**UNIVERSITATEA DE MEDICINĂ, FARMACIE, ȘTIINȚE ȘI**

**TEHNOLOGIE DIN TÂRGU MUREȘ**

**FACULTATEA DE INGINERIE ȘI TEHNOLOGIA**

**INFORMAȚIEI**

**SPECIALIZAREA: INFORMATICĂ**

**PROIECT DE LABORATOR**

Proiect Paypal

Îndrumător științific: Istvan Kiss

Student:Iagăr Catalin-Ionuț

Cuprins

[Capitol 1: Introducere 3](#_Toc100685251)

[Capitol 1.1: Ce este PayPal 3](#_Toc100685252)

[Capitol 1.2: Proiect PayPal 3](#_Toc100685253)

[Capitol 2: Tehnologii folosite 4](#_Toc100685254)

[Capitol 2.1: Partea de client 4](#_Toc100685255)

[Capitol 2.1.1: Android studio 4](#_Toc100685256)

[Capitol 2.1.1: Java 4](#_Toc100685257)

[Capitol 2.2: Partea de server 4](#_Toc100685258)

[Capitol 2.2.1: Visual Studio 2022 4](#_Toc100685259)

[Capitol 2.2.1: .NET 5](#_Toc100685260)

[Capitol 2.2.1: C# 5](#_Toc100685261)

[Capitol 2.2.1: Entity Framework 5](#_Toc100685262)

[Capitol 2.2.1: LINQ 5](#_Toc100685263)

[Capitol 3: Implementare și proiectare 6](#_Toc100685264)

[Capitol 3.1: Proiectarea aplicației 6](#_Toc100685265)

[Capitol 3.2: Implementare server 6](#_Toc100685266)

[Capitol 3.2: Implementare client 7](#_Toc100685267)

[Capitol 3.2.1: MainActivity 7](#_Toc100685268)

[Capitol 3.2.2: LoginActivity 9](#_Toc100685269)

[Capitol 3.3: Conexiune server – client 12](#_Toc100685270)

[Capitolul 4: Bibliografie 13](#_Toc100685271)

# Capitol 1: Introducere

## Capitol 1.1: Ce este PayPal

**PayPal** este un operator al unui sistem de plăți on-line, care poate fi utilizat la achitarea sumelor medii și mici, de exemplu, la cumpărarea și vânzarea în [comerț electronic](https://ro.wikipedia.org/wiki/Comer%C8%9B_electronic" \o "Comerț electronic). Conform propriilor informații, PayPal are mai mult de 192 de milioane de utilizatori activi în peste 200 de piețe, cu posibilitatea plăților în peste 100 de valute (din noiembrie 2016).

Functionalități PayPal:

* Creare cont
* Coectare
* Adaugare sold
* Transfer catre alt utilizator
* Cerere de la alt utilizator
* Istoric tranzactii
* Ș.a.m.d

## Capitol 1.2: Proiect PayPal

Pe parcursul acestui proiect am dorit sa creez o aplicatie de tipul PayPal, in cadrul careia sa exista abilitatea de creare cont nou, conectare la cont, adaugare sold, transfer catre alt utilizator, abilitatea de a schimba parola/email-ul contului respectiv.

# Capitol 2: Tehnologii folosite

## Capitol 2.1: Partea de client

### Capitol 2.1.1: Android studio

**Android Studio** este mediul de dezvoltare integrat (IDE) oficial pentru dezvoltarea de aplicații Android, bazat pe IntelliJ IDEA. Pe lângă editorul de cod puternic și instrumentele de dezvoltare ale IntelliJ, Android Studio oferă și mai multe funcții care vă îmbunătățesc productivitatea atunci când construiți aplicații Android. În aplicația client am folosit limbajul Java, limbajul clasic folosit de Android Studio.

### Capitol 2.1.1: Java

**Java** este un limbaj de programare orientat pe obiecte care produce software pentru mai multe platforme. Când un programator scrie o aplicație Java, codul compilat (cunoscut sub numele de bytecode) rulează pe majoritatea sistemelor de operare (OS), inclusiv Windows, Linux și Mac OS. Java derivă o mare parte din sintaxa sa din limbajele de programare C și C++.

