**Proiect Android DATM**

**Orarul autobuzelor Cluj**

Îndrumător: Student:

Sl. Dr. Ing. Adriana Stan Macsim Alexandru

TC master an II

Facultatea de Electronică,

Telecomunicații și Tehnologia Informațiilor,

Universitatea Tehnică Cluj Napoca

**Orarul Autobuzelor Cluj-Napoca, Aplicația „Ratuc”**

Prezentul proiect presupune utilizarea fișierelor cu listele orarelor autobuzelor din Cluj-Napoca. Am ales acest proiect deoarece mijloacele de transport în comun reprezintă după părerea mea cea mai rapidă metodă de deplasare în Cluj-Napoca. Aceasta este o aplicație practică ce poate fi utilizată pe telefoanele cu sistemul de operare Android.

Numele acestei aplicații este sugestiv deoarece în cazul în care ar fi implementată în Google Play ar putea fii căutată foarte ușor de către utilizatori. Acesta este intitulat „Ratuc”, reprezentând Regia Autonoma de Transport Urban Cluj.

Am folosit ca și mediu de dezvoltare Android Studio, un mediu integrat (IDE) pentru dezvoltarea aplicațiilor Android, bazat pe Intellij IDEA. Pe lângă editorul de cod și instrumentele de dezvoltare Intellij, acesta oferă mai multe caracteristici ce îmbunătățesc funcționalitățile unei aplicații Android. Câteva exemple:

* Un sistem flexibil de build bazat pe Grandle
* Un emulator rapid și performant
* Un mediu de dezvoltare complex în care se pot dezvolta aplicații pentru toate tipurile de dispozitive Android
* Rulare instantanee și push în aplicații fără a crea un nou fișier APK
* Șabloane de cod și integrare GitHub pentru ajutorul în creearea aplicațiilor, etc.

Aplicația a fost creată și gândită astfel încât să poată fi utilizată de către orice om care dorește să se informeze la ce oră exactă pleacă autobuzul din stațiile CTP. În figura de mai jos se poate vizualiza conținutul aplicației Ratuc.

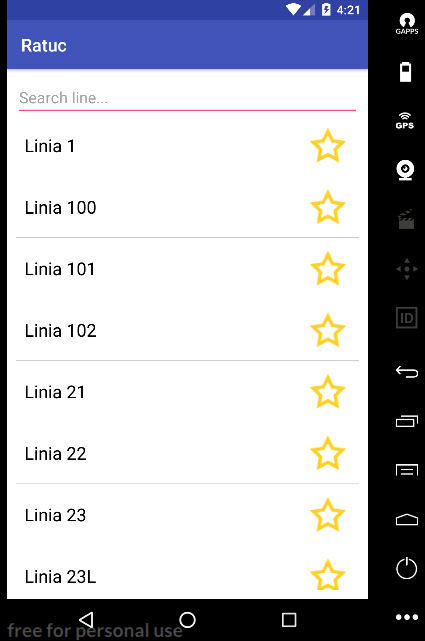


Figura 1. Aplicația Ratuc – Orarul Autobuzelor din Cluj-Napoca

Aplicația are în partea din stângă sus vizibil titlul „Ratuc”, iar apoi are implementat un câmp de search în care se poate căuta orice linie după preferințe sau de asemenea se pot alege ca și favorită, o anumită linie sau chiar și mai multe prin marcarea steluței din dreptul acestora cum se poate observa în cele două imagini de mai jos.

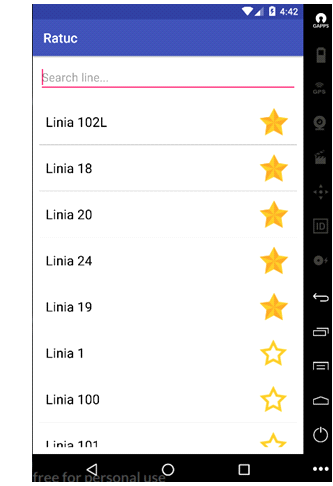
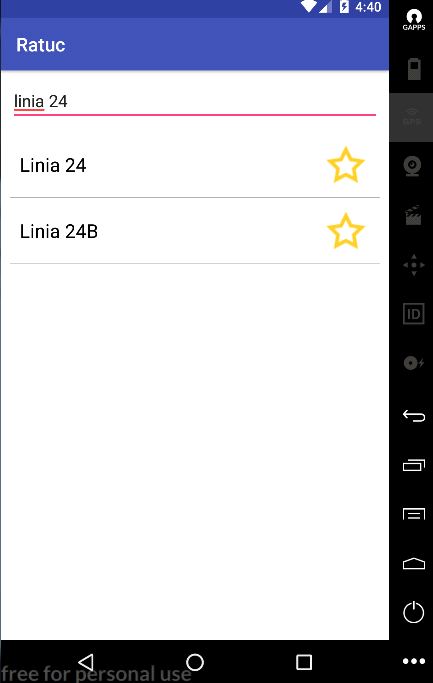


