

# Documentatie securitatea sistemelor informatice

## ~Password generator~

Mihaela Cismaru grupa 221

### Motivatia dezvoltarii unui astfel de program

În general accesul la conturile personale se face prin intermediul parolelor. Păstrarea secretă a parolelor este foarte importantă în protejarea datelor. Pentru siguranță, parolele se salvează în baza de date după ce au fost transformate într-un hash prin diversi algoritmi deja existenți. Nu este posibil să transformi un hash înapoi în parolă, dar o parolă poate fi transformată în hash și apoi comparată cu hash-ul salvat în baza de date în vederea validării.

Problema intervine în existența dictionarelor de hash-uri, în care parolele comune sunt transformate în hash-uri, iar aceste hash-uri sunt asociate direct sirului de caractere de la baza lor. Astfel hash-ul unei parole comune poate fi decodat foarte ușor cu ajutorul unui dictionar online. Alegerea unei parole puternice scade probabilitatea hash-ului aferent de a se afla într-un astfel de dictionar deoarece șansele ca alt utilizator să o folosească sunt foarte mici.

Aplicația implementată își propune să genereze siruri aleatoare de caractere care pot fi folosite drept parole puternice. Șansa ca aceste siruri să se afle într-un dictionar nu este 0 dar este foarte mică, ceea ce este idealul în securitate deoarece nimic nu este 100% sigur.

### Interfata grafică și descrierea modului de funcționare

Pentru acest proiect am ales implementarea unei aplicații desktop cu o singură fereastră ce încorporează întreaga funcționalitate. În fereastră regăsim niste text de ghidaj, 7 câmpuri pentru personalizarea parolelor, un buton și o secțiune unde vor fi afișate parolele generate.

O parola poate contine 3 categorii de caractere: alfabetice (litere mari sau mici din alfabetul latin), numerice (cifre de la 0 la 9) si speciale ( unul din urmatoarele: !#\$%&()\*+,-./:;?@[ ]^\_`{|} ).

Here! Have a password!

Genereaza parole sigure acum!

Caractere alfabetice: MINIM MAXIM

Caractere numerice: MINIM MAXIM

Caractere speciale: MINIM MAXIM

TOTAL CARACTERE 10

GENEREAZA PAROLE

#lhK76FZ74  
zyhu2Ym28P  
891270635x  
PB8MO2wJNW  
462th05r7n  
2a22NI7uqe  
G42YD0I8\$P  
5z65Q:3623  
T3h%8Q098x  
142jEz2LZz  
0xZQ56s9kf  
/1(G9D1xJo  
LmhQ28wWp3  
2TT6d72&5G  
44i60Cd2I7  
Lc01E3V9P5  
OD354VvhBv  
M4U77Ne?59  
xZ230v3Sat  
7955307.3h  
287IS4C55E  
2L0206dJk9

Pentru fiecare categorie de caractere avem 2 campuri ce pot fi completate, minimul de astfel de caractere si maximul. Aceste campuri pot fi completate caz in care parolele pot contine maxim atatea caractere de acest fel sau minim atatea caractere de acest fel sau pot fi lasate libere caz in care programul tine cont doar de campul completat sau de niciunul. In cazul de mai sus niciun camp nu este completat asa ca programul va genera parole fara sa tina cont de cate caractere din fiecare tip exista.

De exemplu daca alegem sa fie minim 10 caractere alfabetice, parolele vor avea doar caractere alfabetice.

Here! Have a password!

Genereaza parole sigure acum!

Caractere alfabetice: MINIM 10 MAXIM

Caractere numerice: MINIM MAXIM

Caractere speciale: MINIM MAXIM

TOTAL CARACTERE 10

GENEREAZA PAROLE

LbHHGEhRwt  
QFHeKNIyWJ  
ZmviatWpYZ  
wNrvLGsFHI  
MqlaotamZy  
KWlyWmrPSb  
PUVpPSPaTE  
MKdTZHiPZo  
VwLrxwXEVy  
DBFOYYfJXg  
MCwIkoOhYB  
vJluXHpkrm  
LNHjHIFeAn  
FjPrPjaaRS  
ognaGzvWke  
miPPvrdjMC  
DKPQCwcDKz  
CVHPwDJST  
algKnfaiRG  
aCuHglwgrZ  
zaejQgfeyV  
sCDdFhHl

Daca alegem sa avem 0 caractere alfabetice programul va genera parole doar din caractere numerice sau speciale.