## Capitol 2.2: Partea de server

### Capitol 2.2.1: Visual Studio 2022

**Visual Studio** este un mediu de dezvoltare integrat (IDE) de la Microsoft. Este folosit pentru a dezvolta programe de calculator, precum și site-uri web, aplicații web, servicii web și aplicații mobile. Visual Studio utilizează platforme de dezvoltare software Microsoft, cum ar fi Windows API, Windows Forms, Windows Presentation Foundation, Windows Store și Microsoft Silverlight. Poate produce atât cod nativ, cât și cod gestionat

### Capitol 2.2.1: .NET

**.NET** este o platformă pentru dezvoltatori open source, multiplatformă, gratuită, pentru construirea multor tipuri diferite de aplicații. Cu .NET, se pot utiliza mai multe limbaje de programare, editori și biblioteci pentru a crea aplicatii web, mobile, desktop, jocuri și IoT.

### Capitol 2.2.1: C#

**C#** este un limbaj de programare *strongly-typed*, *orientat pe obiecte*. C# este *open-source*, simplu, modern, flexibil și versatil.

### Capitol 2.2.1: Entity Framework

Entity framework reprezintă un model obiectual pentru interacționarea cu bazele de date Începând cu versiunea 6, aceasta nu mai este integrată în .NET ci vine ca o componentă distinctă Din Junie 2016, Entity Framework (EF) a fost înlocuit cu EF Core, o versiune modernă care vine să sprijine dezvoltarea aplicațiilor orientate pe baze de date din .NET Core. Am utilizat conceptele Code first pentru server și DataBase first pentru Data Viewer.

Instalare: Install-Package EntityFramework

### Capitol 2.2.1: LINQ

**LINQ** definește un set de operatori care permit interogarea, stocarea și traversarea datelor - toate acestea fiind realizate prin intermediul limbajului C#.

Utilizare: Using System.Linq

# Capitol 3: Implementare și proiectare

## Capitol 3.1: Proiectarea aplicației



## Capitol 3.2: Implementare server

Serverul se ocupa cu gestionarea bazei de date si prelucrarea request-urilor de la client, care sunt de forma ”sTipRequest mesaj1 mesaj2 ...”. La primirea string-ului, serverul il imparte in mai multe substring-uri , apoi verifica primul argument pentru a determina ce tip de request este:

* public static String sLoginReq = "login";
* public static String sSignupReq = "signup";
* public static String sAddBalance = "addBalance";
* public static String sTransfer = "transfer";
* public static String sChangePass = "changePass";
* public static String sChangeEmail = "changeEmail";
* public static String sForgotPass = "forgotPass";

Dupa identificarea tipului de request restul argumentelor sunt trimise la o alta metoda (în funcție de tipul de request) dupa care informațiile sunt prelucrate si este trimis un raspuns țnapoi spre client de forma ”cTipRaspuns” urmat de alte argumente țn funcție de caz.

Posibilie raspunsuri in cazul unui signUp:

* public static String cSignupOk = "signup-ok";
* public static String cSignupUserErr = "signup-user-err";
* public static String cSignupEmailErr = "signup-email-err";

Prelucrarea datelor din baza de date se face utilizând EntityFramework, pentru gestionarae bazei de date, și LINQ pentru a efectua diferite operatții asupra datelor.