Figura 2. Utilizarea butonului search Figura 3. Utilizarea opțiunii favorit pentru

pentru linia 24 și 24B liniile alese

În AndroidManifest am declarat activitățile iar în Build.Grandle am adăugat importuri și librării.

În MainActivity deschidem ecranul care arată lista cu pdf-urile, setăm listener pentru search și salvăm lista în baza de date când apăsăm pe stea. De asemenea avem un recycleview și un edittext în layout, edittext-ul reprezentând search-ul iar recycleview reprezintă lista. În MainActivity în onCreate luăm view-urile după id și creăm instanța de adapter și atribuim adapter-ul la RecycleView.

În folderul Assets am stocat toate pdf-urile cu orarele pentru fiecare linie iar în MainActivity luăm acea listă de pdf-uri din Assets și creăm o listă de obiecte linie. Un obiect linie are ca și atribute: numele liniei, numele pdf-ului corespunzător și un flag pentru a putea identifica dacă este adăugat la favorite sau dacă nu iar numele liniei il luăm din numele pdf-ului cu metoda getNameFrontAsset din MainActivity.

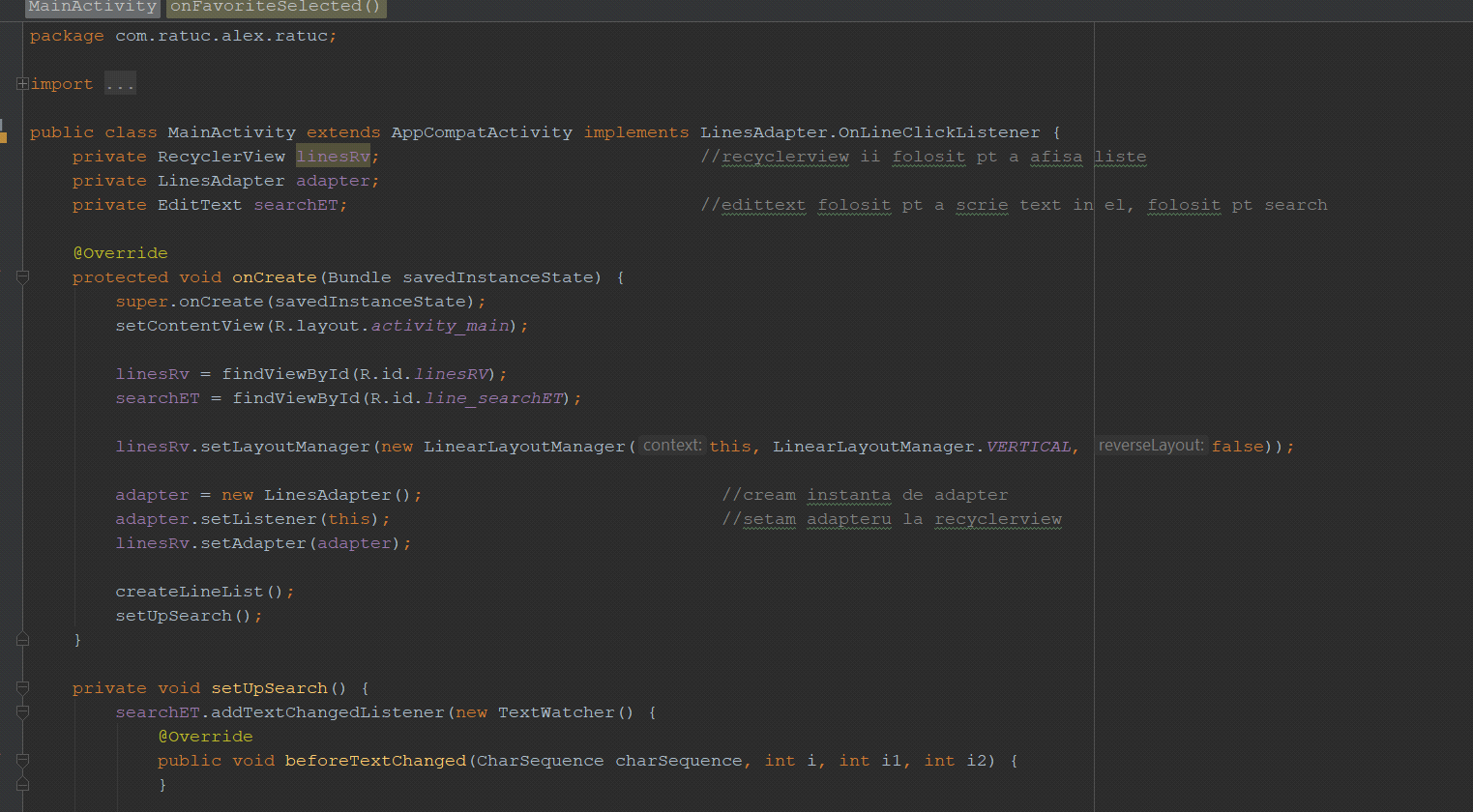


Figura 4. MainActivity

Metoda getNameFromAsset din MainActivity este utilizată pentru a lua numele liniei din numele pdf-ului. Iar linia return assetName.substring(0,assetName.indexOF(”.”)).replace(”\_”,” ”); este împărâțită astfel: assetName = numele fisierului pdf din folder, substring (0,assetName,indexOF(”.”)) se selectează tot string-ul până la „.”. De exemplu pentru numele „Linia\_6.pdf” se va selecta „Linia\_6” deoarece nu afișăm extensia în lista iar cu „.replace(”\_”,” ”);” eliminăm semnul „\_”, practic după această conversie vom putea vedea selectat tot numele de forma „Linia 6”.

