare <input type="checkbox"/>	Realizare practică <input type="checkbox"/>	Altul <i>[se detaliază]</i>
Aparatul matematic utilizat		Simplu <input type="checkbox"/>	Mediu <input type="checkbox"/>	Complex <input type="checkbox"/>	Absent <input type="checkbox"/>
Utilitate		Contract de cercetare <input type="checkbox"/>	Cercetare internă <input type="checkbox"/>	Utilare <input type="checkbox"/>	Altul <i>[se detaliază]</i>
Redactarea lucrării		Insuficient <input type="checkbox"/>	Satisfăcător <input type="checkbox"/>	Bine <input type="checkbox"/>	Foarte bine <input type="checkbox"/>
Partea grafică, desene		Insuficientă <input type="checkbox"/>	Satisfăcătoare <input type="checkbox"/>	Bună <input type="checkbox"/>	Foarte bună <input type="checkbox"/>
Realizarea practică	Contribuția autorului	Insuficientă <input type="checkbox"/>	Satisfăcătoare <input type="checkbox"/>	Mare <input type="checkbox"/>	Foarte mare <input type="checkbox"/>
	Complexitatea temei	Simplă <input type="checkbox"/>	Medie <input type="checkbox"/>	Mare <input type="checkbox"/>	Complexă <input type="checkbox"/>
	Analiza cerințelor	Insuficient <input type="checkbox"/>	Satisfăcător <input type="checkbox"/>	Bine <input type="checkbox"/>	Foarte bine <input type="checkbox"/>
	Arhitectura	Simplă <input type="checkbox"/>	Medie <input type="checkbox"/>	Mare <input type="checkbox"/>	Complexă <input type="checkbox"/>



	Întocmirea specificațiilor funcționale	Insuficientă <input type="checkbox"/>	Satisfăcătoare <input type="checkbox"/>	Bună <input type="checkbox"/>	Foarte bună <input type="checkbox"/>
	Implementarea	Insuficientă <input type="checkbox"/>	Satisfăcătoare <input type="checkbox"/>	Bună <input type="checkbox"/>	Foarte bună <input type="checkbox"/>
	Testarea	Insuficientă <input type="checkbox"/>	Satisfăcătoare <input type="checkbox"/>	Bună <input type="checkbox"/>	Foarte bună <input type="checkbox"/>
	Funcționarea	Da <input type="checkbox"/>	Parțială <input type="checkbox"/>	Nu <input type="checkbox"/>	
Rezultate experimentale		Experiment propriu <input type="checkbox"/>		Preluare din bibliografie <input type="checkbox"/>	
Bibliografie		Cărți	Reviste	Articole	Referințe web
Comentarii și observații					

În concluzie, se propune:

ADMITEREA PROIECTULUI <input type="checkbox"/>	RESPINGEREA PROIECTULUI <input type="checkbox"/>
---	---

Data,

Semnătura conducătorului științific,

## REZUMATUL PROIECTULUI

Am dezvoltat o aplicație mobilă pentru platforma Android, folosind Android Studio și limbajul Java. Scopul principal al acestei aplicații este să ofere utilizatorilor o experiență eficientă în gestionarea notificărilor primite de la portalul facultății lor, cu ajutorul serviciului OneSignal.

Am ales Android Studio și limbajul Java pentru proiectul meu deoarece sunt o combinație puternică și bine stabilită în dezvoltarea de aplicații mobile Android. Android Studio oferă un mediu de dezvoltare integrat robust, cu instrumente avansate de editare a codului, gestionare a resurselor și depanare, ceea ce facilitează dezvoltarea și testarea aplicațiilor. Java, fiind un limbaj versatil și platformă-independent, permite crearea de aplicații Android care rulează pe mai multe dispozitive fără a necesita modificări semnificative.

Aplicația se structurează în trei pagini principale: "Home" (Acasă), "Settings" (Setări) și "Search" (Căutare). Aceste pagini pot fi accesate ușor folosind un meniu de navigare din partea de jos a aplicației, cunoscut sub numele de "bottom navigation menu".

Pagina "Home" reprezintă punctul central al aplicației, unde utilizatorii pot vedea și gestiona notificările primite de la portalul facultății. Prin intermediul acestei pagini, ei pot filtra și sorta notificările în funcție de preferințele lor, ceea ce îi ajută să rămână la curent cu informațiile importante.

Pagina "Settings" oferă utilizatorilor posibilitatea de a personaliza aplicația conform preferințelor lor. Aici pot seta ora preferată pentru primirea notificărilor și pot activa/dezactiva vibrațiile, ceea ce contribuie la adaptarea aplicației la nevoile individuale ale fiecărui utilizator.

Pagina "Search" vine în ajutorul utilizatorilor care doresc să găsească rapid anumite notificări sau informații specifice. Cu ajutorul funcționalităților de căutare, aceștia pot accesa rapid și eficient informațiile de interes pentru ei.

Un aspect deosebit de important în dezvoltarea acestei aplicații este integrarea serviciului OneSignal, care facilitează transmiterea notificărilor de la portalul facultății către dispozitivele mobile ale utilizatorilor. Astfel, utilizatorii pot fi mereu la curent cu informațiile recente și esențiale, fără a pierde timp în a căuta sau a verifica în mod repetat portalul.

În esență, această aplicație oferă o modalitate simplă și eficientă de a gestiona notificările de la portalul facultății, îmbunătățind astfel comunicarea și interacțiunea între utilizatori și informațiile furnizate de instituție.

***Termenii cheie:*** Android Studio, Java, OneSignal, home, settings, search, bottom navigation menu.

# PROLOG

# CUPRINSUL

<b>1</b>	<b>INTRODUCERE .....</b>	<b>1</b>
1.1	SCOPUL.....	1
1.2	MOTIVAȚIA.....	1
<b>2</b>	<b>CONCEPTE TEORETICE .....</b>	<b>3</b>
2.1	TEHNOLOGII.....	3
2.1.1	Java.....	3
2.1.2	XML (Extensible Markup Language) .....	3
2.2	INSTRUMENTE DE DEZVOLTARE .....	4
2.2.1	Android Studio .....	4
2.2.2	OneSignal.....	4
2.3	MOTIVAȚIA ALEGERII .....	4
<b>3</b>	<b>DESPRE APLICAȚIE .....</b>	<b>5</b>
3.1	PROIECTAREA APLICAȚIEI.....	5
3.1.1	Use case .....	5
3.1.2	Structura proiectului .....	5
3.2	FUNCȚIONALITĂȚI.....	7
3.2.1	MainActivity.....	7
3.2.2	HomeFragment.....	8
3.2.3	SettingsFragment.....	9
3.2.4	SearchFragment.....	10
3.2.5	SearchRecyclerAdpater .....	12
3.2.6	SearchResult .....	13
3.3	FIGURA 11. SEARCHRESULT CODE .....	13
3.3.1	NotificationAdapter .....	13
3.3.2	AppNotification.....	14
3.4	UI DESIGN .....	15
<b>4</b>	<b>CONCLUZII .....</b>	<b>21</b>
<b>5</b>	<b>BIBLIOGRAFIE .....</b>	<b>22</b>
<b>6</b>	<b>REFERINȚE WEB.....</b>	<b>23</b>
<b>A.</b>	<b>CODUL SURSĂ.....</b>	<b>24</b>
<b>B.</b>	<b>SITE-UL WEB AL PROIECTULUI .....</b>	<b>41</b>

C. CD / DVD .....	42
INDEX .....	43

# LISTA FIGURILOR

FIGURA 1. USE-CASE APLICAȚIE .....	5
FIGURA 2. INIȚIALIZARE ONE SIGNAL.....	7
FIGURA 3. MAINACTIVITY CODE .....	8
FIGURA 4. RECYCLERVIEW CODE.....	9
FIGURA 5. CATEGORY SPINNER CODE .....	9
FIGURA 6. NOTIFICATION TIME CODE.....	10
FIGURĂ 7. ALLOW VIBRATION CODE .....	10
FIGURA 8. SEARCHFRAGMENT CODE .....	11
FIGURA 9. PERFORMSEARCHANDGETRESULTS CODE .....	12
FIGURA 10. SEARCHRESULTADAPTER CODE .....	12
3.3 FIGURA 11. SEARCHRESULT CODE .....	13
FIGURA 12. NOTIFICATIONADAPTER CODE .....	14
FIGURA 13. APPNOTIFICATION CODE .....	15
FIGURA 14. HOME PAGE.....	16
FIGURA 15. SPINNER.....	17
FIGURA 16. SETTINGS PAGE.....	18
FIGURA 17. NOTIFICATION TIME.....	19
FIGURA 18. SEARCH PAGE.....	20

# 1 INTRODUCERE

## 1.1 Scopul

Scopul principal al acestei aplicații mobile este să ofere utilizatorilor o modalitate practică și eficientă de a gestiona notificările primite de la portalul facultății, utilizând tehnologia OneSignal. Această aplicație își propune să asigure că utilizatorii sunt mereu conectați la informațiile și evenimentele relevante din cadrul instituției lor de învățământ superior.