Exemplu LINQ: Users user = (from u in context.UsersList

where u.Username == tempUsername

select u).First();

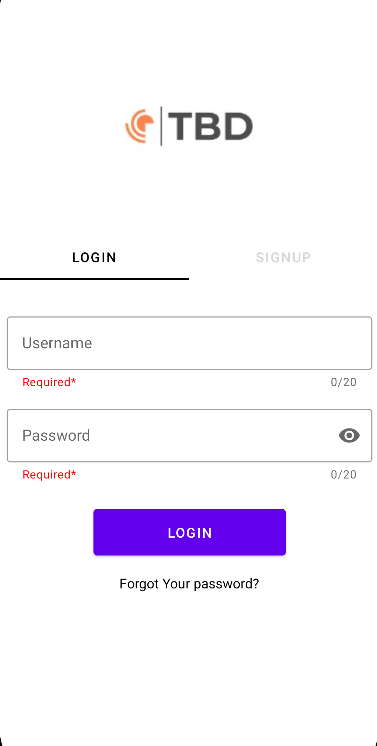
## Capitol 3.2: Implementare client

Implementarea meniurilor a fost realizată folosind un TabLayour, care conține opțiuniile utilizatorului la momentul folositrii aplicației, și un viewPager2, care conține mai multe fragmente în funcție de activitatea în care se află utilizatorul.

### Capitol 3.2.1: MainActivity

Pentru început, utilizatorul se afla în MainActivity, unde sunt două opțiuni, logarea în aplicație în cazul în care utilizatorul are deja un cont creat, sau opțiunea de SignUp, unde se poate crea un cont nou.

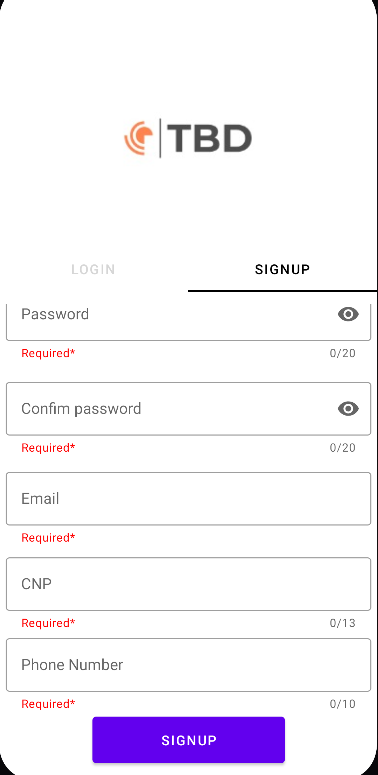
#### Capitolul 3.2.1.1: LoginFragment



În cadrul acestui fragment utilizatorul trebuie să își introduce datele de logare, iar la apasarea butonului de ”LOGIN”, este trimis un request spre server pentru validarea datelor.

De asemenea tot aici, utilizatorul are opțiunea de aș-i recupera parola (WIP), după completarea usernaemului și apăsarea butonuluiu, utilizatorul va primi un mail cu parola contului.

#### Capitolul 3.2.1.2: SignUpFragment

 În cadrul acestui fragment, utilizatorul trebuie sa adauge anumite date, pentru crearea unui cont nou.

Validarea acestor date se face în client, iar în momentul în care datele introduse sunt corecte, textul roșu de sub câmp va dispărea, sau utilizatorul va fi avertizat ca datele introduse nu sunt valide.

La apăsarea butonului de ”SIGNUP”, serverul primește datele respective, iar daca username-ul și email-ul nu sunt utilizate deja, contul va fi creat cu succes, în caz contrar va fi trimis un mesaj de eroare catre client.

Validarea datelor:

* username: cel puțin 8 caractere
* parola: cel puțin 8 caractere, litera mică/mare, număr și caracter special
* email: care trebuie sa fie sub forma (text)@(text).(text)
* CNP: validarea acestuia se face folosind o formula
* Numar telefon: trebuie sa aiba fix 10 numere

Pentru validarea CNP-ului este folosita următoarea formulă:

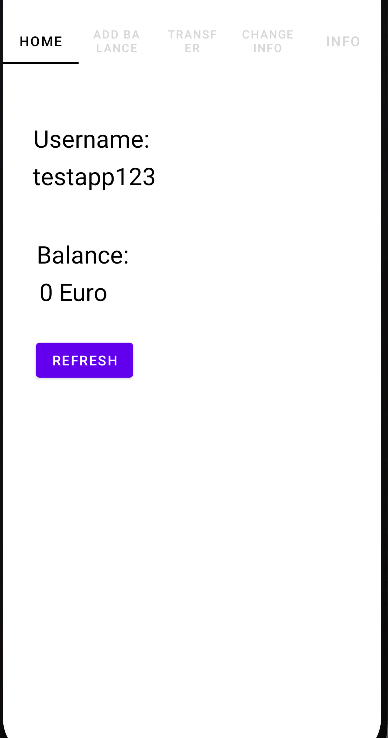
public boolean isCNPValid(String CNP) {  
 String constCNP = "279146358279";  
 int c = 0;  
  
 for (int i = 0; i < 12; i++){  
 String digit = String.*valueOf*(CNP.charAt(i));  
 String testDigit = String.*valueOf*(constCNP.charAt(i));  
 c += (Integer.*parseInt*(digit) \* Integer.*parseInt*(testDigit));  
 }  
  
 int res = c % 11;  
  
 if(res == 10) res = 1;  
  
 if (res != Integer.*parseInt*(String.*valueOf*(CNP.charAt((12))))) return false;  
  
 return true;  
}

Pentru restul verificărilor sunt folosite clasele Pattern și Matcher din Java.

### Capitol 3.2.2: LoginActivity

După ce utilizatorul se conectează, folosind datele de logare, trece în LoginActivity, care este compus din 5 opțiuni: ”Home”, ”Add balance”, ”Transfer”, ”Change Info”, ”Info”,

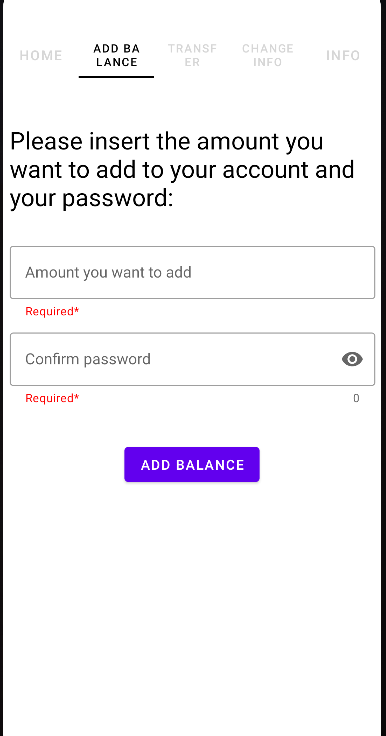
#### Capitolul 3.2.2.1: HomeFragment



În cadrul acestui fragment, utilizatorul poate sa vada numele de utilizator și soldul care îl deține în cont.

De asemenea se găsște și un buton de refresh, în cazul în care utilizatorul adauga sold sau transfer unui alt utilizator.

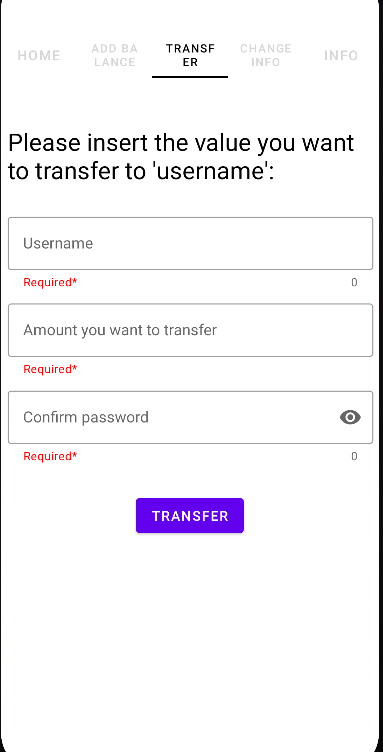
#### Capitolul 3.2.2.2: AddBalanceFragment



În cadrul acestui fragment, după introducerea sumei dorite si a parolei, este trimis un request la server care va adauga suma respectiva în contul utilizatorului.