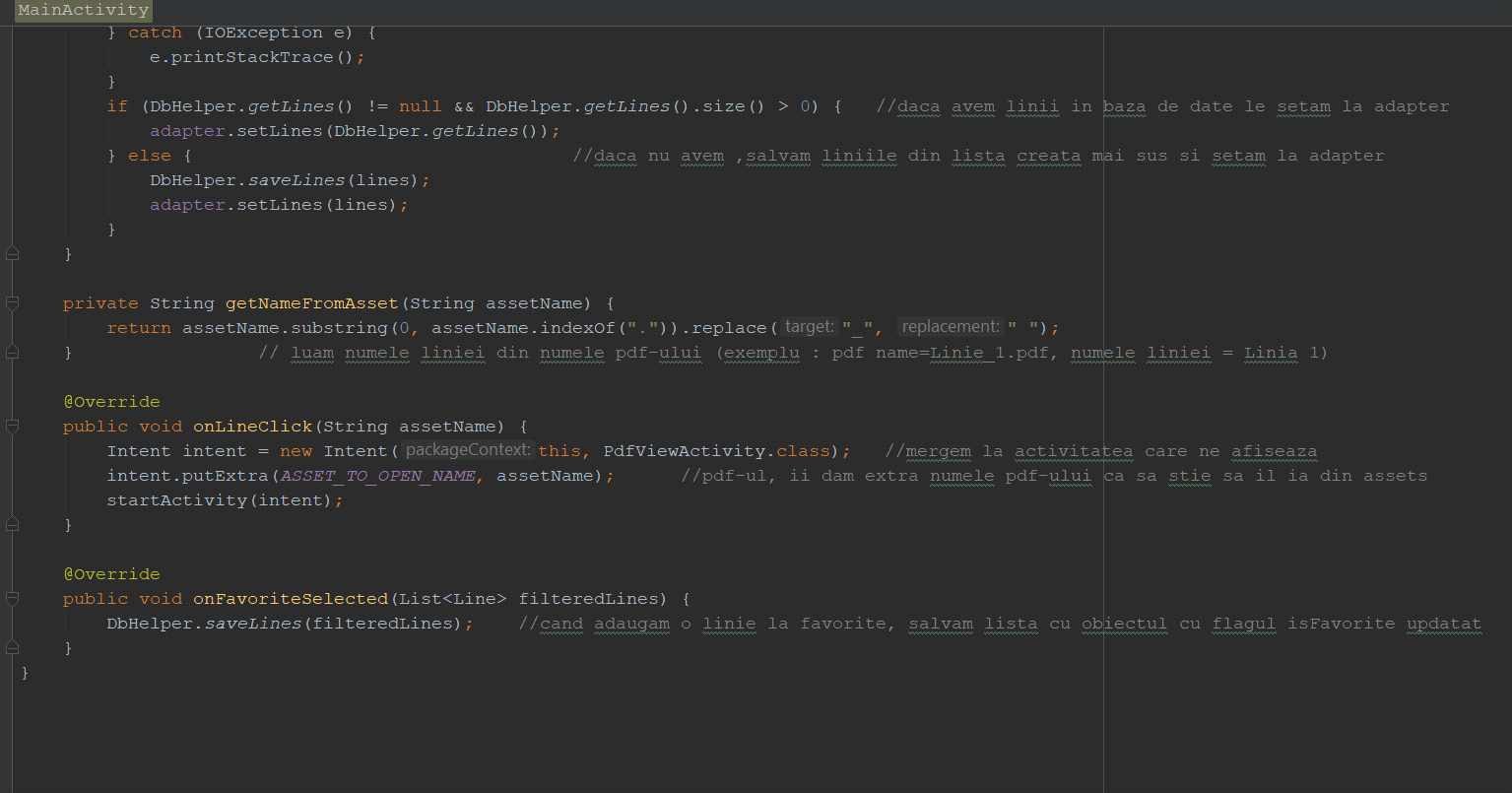


Figura 5. Setarea numelui din titlul pdf-ului

De asemenea în MainActivity am tratat eventurile de click pe linie și click pe stea. De exemplu în metoda onLineClick se intră de fiecare dată când se dă click pe o linie din listă, practic se deschide cea de a doua activitate în care arătam pdf-ul și ii trimitem numele pdf-ului de la linia pe care am dat click. Activitățile au fost deschise prin „intent” „Intent intent=new Intent(this, PdfViewActivity.class)”, unde this reprezintă activatea curentă iar „PdfViewActivity.class” reprezintă activitatea în care vreau să ajung.

LineAdapterul populează lista cu date, filtrează lista după liniile favorite, de asemenea se ocupă de afișarea listei în RecycleView și de eventurile la click, acest lucru se referă la momentul în care un user dă click pe o linie de autobuz sau când dă click pe steaua pentru favorite. Practic există două ecrane, unul dintre ele reprezintă activitatea din MainActivity unde am lista cu autobuzele iar cel de-al doilea ecran este cel în care se deschide pdf-ul cu orarul autobuzului selectat.