Prin intermediul unei interfețe prietenoase și cu opțiuni de personalizare, aplicația permite utilizatorilor să adapteze experiența lor la propriile preferințe. Cu funcționalități de filtrare și căutare, utilizatorii pot găsi cu ușurință informațiile de interes, sporind eficiența gestionării notificărilor.

De asemenea, integrarea serviciului OneSignal asigură transmiterea rapidă și fiabilă a notificărilor către dispozitivele mobile ale utilizatorilor, garantând astfel că aceștia sunt mereu la curent cu cele mai recente informații academice. În esență, această aplicație mobilă Android facilitează comunicarea între facultate și studenți, îmbunătățind gestionarea și accesibilitatea notificărilor academice.

## 1.2 Motivația

Motivele pentru care am ales și m-am arătat interesată de această temă sunt variate și bine întemeiate. În primul rând, consider că subiectul gestionării notificărilor reprezintă o componentă esențială în contextul actual, unde comunicarea digitală și accesul rapid la informații sunt vitale. O aplicație mobilă capabilă să faciliteze gestionarea notificărilor academice oferă un instrument valoros, benefic pentru studenți, ajutându-i să rămână conectați la evenimentele și actualizările din mediul academic.

În al doilea rând, recunosc utilitatea și avantajele pe care această aplicație le poate aduce utilizatorilor. Prin intermediul ei, studenții pot fi la curent cu anunțurile relevante și noutățile semnificative, contribuind astfel la îmbunătățirea calității experienței lor academice și la o gestionare mai eficientă a informațiilor. Aceasta înseamnă că aplicația poate crea un mediu mai informat și mai bine structurat în cadrul instituției de învățământ.

În plus, am constatat că dezvoltarea acestei aplicații reprezintă o oportunitate valoroasă pentru învățare și dezvoltare profesională. Abordarea și rezolvarea provocărilor tehnice și conceptuale



întâlnite pe parcursul procesului de dezvoltare au contribuit semnificativ la consolidarea abilităților mele în dezvoltarea de aplicații mobile și programare în general. Astfel, alegerea temei și angajamentul meu în acest proiect au generat beneficii considerabile în ceea ce privește dezvoltarea mea ca programator și înțelegerea aspectelor practice ale dezvoltării software.

În ansamblu, aceste motive justifică alegerea mea și interesul meu pentru această temă și evidențiază importanța și avantajele pe care aplicația le poate aduce utilizatorilor și dezvoltării mele personale.

## **2 CONCEPTE TEORETICE**

### **2.1 Tehnologii**

#### **2.1.1 Java**

Limbajul de programare Java a fost dezvoltat de Sun Microsystems la începutul anilor 1990. Deși este folosit în principal pentru aplicații bazate pe Internet, Java este un limbaj simplu, eficient, de uz general. Java a fost conceput inițial pentru aplicații de rețea încorporate care rulează pe mai multe platforme. Este un limbaj portabil, orientat pe obiecte, interpretat. [Sci01]

Java poate fi considerat atât un limbaj compilat, cât și un limbaj interpretat, deoarece codul său sursă este mai întâi compilat într-un octet-cod binar. Acest byte-code rulează pe Java Virtual Machine (JVM), care este de obicei un interpret bazat pe software. [Sci01]

Limbajul împrumută o mare parte din sintaxă de la C și C++, dar are un model al obiectelor mai simplu și prezintă mai puține facilități de nivel jos. [Wik01]

#### **2.1.2 XML (Extensible Markup Language)**

Extensible Markup Language (XML) este un limbaj de marcare și un format de fișier pentru stocarea, transmiterea și reconstrucția datelor arbitrare. Acesta definește un set de reguli pentru codificarea documentelor într-un format care este atât citibil de om, cât și citibil de mașină. [Wik02]

Obiectivele de proiectare ale XML subliniază simplitatea, generalitatea și capacitatea de utilizare pe internet. Este un format de date textuale cu suport puternic prin Unicode pentru diferite limbi umane. Deși proiectarea XML se concentrează pe documente, limbajul este utilizat pe scară largă pentru reprezentarea structurilor de date arbitrare, cum ar fi cele utilizate în serviciile web. [Wik02]

Scopul principal al XML este serializarea, adică stocarea, transmiterea și reconstrucția datelor arbitrare. Pentru ca două sisteme diferite să facă schimb de informații, trebuie să convină asupra unui format de fișier. XML standardizează acest proces. [Wik02]

## **2.2 Instrumente de dezvoltare**

### **2.2.1 Android Studio**

Android Studio este mediul de dezvoltare integrat (IDE) oficial pentru sistemul de operare Android de la Google, construit pe software-ul IntelliJ IDEA de la JetBrains și conceput special pentru dezvoltarea Android. [Wik03]

Android Studio a fost anunțat pe 16 mai 2013, la conferința Google I/O. A fost în stadiul de previzualizare cu acces timpuriu începând cu versiunea 0.1 în mai 2013, apoi a intrat în stadiul beta începând cu versiunea 0.8 care a fost lansată în iunie 2014. Prima versiune stabilă a fost lansată în decembrie 2014, începând cu versiunea 1.0. La sfârșitul anului 2015, Google a renunțat la suportul pentru Eclipse ADT, făcând Android Studio singurul IDE acceptat oficial pentru dezvoltarea Android. [Wik03]

Pe 7 mai 2019, Kotlin a înlocuit Java ca limbaj preferat de Google pentru dezvoltarea aplicațiilor Android. Java este încă acceptat, la fel ca și C++. [Wik03]

### **2.2.2 OneSignal**

OneSignal este cel mai rapid și mai de încredere serviciu pentru a trimite notificări push, mesaje în aplicație, SMS și e-mailuri. [Doc01]

OneSignal dă acum puterea peste un milion de companii să trimită 12 miliarde de mesaje zilnic. Una din cinci aplicații alege acum OneSignal. [One01]

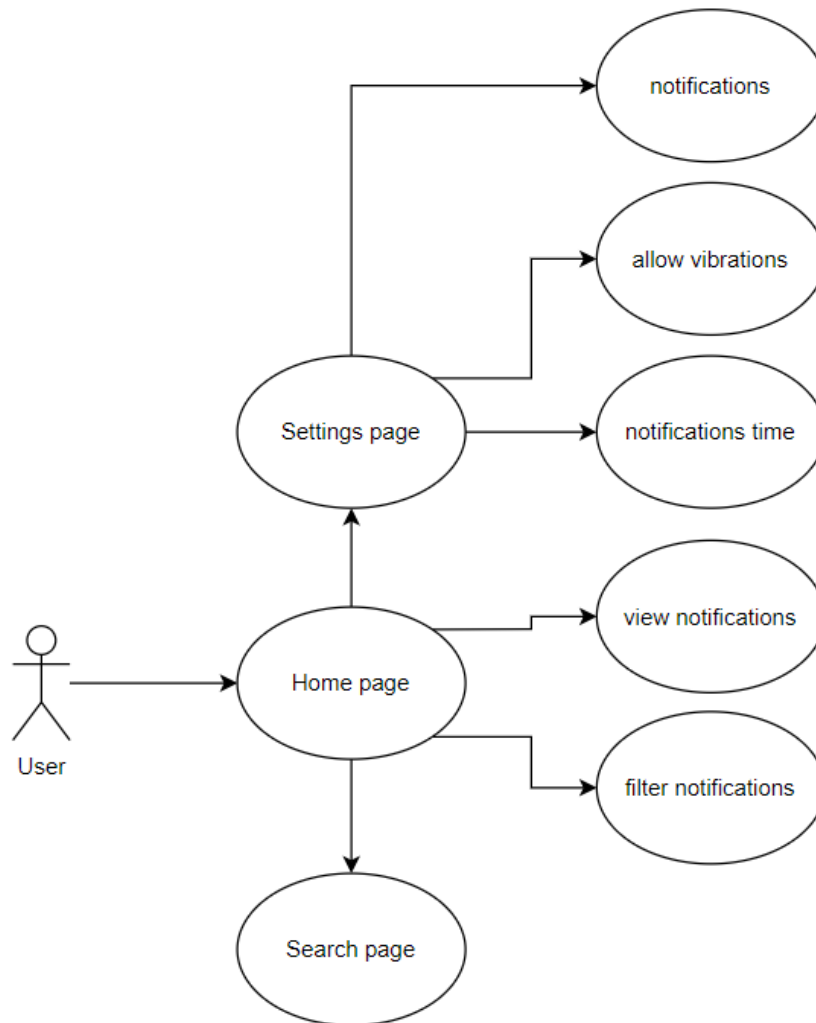
## **2.3 Motivația alegerii**

Am ales tehnologiile și instrumentele de dezvoltare pentru dezvoltarea aplicației mobile de gestionare a notificărilor cu baza solidă în considerente practice și eficiență. Java, ca limbaj de programare principal, oferă stabilitate și recunoaștere în dezvoltarea de aplicații Android. XML a fost utilizat pentru definirea interfeței de utilizator, iar Android Studio, mediul de dezvoltare oficial, a furnizat un set complet de instrumente și resurse specifice Android. OneSignal a fost alegerea evidentă pentru gestionarea notificărilor datorită facilității de utilizare și funcționalităților sale avansate. Aceste alegeri au condus la dezvoltarea unei aplicații Android robuste și eficiente pentru gestionarea notificărilor academice, adaptată nevoilor utilizatorilor.