În cazul în care parola introdusă este greșită, serverul va trimite un raspuns de eroare, urmănd ca utilizatorul sa fie avertizat print-un SnackBar, că datele introduse nu sunt corecte

#### Capitolul 3.2.2.3: TransferFragment

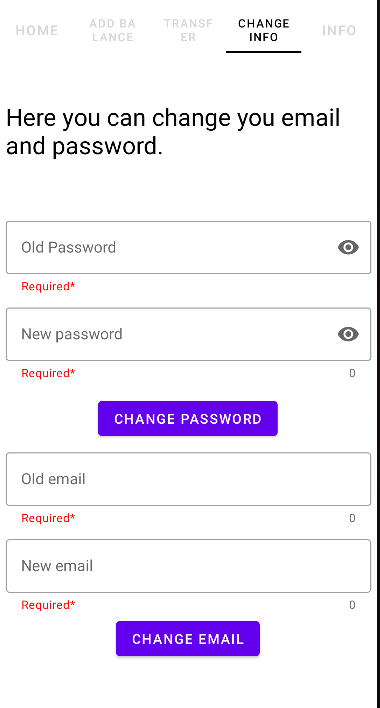


În cadrul acestui fragment, după introducerea destinatarului, sumei dorite si a parolei, este trimis un request la server care va transfera suma respectiva în contul destinatarului.

În cazul în care parola introdusă este greșită, serverul va trimite un raspuns de eroare, urmănd ca utilizatorul sa fie avertizat print-un SnackBar

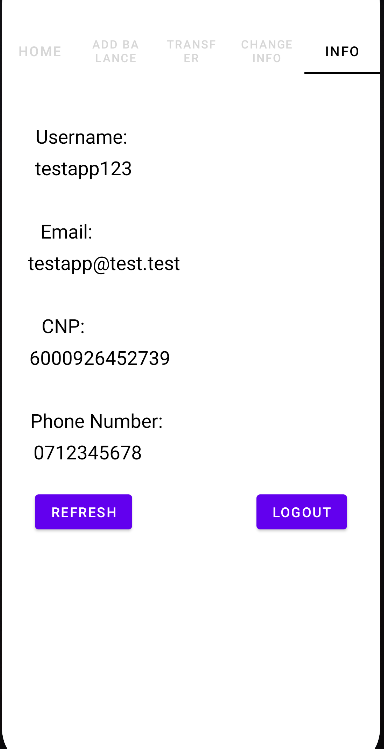
În cazul în care username-ul introdus este greșit, serverul va trimite un raspuns de eroare, urmănd ca utilizatorul sa fie avertizat print-un SnackBar

#### Capitolul 3.2.2.4: ChangeInfoFragment



În cadrul acestui fragment, utilizatorul poate sa schimba parola sau email-ul contului.

#### Capitolul 3.2.2.5: InfoFragment



În cadrul acestui fragment, utilizatorul poate să își vizualizeze informațiile despre cont.

De asemenea tot aici se regăsește un buton de ”LOGOUT”.

## Capitol 3.3: Conexiune server – client

Pentru conexiunea la server sunt folosite mai multe clase de tip Thread, în funcție de operația dorită, de exemplu: LoginThread, SignupThread ș.a.m.d. În interiorul acestor clase este nevoie de ip-ul clientului și de un port, în cazul acesta 8000. Informațiile sunt trimise și primite folosind un PrintWriter si un Buffered Reader.

Conexiunea la server se realizeaza prin intermediul unui socket care primește ca parametrii ip-ul si port-ul.

Pentru a se putea realiza conexiunea intre server și client în rețea este nevoie de urmatoarele două permisiuni:

* <uses-permission android:name="android.permission.INTERNET"/>
* <uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />

# Capitolul 4: Bibliografie

* Documentația Android : <https://developer.android.com/docs>
* View Binding: <https://www.youtube.com/watch?v=omml4lK_b-A>
* EntityFramework: <https://docs.microsoft.com/en-us/ef/>
* Validare CNP: <https://ro.wikipedia.org/wiki/Cod_numeric_personal_(Rom%C3%A2nia)>
* Validare Email: <https://stackoverflow.com/questions/624581/what-is-the-best-java-email-address-validation-method>