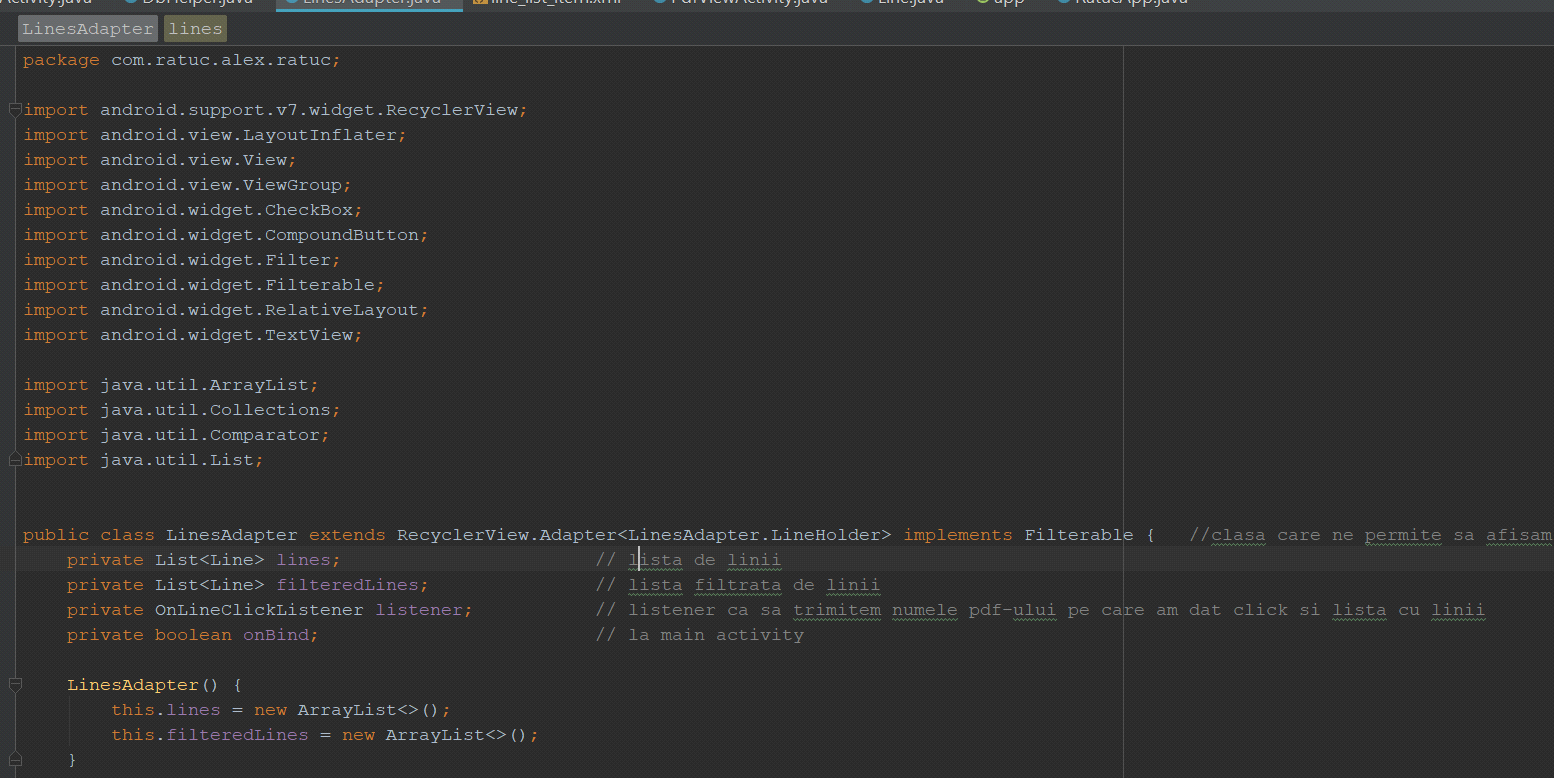


Figura 6. LineAdapter

În LineAdapter facem inflate la layout-ul care reprezinta un rând din listă in metoda onCreateViewHolder. Tot ce se vede grafic în Android reprezintă un view, de exemplu lista este un view, butonul este un view, practic în holder se declară ca variabile toatele elementele de user interface din layout-ul din onCreateViewHolder, acestea alegându-se după un id.