### 3 DESPRE APLICAȚIE

#### 3.1 Proiectarea aplicației

##### 3.1.1 Use case



**Figura 1. Use-case aplicație**

##### 3.1.2 Structura proiectului

Proiectul este dezvoltat in Android Studio, care este structurat astfel:

- **manifests:** Acest director conține fișierul `AndroidManifest.xml`, care conține configurațiile specifice aplicației dvs. Android, inclusiv permisiunile, activitățile, serviciile și configurările necesare pentru funcționarea aplicației.
- **com.example.licenta:** Acesta este pachetul rădăcină al aplicației dvs., care conține codul sursă al aplicației Android. În interiorul acestui pachet, veți găsi subdirectoare și fișiere relevante pentru diferite componente ale aplicației, cum ar fi activități, fragmente, adaptoare, modele, utilitare și orice alte clase personalizate.
- **res:** Acest director conține resursele utilizate în aplicația dvs. Android.
- **drawable:** Aici se află resurse grafice, cum ar fi imagini, iconițe și alte fișiere vizuale.
- **layout:** Directorul care conține fișierele XML pentru aspectul interfeței de utilizator al aplicației, inclusiv layout-urile pentru diferite ecrane și componente.
- **menu:** Acesta conține fișiere XML pentru meniurile aplicației, utilizate pentru definirea opțiunilor de meniu.
- **values:** Aici sunt stocate resurse precum șiruri de caractere (`strings.xml`), dimensiuni (`dimens.xml`), stiluri (`styles.xml`) și alte valori de configurare.
- **xml:** Directorul care poate conține fișiere XML pentru orice alte configurații sau date necesare aplicației.
- **Gradle Scripts:** Acest director conține fișierele de configurare Gradle pentru proiectul dvs. Android. Aici sunt gestionate dependențele, configurările de construire și altele.

## 3.2 Funcționalități

### 3.2.1 MainActivity

MainActivity servește ca punct de intrare în aplicație și conține logica principală pentru gestionarea fragmentelor și a meniului de navigare.

La începutul clasei, este inițializată librăria OneSignal pentru gestionarea notificărilor push. Acest lucru se realizează în metoda onCreate. Librăria OneSignal este configurată pentru a afișa înregistrări detaliate de depanare (verbose logging) pentru a ajuta la detectarea și rezolvarea problemelor în timpul dezvoltării.

```
// Verbose Logging set to help debug issues, remove before releasing your app.  
OneSignal.getDebug().setLogLevel(LogLevel.VERBOSE);  
  
// OneSignal Initialization  
OneSignal.initWithContext(context: this, ONESIGNAL_APP_ID);
```

**Figura 2. Inițializare OneSignal**

Aplicația utilizează trei fragmente principale: HomeFragment, SettingsFragment și SearchFragment. Aceste fragmente sunt inițializate în metoda onCreate și gestionate cu ajutorul unui FragmentManager. De asemenea, am configurat un meniu de navigare în partea de jos a ecranului (BottomNavigationView) pentru a permite utilizatorului să navigheze între aceste fragmente. Atunci când utilizatorul selectează un element din meniul de navigare, se încarcă fragmentul corespunzător și se înlocuiește conținutul existent din activitate cu fragmentul selectat.

```

//Initialize fragments
homeFragment = new HomeFragment();
settingsFragment = new SettingsFragment();
searchFragment = new SearchFragment();

FragmentManager fragmentManager = getSupportFragmentManager();
FragmentTransaction fragmentTransaction = fragmentManager.beginTransaction();
fragmentTransaction.add(R.id.contentFrame, homeFragment);
fragmentTransaction.commit();

bottomNavigation = findViewById(R.id.bottomNavigation);
//Set the first item (Home) as selected when the app starts
bottomNavigation.setSelectedItemId(R.id.action_home);

bottomNavigation.setOnItemSelectedListener(item -> {
    Fragment selectedFragment = null;

    if (item.getItemId() == R.id.action_home) {
        selectedFragment = homeFragment;
    } else if (item.getItemId() == R.id.action_settings) {
        selectedFragment = settingsFragment;
    } else if (item.getItemId() == R.id.action_search) {
        selectedFragment = searchFragment;
    }

    FragmentTransaction transaction = fragmentManager.beginTransaction();
    transaction.replace(R.id.contentFrame, selectedFragment);
    transaction.commit();

    return true;
});

```

Figura 3. MainActivity code

### 3.2.2 HomeFragment

HomeFragment reprezintă o parte importantă a aplicației. Acest fragment este destinat afișării notificărilor și oferă funcționalități precum afișarea notificărilor filtrate după categorie.

La nivelul design-ului interfeței, acest fragment include un ‘RecyclerView’ pentru afișarea notificărilor și un ‘Spinner’ pentru selectarea categoriilor. Notificările sunt afișate utilizând un ‘NotificationAdapter’, care este responsabil pentru popularea listei de notificări în ‘RecyclerView’.

Utilizând RecyclerView, fragmentul afișează o listă de notificări pentru utilizator. Aceasta oferă o modalitate eficientă de a rula prin notificări și de a le vizualiza într-o listă scrollabilă.

```
recyclerView = rootView.findViewById(R.id.notificationRecyclerView);
recyclerView.setLayoutManager(new LinearLayoutManager(requireContext()));
notificationAdapter = new NotificationAdapter(requireContext(), new ArrayList<>());
recyclerView.setAdapter(notificationAdapter);
```

**Figura 4. RecyclerView code**

Prin intermediul unui Spinner (meniu derulant), utilizatorii pot selecta o categorie de notificări. Odată ce o categorie este selectată, notificările sunt filtrate pentru a afișa doar cele care aparțin acelei categorii.

```
Spinner categorySpinner = rootView.findViewById(R.id.categorySpinner);

//Create an ArrayAdapter and set it to the categorySpinner
ArrayAdapter<CharSequence> adapter = ArrayAdapter.createFromResource(
    requireContext(),
    R.array.notification_categories, //An array resource containing your categories
    android.R.layout.simple_spinner_item
);
//Specify the layout to use when the list of choices appears
adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
//Apply the adapter to the spinner
categorySpinner.setAdapter(adapter);
```

**Figura 5. Category Spinner code**

### 3.2.3 SettingsFragment

SettingsFragment oferă utilizatorilor control asupra unor setări importante ale aplicației.

Acest fragment include o opțiune numită "notification\_time", care permite utilizatorilor să aleagă ora la care doresc să primească notificările. Utilizând un obiect ListPreference, utilizatorii pot selecta ora preferată dintr-o listă de opțiuni disponibile. Atunci când utilizatorii își schimbă preferința, este salvată în mod corespunzător în preferințele aplicației.



```

ListPreference listPreference = findPreference( key: "notification_time");
if (listPreference != null) {
    //Setting the listener for changes in ListPreference
    listPreference.setOnPreferenceChangeListener(new Preference.OnPreferenceChangeListener() {
        1 usage
        @Override
        public boolean onPreferenceChange(@NonNull Preference preference, Object newValue) {
            savePreferredNotificationTime((String) newValue);

            return true; //Return true to enable saving
        }
    });
}

```

**Figura 6. Notification time code**

De asemenea, acest fragment include și o opțiune numită "allow\_vibration" sub forma unui SwitchPreferenceCompat. Atunci când utilizatorii activează această opțiune, vibrațiile sunt activate în aplicație. În caz contrar, vibrațiile sunt dezactivate. Această opțiune poate fi utilizată pentru personalizarea experienței de notificare a utilizatorului.

```

//Get the SwitchPreferenceCompat
SwitchPreferenceCompat allowVibrationPreference = findPreference( key: "allow_vibration");

//Set up a listener to detect preference changes
allowVibrationPreference.setOnPreferenceChangeListener((preference, newValue) -> {
    boolean allowVibration = (Boolean) newValue;

    if (allowVibration) {
        //Enable vibration
        enableVibration();
    } else {
        //Disable Vibration
        disableVibration();
    }

    return true;
});

```

**Figură 7. Allow vibration code**

### 3.2.4 SearchFragment

SearchFragment furnizează funcționalități de căutare în notificările disponibile.

Acest fragment include un câmp de introducere text numit `searchEditText`, în care utilizatorii pot introduce cuvintele cheie pentru căutarea notificărilor. Atunci când utilizatorii apasă tasta "Enter" sau efectuează o acțiune de căutare, se declanșează o căutare.