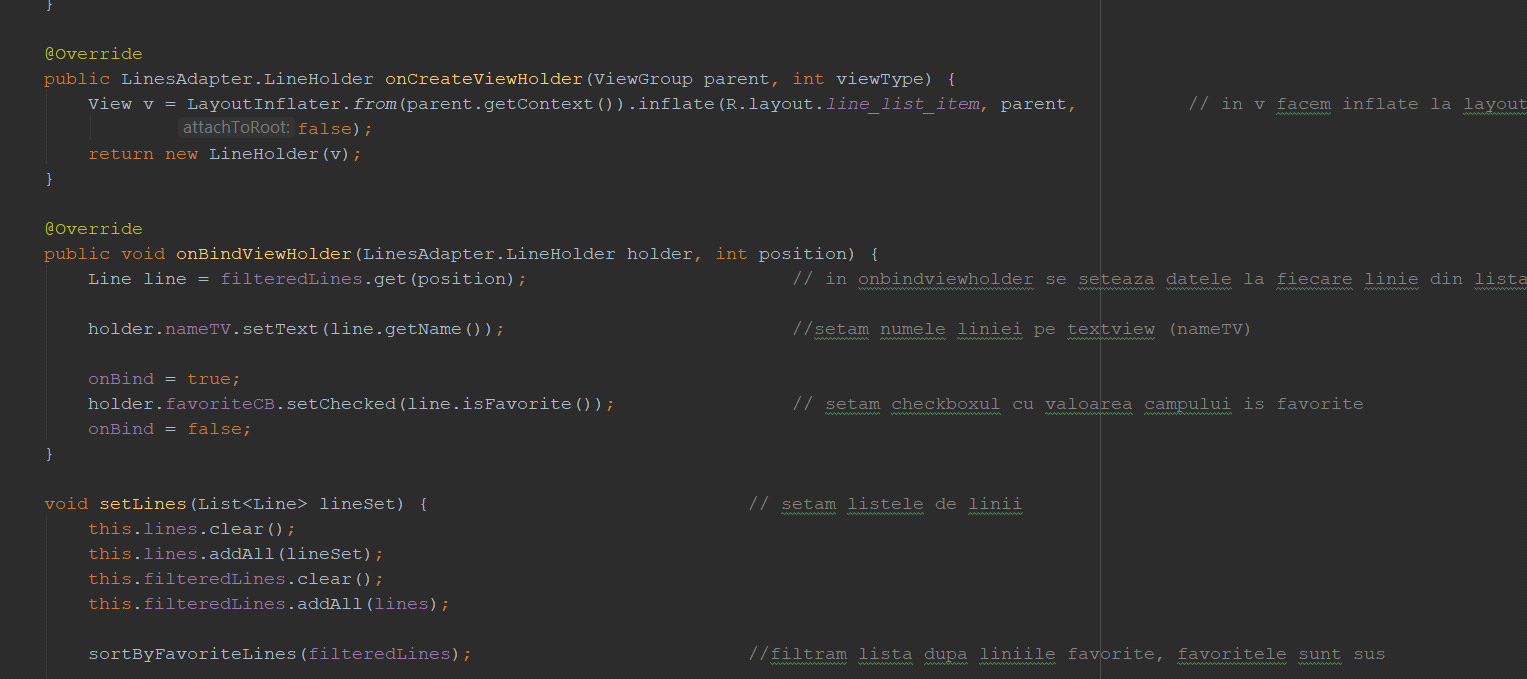


Figura 7. Metoda onBindViewHolder din LineAdapter

Cu aceasta metodă, „onBindViewHolder” setăm datele pe fiecare rând din listă deoarece cu metoda getItemCount se știe câte date vor fi inserate. Returnarea marimei listei este dată de return filteredLines != null ? filteredLines.size() :0;, unde dacă lista ii nulă returnează 0. În holder.nameTV.setText(line.getName()) se seteaza numele unei linii din lista.

Line line = filteredLines.get(position) este folosit pentru a lua obiectul line de la pozitia position. Metoda holder.favoriteCV.setChecked(line.isFavorite()) setează steaua, practic dacă o linie de autobuz este la favorite sau nu.

Avem o metodă performFiltering în care se realizează filtrarea datelor introduse in search iar de exemplu linia if (line.getName().toLowerCase().contains(filterPattern)) ne face filtrarea în search astfel încât dacă o linie conține vreun caracter din stringul de la search, luam linia și o adăugăm în lista de linii filtrate.