De asemenea, fragmentul conține și un `RecyclerView` numit `searchRecyclerView`, care afișează rezultatele căutării sub formă de liste. Atunci când utilizatorii introduc un termen de căutare și apasă tasta "Enter", fragmentul efectuează o căutare în lista de notificări disponibile și afișează rezultatele relevante în acest `recycler view`. Fiecare rezultat de căutare este afișat sub formă de element în listă.

```
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                          Bundle savedInstanceState) {

    View view = inflater.inflate(R.layout.fragment_search, container, attachToRoot: false);
    EditText searchEditText = view.findViewById(R.id.searchEditText);
    RecyclerView searchRecyclerView = view.findViewById(R.id.searchRecyclerView);

    searchEditText.setOnEditorActionListener((v, actionId, event) -> {
        if (actionId == EditorInfo.IME_ACTION_SEARCH) {
            String query = searchEditText.getText().toString().trim();
            List<SearchResult> searchResults = performSearchAndGetResults(query);
            SearchResultAdapter adapter = new SearchResultAdapter(requireContext(), searchResults);
            searchRecyclerView.setLayoutManager(new LinearLayoutManager(requireContext()));
            searchRecyclerView.setAdapter(adapter);

            return true;
        }
        return false;
    });

    return view;
}
```

**Figura 8. SearchFragment code**

Logica de căutare este gestionată în funcția `performSearchAndGetResults(query)`. Această funcție primește un cuvânt cheie introdus de utilizator și compară acest cuvânt cheie cu titlurile și descrierile notificărilor disponibile. Dacă un titlu sau o descriere conține termenul căutat (ignorând diferențele de literă mică/mare), notificarea este inclusă în lista de rezultate a căutării.

```

private List<SearchResult> performSearchAndGetResults(String query) {
    List<SearchResult> searchResults = new ArrayList<>();
    for (AppNotification notification : notificationsList) {
        if (notification.getTitle().toLowerCase().contains(query.toLowerCase()) ||
            notification.getDescription().toLowerCase().contains(query.toLowerCase())) {
            // Convert the AppNotification to a SearchResult and add it to the searchResults list
            SearchResult searchResult = new SearchResult(notification.getTitle(), notification.getDescription(), notification.getCategory());
            searchResults.add(searchResult);
        }
    }
    return searchResults;
}

```

**Figura 9. performSearchAndGetResults code**

Rezultatele căutării sunt afișate în searchRecyclerView utilizând un adaptor numit SearchResultAdapter. Adaptorul convertește obiectele de tip SearchResult în elemente afișate în recycler view, iar lista rezultatelor de căutare este actualizată în funcție de termenul de căutare introdus de utilizator.

### 3.2.5 SearchRecyclerAdpater

SearchResultAdapter este un adaptor personalizat pentru RecyclerView care se ocupă de afișarea rezultatelor căutării în fragmentul 'SearchFragment'. Acest adaptor primește un context și o listă de obiecte 'SearchResult', pe care le leagă de elementele vizuale din RecyclerView. În cadrul metodei 'onCreateViewHolder', se crează un obiect 'ViewHolder' pentru fiecare element de afișare, iar în metoda 'onBindViewHolder', datele sunt setate în câmpurile corespunzătoare ale elementului vizual. Cu ajutorul acestui adaptor, rezultatele căutării sunt prezentate într-o listă ordonată și ușor de navigat pentru utilizatorii aplicației Android.

```

public SearchResultAdapter(Context context, List<SearchResult> searchResults) {
    this.context = context;
    this.searchResults = searchResults;
}

@NonNull
@Override
public ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
    View view = LayoutInflater.from(context).inflate(R.layout.item_search_result, parent, attachToRoot: false);
    return new ViewHolder(view);
}

@Override
public void onBindViewHolder(@NonNull ViewHolder holder, int position) {
    SearchResult result = searchResults.get(position);
    holder.titleTextView.setText(result.getTitle());
    holder.descriptionTextView.setText(result.getDescription());
}

```

**Figura 10. SearchResultAdapter code**

### 3.2.6 SearchResult

Clasa SearchResult reprezintă un model de date folosit pentru a stoca informații despre rezultatele căutării. Această clasă are câteva atribute și un constructor pentru a inițializa obiectele SearchResult.

```
public class SearchResult {  
    2 usages  
    private String title;  
    2 usages  
    private String description;  
    2 usages  
    private String category;  
  
    1 usage  
    public SearchResult(String title, String description, String category) {  
        this.title = title;  
        this.description = description;  
        this.category = category;  
    }  
  
    1 usage  
    public String getTitle() { return title; }  
  
    1 usage  
    public String getDescription() { return description; }  
    no usages  
    public String getCategory() { return category; }  
}
```

## 3.3 Figura 11. SearchResult code

### 3.3.1 NotificationAdapter

NotificationAdapter este o componentă esențială a aplicației, responsabilă pentru afișarea notificărilor într-un mod organizat și interactiv în interfața de utilizator. Această clasă extinde ‘RecyclerView.Adapter’, gestionând atât datele notificărilor cât și crearea vizualizărilor pentru fiecare element din lista de notificări. Prin intermediul constructorului său, primește contextul aplicației și

lista de notificări, iar apoi, în metodele ‘onCreateViewHolder’ și ‘onBindViewHolder’, se ocupă de inflarea elementelor de interfață și de actualizarea acestora cu datele corespunzătoare notificărilor. Astfel, NotificationAdapter facilitează comunicarea eficientă între datele notificărilor și RecyclerView, asigurând că utilizatorii pot explora și interacționa cu notificările într-un mod prietenos și intuitiv.

```
public NotificationAdapter(Context context, List<AppNotification> notificationsList) {  
    this.context = context;  
    this.notificationsList = notificationsList;  
}  
  
@NonNull  
@Override  
public NotificationAdapter.NotificationViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {  
    View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.item_notification, parent, attachToRoot: false);  
  
    return new NotificationViewHolder(view);  
}  
  
@Override  
public void onBindViewHolder(@NonNull NotificationAdapter.NotificationViewHolder holder, int position) {  
    AppNotification notification = notificationsList.get(position);  
    holder.titleTextView.setText(notification.getTitle());  
    holder.descriptionTextView.setText(notification.getDescription());  
    holder.categoryTextView.setText(notification.getCategory());  
}
```

**Figura 12. NotificationAdapter code**

### 3.3.2 AppNotification

AppNotification este o clasă esențială a aplicației, reprezentând notificările care sunt afișate în aplicație. Această clasă are câteva atribute cheie, precum titlu, descriere și categorie, care stochează informații despre fiecare notificare. De asemenea, conține și atribute pentru URL-ul și imaginea asociate notificării, permițându-ți să furnizezi link-uri și conținut vizual relevant pentru utilizatori.

```

public class AppNotification {
    2 usages
    private String title;
    2 usages
    private String description;
    2 usages
    private String category;
    1 usage
    private String url;
    1 usage
    private String imageUrl;

    no usages
    public AppNotification(String title, String description, String category) {
        this.title = title;
        this.description= description;
        this.category =category;
    }

    3 usages
    public String getTitle() { return title; }
    3 usages
    public String getDescription() { return description; }
    3 usages
    public String getCategory() { return category; }
    no usages
    public String getUrl() { return url; }
    no usages
    public String getImageUrl() { return imageUrl; }
}

```

**Figura 13. AppNotification code**

### 3.4 UI Design

Când deschideți aplicația, pagina Home va fi afișată, oferindu-vă acces la notificările recente. Aceste notificări vor fi prezentate într-o listă scrollabilă și puteți naviga prin ele. Pentru a vă personaliza experiența, există un spinner, sau listă derulantă, care vă permite să alegeți categoria notificărilor pe care doriți să le vizualizați. Acest lucru vă ajută să filtrați notificările și să găsiți rapid informațiile de care aveți nevoie. Cu această combinație de funcționalități, aplicația vă oferă o experiență eficientă și personalizată pentru gestionarea și explorarea notificărilor.

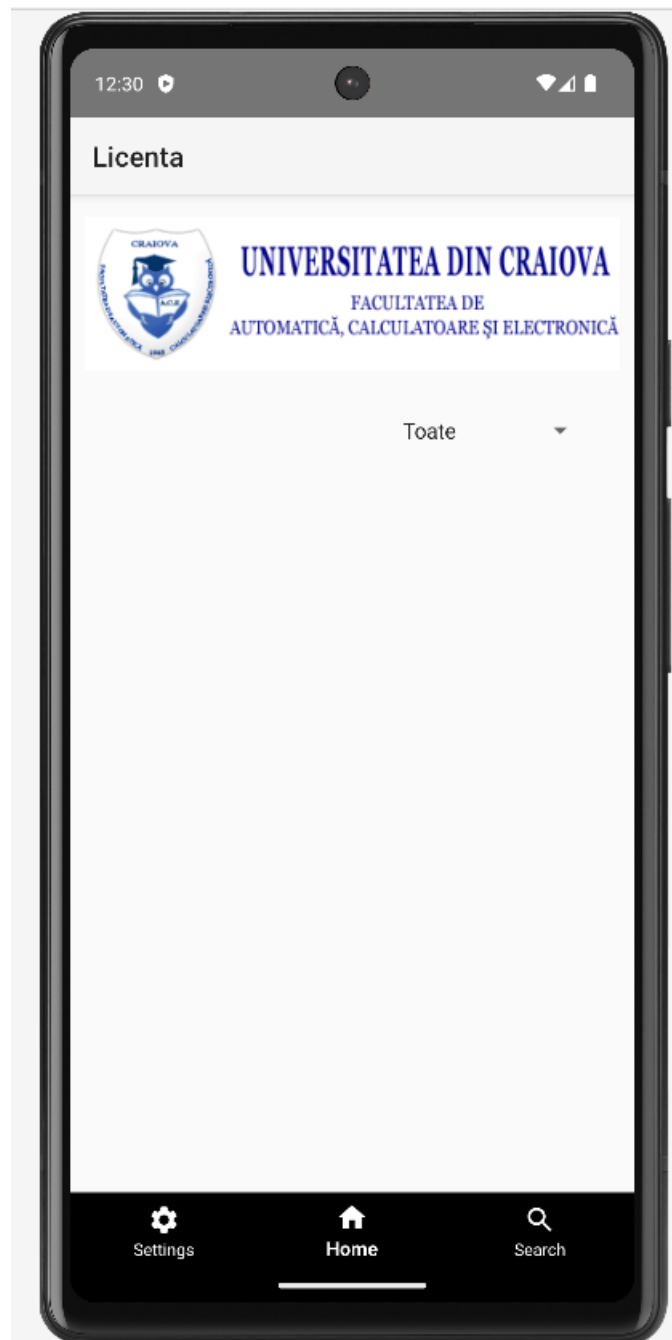
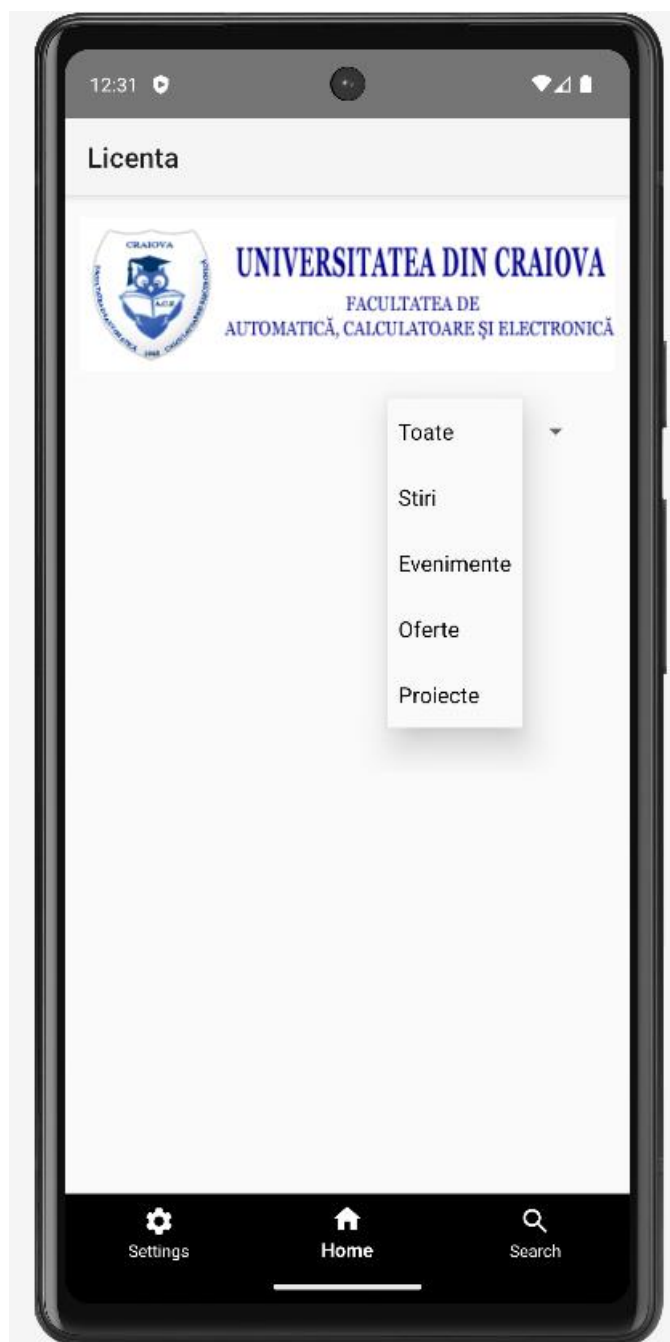


Figura 14. Home page

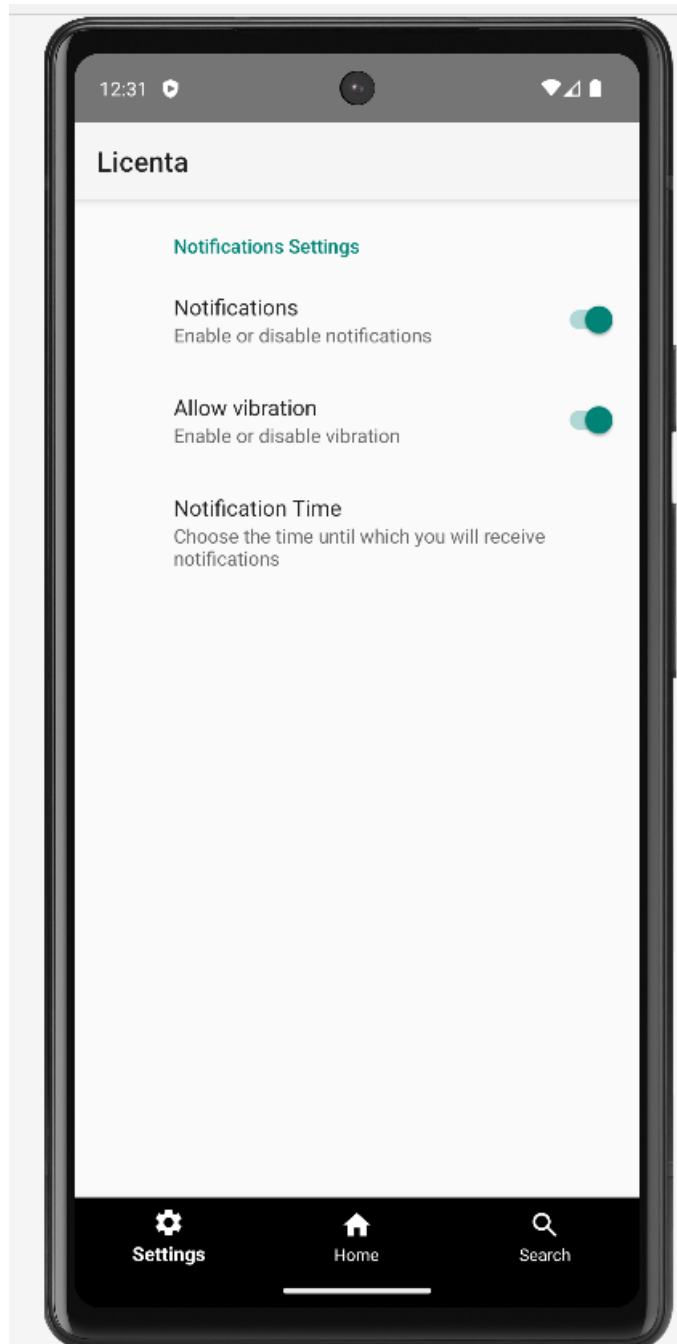


**Figura 15. Spinner**

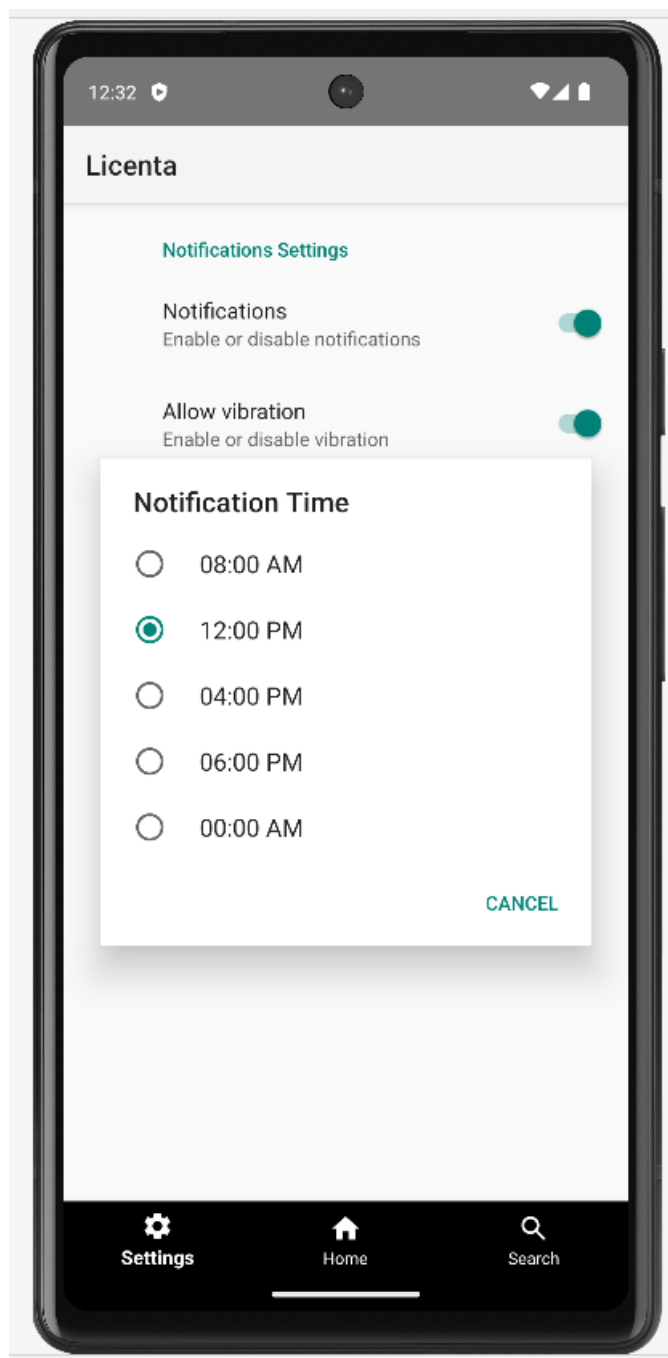
Din meniul de navigare din partea de jos a ecranului (bottom navigation), puteți accesa paginile de Setări (Settings) și Căutare (Search).