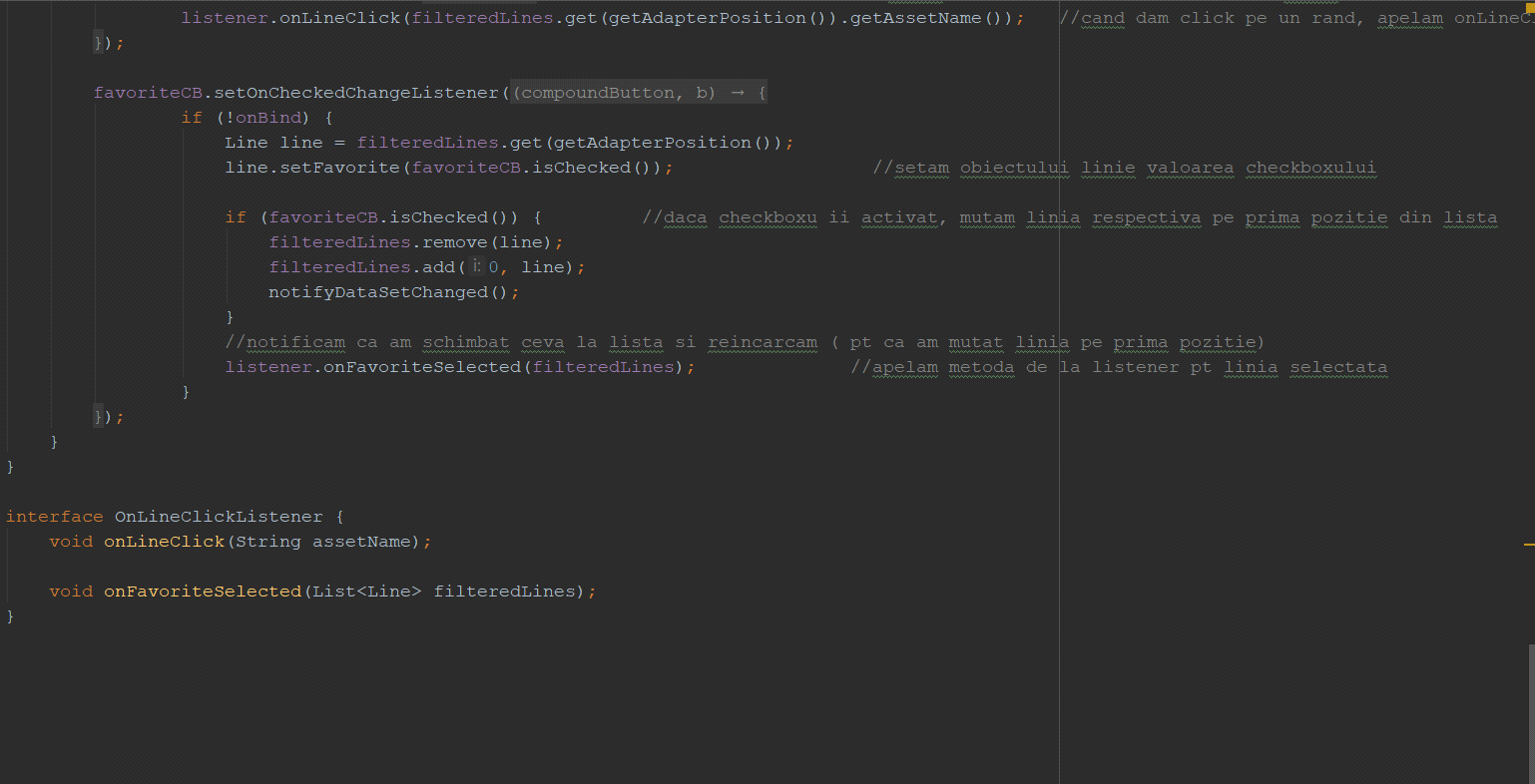


Figura 8. Tratarea butonului search

Click-ul pe stea este tratat în metoda favoriteCB.setOnCheckedChangeListener(newCompoundButton.OnCheckerdChangeListener()); în care practic se verifică poziția liniei pe care doresc să o adaug la favorite iar cu ajutorul lui „if(favoriteCB.isChecked())” se face un check dacă steaua de la o anumită linie este sau nu bifată iar dacă aceasta de exemplu este bifată, șterge din listă obiectul prin „filteredLines.remove(line);” și îl readaugă pe prima poziție în listă prin „filteredLines.add(0,line);” apoi prin „notifyDataSetChanged()” anunțăm că ceva s-a schimbat în listă și că trebuie să se dea un refrash iar prin „listener.OnFavoriteSelected(filteredLines);” trimitem la MainActivity notificare că am adăugat ceva la favorite.

Ca și concluzie se remarcă faptul că aplicația este una accesibilă și usor de utilizat de către orice persoana care dorește să afle ora de la care pleacă un autobuz către Mall, către Centru, către Aeroport sau către orice altă locație din Cluj-Napoca, de asemenea ușurință prin care o linie poate fi căutată sau adăugată la favorite face ca această aplicație să fie una ideală pentru orice utilizator.