Pe pagina de Setări, aveți controlul asupra setărilor de notificare, unde puteți activa sau dezactiva notificările, permite sau interzice vibrațiile și selecta ora la care doriți să primiți notificările. Aceste opțiuni vă permit să personalizați cum doriți să fiți informat de aplicație.





**Figura 16. Settings page**



**Figura 17. Notification time**

Pe pagina de Căutare, puteți căuta notificări în funcție de cuvinte cheie, pentru a găsi rapid informațiile pe care le căutați. Cu aceste funcționalități adăugate la bara de navigație, aveți acces ușor la toate aspectele importante ale aplicației.



**Figura 18. Search page**

## 4 CONCLUZII

În concluzie, această aplicație reprezintă cel mai complex proiect al meu, și sunt extrem de mândră că am reușit să o finalizez. Realizarea acestei aplicații a presupus depășirea unor obstacole dificile și rezolvarea unor erori pe care le consideram inițial imposibile. Această experiență m-a ajutat să devin un programator mai competent și mai încrezător în abilitățile mele. Sunt, de asemenea, bucuroasă că am avut oportunitatea de a aprofunda cunoștințele mele în limbajul Java și utilizarea Android Studio, precum și în gestionarea problemelor și tratarea erorilor. Cei patru ani de facultate au adus cu sine o mulțime de învățăminte și realizări, iar acest proiect reprezintă o culminare a eforturilor mele în domeniul dezvoltării de aplicații mobile.

## 5 BIBLIOGRAFIE

- [Doc01] - <https://documentation.onesignal.com/docs>
- [One01] - <https://onesignal.com/about>
- [Sci01] - <https://www.sciencedirect.com/topics/computer-science/java-programming-language>
- [Wik01] - [https://ro.wikipedia.org/wiki/Java\\_\(limbaj\\_de\\_programare\)](https://ro.wikipedia.org/wiki/Java_(limbaj_de_programare))
- [Wik02] - <https://en.wikipedia.org/wiki/XML>
- [Wik03] - [https://en.wikipedia.org/wiki/Android\\_Studio](https://en.wikipedia.org/wiki/Android_Studio)

## 6 REFERINȚE WEB

- [Doc01] - <https://documentation.onesignal.com/docs>
- [One01] - <https://onesignal.com/about>
- [Sci01] - <https://www.sciencedirect.com/topics/computer-science/java-programming-language>
- [Wik01] - [https://ro.wikipedia.org/wiki/Java\\_\(limbaj\\_de\\_programare\)](https://ro.wikipedia.org/wiki/Java_(limbaj_de_programare))
- [Wik02] - <https://en.wikipedia.org/wiki/XML>
- [Wik03] - [https://en.wikipedia.org/wiki/Android\\_Studio](https://en.wikipedia.org/wiki/Android_Studio)

## A. CODUL SURSĂ

### Main Activity

```
package com.example.licenta;

import android.os.Bundle;

import androidx.appcompat.app.AppCompatActivity;
import androidx.fragment.app.Fragment;
import androidx.fragment.app.FragmentManager;
import androidx.fragment.app.FragmentTransaction;

import com.google.android.material.bottomnavigation.BottomNavigationView;
import com.onesignal.OneSignal;
import com.onesignal.debug.LogLevel;

import org.json.JSONObject;

import java.util.ArrayList;
import java.util.List;

public class MainActivity extends AppCompatActivity {

    private static final String ONESIGNAL_APP_ID = "a836eb27-1c34-4a97-b8c5-284a4becb06e";

    private BottomNavigationView bottomNavigation;

    private HomeFragment homeFragment;
    private SettingsFragment settingsFragment;
    private SearchFragment searchFragment;
    private NotificationAdapter notificationAdapter;
    private List<AppNotification> notificationsList = new ArrayList<>();

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // Verbose Logging set to help debug issues, remove before releasing your app.
        OneSignal.getDebug().setLogLevel(LogLevel.VERBOSE);

        // OneSignal Initialization
        OneSignal.initWithContext(this, ONESIGNAL_APP_ID);

        // optIn will show the native Android notification permission prompt.
        // We recommend removing the following code and instead using an In-App Message to prompt for notification permission (See step 7)
        //OneSignal.getUser().getPushSubscription().optIn();

        //Set handler for opening notifications
        /*OneSignal.setNotificationOpenedHandler(result -> {
            OSNotificationOpenedResult openedResult =

```

```

(OSNotificationOpenedResult) result;
    JSONObject data =
openedResult.getNotification().getAdditionalData();
    if (data != null) {
        String title = data.optString("title", null);
        String description = data.optString("description", null);
        String category = data.optString("category", null);
        if (title != null && description != null && category !=
null) {
            //Add the notification to the notification list
            AppNotification appNotification = new
AppNotification(title, description, category);
            notificationsList.add(appNotification);
            //Update the adapter to reflect the changes
            notificationAdapter.notifyDataSetChanged();
        }
    }
});*/

//Initialize fragments
homeFragment = new HomeFragment();
settingsFragment = new SettingsFragment();
searchFragment = new SearchFragment();

FragmentManager fragmentManager = getSupportFragmentManager();
FragmentTransaction fragmentTransaction =
fragmentManager.beginTransaction();
fragmentTransaction.add(R.id.contentFrame, homeFragment);
fragmentTransaction.commit();

bottomNavigation = findViewById(R.id.bottomNavigation);
//Set the first item (Home) as selected when the app starts
bottomNavigation.setSelectedItemId(R.id.action_home);

bottomNavigation.setOnItemClickListener(item -> {
    Fragment selectedFragment = null;

    if (item.getItemId() == R.id.action_home) {
        selectedFragment = homeFragment;
    } else if (item.getItemId() == R.id.action_settings) {
        selectedFragment = settingsFragment;
    } else if (item.getItemId() == R.id.action_search) {
        selectedFragment = searchFragment;
    }

    FragmentTransaction transaction =
fragmentManager.beginTransaction();
    transaction.replace(R.id.contentFrame, selectedFragment);
    transaction.commit();

    return true;
});
}

public interface NotificationHandler {
    void onNotificationReceived(AppNotification notification);
}
}

```



## Home Fragment

```
package com.example.licenta;

import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.ArrayAdapter;
import android.widget.Spinner;

import androidx.fragment.app.Fragment;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import java.util.ArrayList;
import java.util.List;

/**
 * A simple {@link Fragment} subclass.
 * Use the {@link HomeFragment#newInstance} factory method to
 * create an instance of this fragment.
 */
public class HomeFragment extends Fragment {

    // TODO: Rename parameter arguments, choose names that match
    // the fragment initialization parameters, e.g. ARG_ITEM_NUMBER
    private static final String ARG_PARAM1 = "param1";
    private static final String ARG_PARAM2 = "param2";

    // TODO: Rename and change types of parameters
    private String mParam1;
    private String mParam2;

    private RecyclerView recyclerView;
    private NotificationAdapter notificationAdapter;

    public HomeFragment() {
        // Required empty public constructor
    }

    /**
     * Use this factory method to create a new instance of
     * this fragment using the provided parameters.
     *
     * @param param1 Parameter 1.
     * @param param2 Parameter 2.
     * @return A new instance of fragment HomeFragment.
     */
    // TODO: Rename and change types and number of parameters
    public static HomeFragment newInstance(String param1, String param2) {
        HomeFragment fragment = new HomeFragment();
        Bundle args = new Bundle();
        args.putString(ARG_PARAM1, param1);
        args.putString(ARG_PARAM2, param2);
        fragment.setArguments(args);
        return fragment;
    }
}
```

```

    }

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        if (getArguments() != null) {
            mParam1 = getArguments().getString(ARG_PARAM1);
            mParam2 = getArguments().getString(ARG_PARAM2);
        }
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {

        View rootView = inflater.inflate(R.layout.fragment_home, container,
false);

        recyclerView =
rootView.findViewById(R.id.notificationRecyclerView);
        recyclerView.setLayoutManager(new
LinearLayoutManager(requireContext()));
        notificationAdapter = new NotificationAdapter(requireContext(), new
ArrayList<>());
        recyclerView.setAdapter(notificationAdapter);

        Spinner categorySpinner =
rootView.findViewById(R.id.categorySpinner);

        //Create an ArrayAdapter and set it to the categorySpinner
        ArrayAdapter<CharSequence> adapter =
ArrayAdapter.createFromResource(
            requireContext(),
            R.array.notification_categories, //An array resource
containing your categories
            android.R.layout.simple_spinner_item
        );
        //Specify the layout to use when the list of choices appears

        adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_it
em);
        //Apply the adapter to the spinner
        categorySpinner.setAdapter(adapter);

        categorySpinner.setOnItemSelectedListener(new
AdapterView.OnItemSelectedListener() {
            @Override
            public void onItemSelected(AdapterView<?> adapterView, View
view, int position, long id) {
                String selectedCategory =
adapterView.getItemAtPosition(position).toString();
                List<AppNotification> filteredNotifications =
filterNotificationsByCategory(selectedCategory);
                NotificationAdapter notificationAdapter = new
NotificationAdapter(requireContext(), filteredNotifications);
                recyclerView.setAdapter(notificationAdapter);
            }

            @Override

```

```

        public void onNothingSelected(AdapterView<?> adapterView) {

        }

    });

    return rootView;
}

private List<AppNotification> filterNotificationsByCategory(String
selectedCategory) {
    List<AppNotification> notifications =
NotificationDataManager.getInstance().getNotifications();
    List<AppNotification> filteredList = new ArrayList<>();
    if (notifications != null) {
        for (AppNotification notification : notifications) {
            if (notification.getCategory().equals(selectedCategory)) {
                filteredList.add(notification);
            }
        }
    }
    return filteredList;
}
}

```

## **Settings Fragment**

```

package com.example.licenta;

import android.app.AlarmManager;
import android.content.Context;
import android.content.SharedPreferences;
import android.os.Build;
import android.os.Bundle;
import android.os.VibrationEffect;
import android.os.Vibrator;

import androidx.annotation.NonNull;
import androidx.preference.ListPreference;
import androidx.preference.Preference;
import androidx.preference.PreferenceFragmentCompat;
import androidx.preference.PreferenceManager;
import androidx.preference.SwitchPreferenceCompat;

import com.onesignal.OneSignal;

import java.util.Calendar;

public class SettingsFragment extends PreferenceFragmentCompat {

    @Override
    public void onCreatePreferences(Bundle savedInstanceState, String
rootKey) {
        setPreferencesFromResource(R.xml.preferences, rootKey);

        //Set the notification time
        //Get reference to ListPreference
        ListPreference listPreference =

```

```

findPreference("notification_time");
    if (listPreference != null) {
        //Setting the listener for changes in ListPreference
        listPreference.setOnPreferenceChangeListener(new
Preference.OnPreferenceChangeListener() {
            @Override
            public boolean onPreferenceChange(@NonNull Preference
preference, Object newValue) {
                savePreferredNotificationTime((String) newValue);

                return true; //Return true to enable saving
            }
        });
    }

    //Set the Theme
    /*ListPreference themePreference = findPreference("theme");
    themePreference.setOnPreferenceChangeListener((preference,
newValue) -> {
        String selectedTheme = (String) newValue;
        getActivity().setTheme(getThemeId(selectedTheme));
        getActivity().recreate();

        return true;
    });*/

    //Get the SwitchPreferenceCompat
    SwitchPreferenceCompat allowVibrationPreference =
findPreference("allow_vibration");

    //Set up a listener to detect preference changes
    allowVibrationPreference.setOnPreferenceChangeListener((preference,
newValue) -> {
        boolean allowVibration = (Boolean) newValue;

        if (allowVibration) {
            //Enable vibration
            enableVibration();
        } else {
            //Disable Vibration
            disableVibration();
        }

        return true;
    });

    //Get the SwitchPreferenceCompat
    //SwitchPreferenceCompat enableNotificationsPreference =
findPreference("notifications");

    //Set up a listener to detect preference changes

    /*enableNotificationsPreference.setOnPreferenceChangeListener((preference,
newValue) -> {
        boolean enableNotifications = (Boolean) newValue;

        if (enableNotifications) {
            //Enable notifications
            enableNotifications();

```

```

        } else {
            //Disable notifications
            disableNotifications();
        }

        return true;
    });*/
}

private void savePreferredNotificationTime(String time) {
    SharedPreferences sharedPreferences =
PreferenceManager.getDefaultSharedPreferences(requireContext());
    SharedPreferences.Editor editor = sharedPreferences.edit();
    editor.putString("preferred_notification_time", time);
    editor.apply();
}

/*private int getThemeId(String themeName) {
    switch (themeName) {
        case "AppTheme.Light":
            return R.style.AppTheme_Light;
        case "AppTheme.Dark":
            return R.style.AppTheme_Dark;
        default:
            return R.style.AppTheme_Light;
    }
}*/

private void enableVibration() {
    Vibrator vibrator = (Vibrator)
requireActivity().getSystemService(Context.VIBRATOR_SERVICE);
    if (vibrator != null && vibrator.hasVibrator()) {
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
            vibrator.vibrate(VibrationEffect.createOneShot(100,
VibrationEffect.DEFAULT_AMPLITUDE));
        }
    }
}

private void disableVibration() {
    Vibrator vibrator = (Vibrator)
requireActivity().getSystemService(Context.VIBRATOR_SERVICE);
    if (vibrator != null && vibrator.hasVibrator()) {
        // Cancel any ongoing vibration
        vibrator.cancel();
    }
}
}

```

## **Search Fragment**

```

package com.example.licenta;

import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;

```

```

import android.view.ViewGroup;
import android.view.inputmethod.EditorInfo;
import android.widget.EditText;

import androidx.fragment.app.Fragment;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import java.util.ArrayList;
import java.util.List;

/**
 * A simple {@link Fragment} subclass.
 * Use the {@link SearchFragment#newInstance} factory method to
 * create an instance of this fragment.
 */
public class SearchFragment extends Fragment {

    // TODO: Rename parameter arguments, choose names that match
    // the fragment initialization parameters, e.g. ARG_ITEM_NUMBER
    private static final String ARG_PARAM1 = "param1";
    private static final String ARG_PARAM2 = "param2";

    // TODO: Rename and change types of parameters
    private String mParam1;
    private String mParam2;

    private List<AppNotification> notificationsList = new ArrayList<>();

    public SearchFragment() {
        // Required empty public constructor
    }

    /**
     * Use this factory method to create a new instance of
     * this fragment using the provided parameters.
     *
     * @param param1 Parameter 1.
     * @param param2 Parameter 2.
     * @return A new instance of fragment SearchFragment.
     */
    // TODO: Rename and change types and number of parameters
    public static SearchFragment newInstance(String param1, String param2)
    {
        SearchFragment fragment = new SearchFragment();
        Bundle args = new Bundle();
        args.putString(ARG_PARAM1, param1);
        args.putString(ARG_PARAM2, param2);
        fragment.setArguments(args);
        return fragment;
    }

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        if (getArguments() != null) {
            mParam1 = getArguments().getString(ARG_PARAM1);
            mParam2 = getArguments().getString(ARG_PARAM2);
        }
    }
}

```

```

    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {

        View view = inflater.inflate(R.layout.fragment_search, container,
false);
        EditText searchEditText = view.findViewById(R.id.searchEditText);
        RecyclerView searchRecyclerView =
view.findViewById(R.id.searchRecyclerView);

        searchEditText.setOnEditorActionListener((v, actionId, event) -> {
            if (actionId == EditorInfo.IME_ACTION_SEARCH) {
                String query = searchEditText.getText().toString().trim();
                List<SearchResult> searchResults =
performSearchAndGetResults(query);
                SearchResultAdapter adapter = new
SearchResultAdapter(requireContext(), searchResults);
                searchRecyclerView.setLayoutManager(new
LinearLayoutManager(requireContext()));
                searchRecyclerView.setAdapter(adapter);

                return true;
            }
            return false;
        });

        return view;
    }

    private List<SearchResult> performSearchAndGetResults(String query) {
        List<SearchResult> searchResults = new ArrayList<>();
        for (AppNotification notification : notificationsList) {
            if
(notification.getTitle().toLowerCase().contains(query.toLowerCase()) ||
notification.getDescription().toLowerCase().contains(query.toLowerCase()))
{
                // Convert the AppNotification to a SearchResult and add it
to the searchResults list
                SearchResult searchResult = new
SearchResult(notification.getTitle(), notification.getDescription(),
notification.getCategory());
                searchResults.add(searchResult);
            }
        }
        return searchResults;
    }
}

```

## **SearchResultAdapter**

```

package com.example.licenta;

import android.content.Context;
import android.view.LayoutInflater;

```

```

import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;
import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;

import com.example.licenta.R;
import com.example.licenta.SearchResult;

import java.util.List;

public class SearchResultAdapter extends
RecyclerView.Adapter<SearchResultAdapter.ViewHolder> {

    private List<SearchResult> searchResults;
    private Context context;

    public SearchResultAdapter(Context context, List<SearchResult>
searchResults) {
        this.context = context;
        this.searchResults = searchResults;
    }

    @NonNull
    @Override
    public ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int
viewType) {
        View view =
LayoutInflater.from(context).inflate(R.layout.item_search_result, parent,
false);
        return new ViewHolder(view);
    }

    @Override
    public void onBindViewHolder(@NonNull ViewHolder holder, int position)
{
        SearchResult result = searchResults.get(position);
        holder.titleTextView.setText(result.getTitle());
        holder.descriptionTextView.setText(result.getDescription());
    }

    @Override
    public int getItemCount() {
        return searchResults.size();
    }

    public class ViewHolder extends RecyclerView.ViewHolder {
        TextView titleTextView;
        TextView descriptionTextView;

        public ViewHolder(View itemView) {
            super(itemView);
            titleTextView = itemView.findViewById(R.id.titleTextView);
            descriptionTextView =
itemView.findViewById(R.id.descriptionTextView);
        }
    }
}

```



## **SearchResult**

```
package com.example.licenta;

public class SearchResult {
    private String title;
    private String description;
    private String category;

    public SearchResult(String title, String description, String category)
    {
        this.title = title;
        this.description = description;
        this.category = category;
    }

    public String getTitle() {
        return title;
    }

    public String getDescription() {
        return description;
    }

    public String getCategory() { return category; }
}
```

## **NotificationAdapter**

```
package com.example.licenta;

import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;

import com.example.licenta.AppNotification;
import com.example.licenta.R;

import java.util.List;

public class NotificationAdapter extends
RecyclerView.Adapter<NotificationAdapter.NotificationViewHolder> {

    private List<AppNotification> notificationsList;
    private Context context;

    public NotificationAdapter(Context context, List<AppNotification>
notificationsList) {
        this.context = context;
        this.notificationsList = notificationsList;
    }
}
```

```

        @NonNull
        @Override
        public NotificationAdapter.NotificationViewHolder
onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
            View view =
LayoutInflater.from(parent.getContext()).inflate(R.layout.item_notification
, parent, false);

            return new NotificationViewHolder(view);
        }

        @Override
        public void onBindViewHolder(@NonNull
NotificationAdapter.NotificationViewHolder holder, int position) {
            AppNotification notification = notificationsList.get(position);
            holder.titleTextView.setText(notification.getTitle());
            holder.descriptionTextView.setText(notification.getDescription());
            holder.categoryTextView.setText(notification.getCategory());
        }

        @Override
        public int getItemCount() {
            return notificationsList.size();
        }

        public class NotificationViewHolder extends RecyclerView.ViewHolder {
            TextView titleTextView, descriptionTextView, categoryTextView;

            public NotificationViewHolder(@NonNull View itemView) {
                super(itemView);
                titleTextView = itemView.findViewById(R.id.titleTextView);
                descriptionTextView =
itemView.findViewById(R.id.descriptionTextView);
                categoryTextView =
itemView.findViewById(R.id.categoryTextView);
            }
        }
    }
}

```

## **AppNotification**

```

package com.example.licenta;

public class AppNotification {
    private String title;
    private String description;
    private String category;
    private String url;
    private String imageUrl;

    public AppNotification(String title, String description, String
category) {
        this.title = title;
        this.description= description;
        this.category =category;
    }
}

```

```

    public String getTitle() {
        return title;
    }
    public String getDescription() {
        return description;
    }
    public String getCategory() {
        return category;
    }
    public String getUrl() {
        return url;
    }
    public String getImageUrl() {
        return imageUrl;
    }
}

```

### **activity\_main.xml**

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <!-- Content area -->
    <FrameLayout
        android:id="@+id/contentFrame"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1">

    </FrameLayout>

    <!-- Bottom navigation -->
    <com.google.android.material.bottomnavigation.BottomNavigationView
        android:id="@+id/bottomNavigation"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:itemTextColor="@color/white"
        app:itemIconTint="@color/white"
        app:menu="@menu/bottom_nav_menu" />

</LinearLayout>

```

### **fragment\_home.xml**

```

<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"

```

```

tools:context=".HomeFragment">

<ImageView
    android:id="@+id/logoImageView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:src="@drawable/logo2"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="16dp"/>

<Spinner
    android:id="@+id/categorySpinner"
    android:layout_width="wrap_content"
    android:layout_height="48dp"
    android:layout_below="@+id/logoImageView"
    android:layout_alignParentEnd="true"
    android:layout_marginTop="20dp"
    android:layout_marginEnd="30dp"/>

<!--<EditText
    android:id="@+id/searchEditText"
    android:layout_width="match_parent"
    android:layout_height="48dp"
    android:background="@drawable/search_background"
    android:drawableStart="@drawable/ic_search"
    android:hint="Search..."
    android:paddingStart="16dp"
    android:paddingEnd="16dp"
    android:layout_marginTop="20dp"
    android:layout_marginEnd="16dp"
    android:layout_marginStart="55dp"
    android:inputType="text"
    android:layout_below="@+id/logoImageView"
    android:layout_toEndOf="@+id/categorySpinner" /> -->

<androidx.recyclerview.widget.RecyclerView
    android:id="@+id/notificationRecyclerView"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_below="@+id/categorySpinner"
    android:layout_marginTop="16dp"/>

</RelativeLayout>

```

### **fragment\_search.xml**

```

<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    tools:context=".SearchFragment">

    <EditText
        android:id="@+id/searchEditText"

```

```

        android:layout_width="0dp"
        android:layout_height="48dp"
        android:hint="Search"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        android:layout_margin="16dp" />

<androidx.recyclerview.widget.RecyclerView
    android:id="@+id/searchRecyclerView"
    android:layout_width="0dp"
    android:layout_height="0dp"
    app:layout_constraintTop_toBottomOf="@+id/searchEditText"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintBottom_toBottomOf="parent"
    android:layout_marginTop="16dp"
    android:layout_marginBottom="16dp" />

</androidx.constraintlayout.widget.ConstraintLayout>

```

### **item\_search\_result.xml**

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:padding="16dp">

    <TextView
        android:id="@+id/titleTextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textStyle="bold"
        android:textSize="16sp"
        android:textColor="@color/black" />

    <TextView
        android:id="@+id/descriptionTextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="8dp"
        android:textSize="14sp"
        android:textColor="@color/dark_gray" />
</LinearLayout>

```

### **preferences.xml**

```

<?xml version="1.0" encoding="utf-8"?>
<PreferenceScreen
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:android="http://schemas.android.com/apk/res/android">

    <PreferenceCategory

```

```

    app:title="Notifications Settings">

    <SwitchPreferenceCompat
        android:defaultValue="true"
        android:key="notifications"
        android:title="Notifications"
        android:summary="Enable or disable notifications" />

    <SwitchPreferenceCompat
        android:key="allow_vibration"
        android:title="Allow vibration"
        android:summary="Enable or disable vibration"
        android:defaultValue="true" />

    <ListPreference
        android:key="notification_time"
        android:title="Notification Time"
        android:summary="Choose the time until which you will receive
notifications"
        android:entries="@array/notification_times"
        android:entryValues="@array/notification_time_values"
        android:defaultValue="12:00" />

    </PreferenceCategory>

    <!--<PreferenceCategory
        android:title="Appearance">

        <ListPreference
            android:key="theme"
            android:title="Theme"
            android:entries="@array/theme_entries"
            android:entryValues="@array/theme_values"
            android:defaultValue="AppTheme.Light" />

    </PreferenceCategory> -->

</PreferenceScreen>

```

### **item\_notification.xml**

```

<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:padding="16dp">

    <ImageView
        android:id="@+id/notificationImage"
        android:layout_width="80dp"
        android:layout_height="80dp"
        android:layout_alignParentStart="true"
        android:scaleType="centerCrop" />

    <TextView

```

```

        android:id="@+id/titleTextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="@style/TextAppearance.AppCompat.Large"
        android:text="Notification Title"
        android:textColor="@color/black"
        android:layout_toEndOf="@+id/notificationImage"/>

<TextView
    android:id="@+id/descriptionTextView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="@style/TextAppearance.AppCompat.Body1"
    android:text="Notification Description"
    android:textColor="@color/dark_gray"
    android:layout_below="@id/titleTextView"
    android:layout_toEndOf="@id/notificationImage"/>

<TextView
    android:id="@+id/categoryTextView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="@style/TextAppearance.AppCompat.Body2"
    android:text="Notification Category"
    android:textColor="@color/light_gray"
    android:layout_below="@id/descriptionTextView"
    android:layout_toEndOf="@id/notificationImage"/>

</RelativeLayout>

```

### **bottom nav menu.xml**

```

<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">

    <item
        android:id="@+id/action_settings"
        android:title="Settings"
        android:icon="@drawable/ic_settings" />

    <item
        android:id="@+id/action_home"
        android:title="Home"
        android:icon="@drawable/ic_home" />

    <item
        android:id="@+id/action_search"
        android:title="Search"
        android:icon="@drawable/ic_search" />

</menu>

```

## **B. SITE-UL WEB AL PROIECTULUI**

Depozitul GitHub pentru aplicația mea mobilă este  
<https://github.com/PatruGabrielaAndreea/Licenta-MobileApp.git>



## **C. CD / DVD**

Autorul atașează în această anexă obligatorie, versiunea electronică a aplicației, a acestei lucrări, precum și prezentarea finală a tezei.



# INDEX

## **B**

Bibliografie..... 10

## **C**

**CUPRINSUL** ..... xiii

## **D**

Dimensiuni ..... 4

## **L**

**LISTA FIGURILOR**.....xiv

**LISTA TABELELOR**..... xv

## **R**

Referințe web..... 11

## **S**

Structura documentului ..... 3