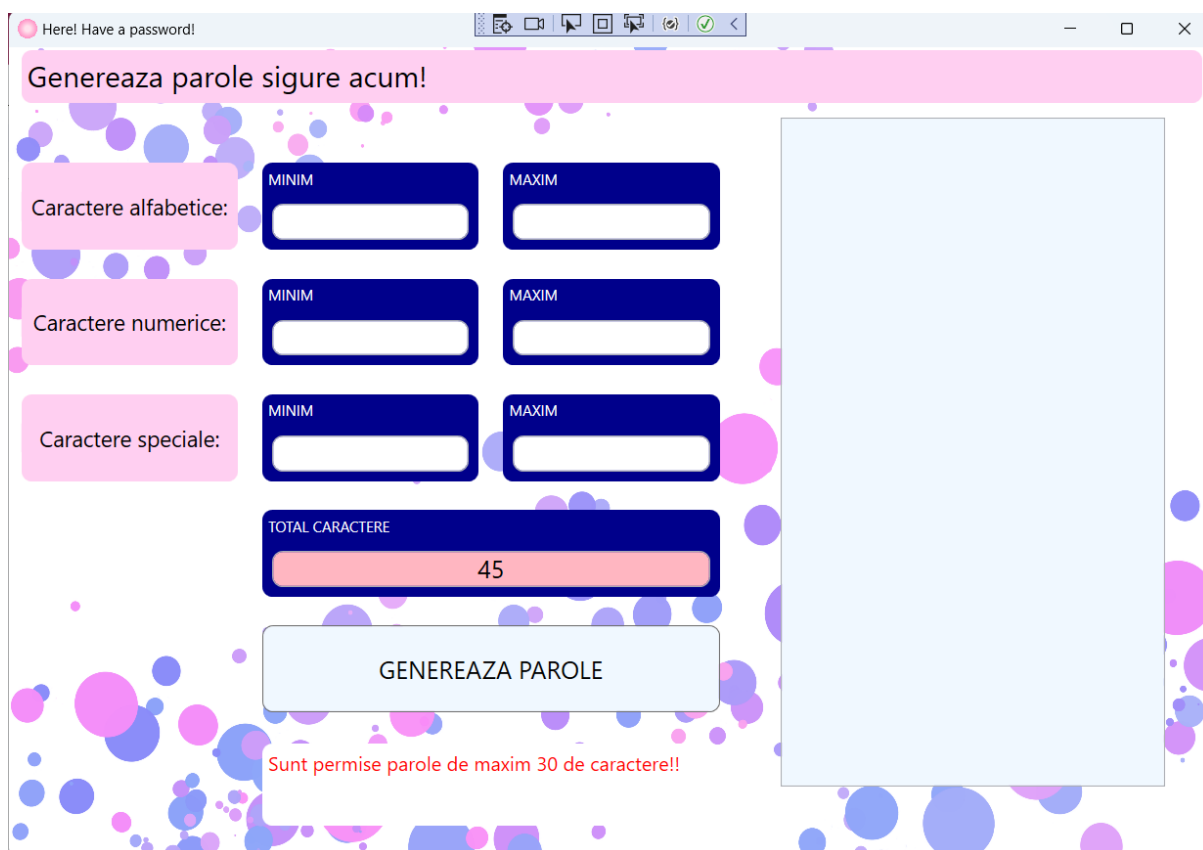
De asemenea vom avea campul foarte important numit : "Total caractere" ce fixeaza lungimea parolei dorite. La lansarea programului acesta are valoarea default de 10 caractere. Aceasta poate fi modificata insa poate avea valoarea cuprinsa doar intre 6 si 30 de caractere.

In exemplu putem remarca parole generate cu lungimea de fix 19 caractere. Pentru a genera parolele este necesara completarea campurilor cu date corecte si apasarea butonului : "genereaza parole". Parolele vor fi afisate in lista ce dispune de optiunea

de scroll în partea din dreapta a ferestrei. Numarul de parole generate va fi intotdeauna de exact 50 de parole ce pot fi copiate usor din lista si folosite. Aceste parole nu sunt salvate nicaieri si vor fi pierdute imediat ce butonul este apasat din nou sau aplicatia este inchisa.

## Validarea valorilor din campuri

În capitolul anterior am menționat că este permis un total de caractere cu valoarea cuprinsă între 6 și 30 de caractere. Dar desigur au fost implementate mai multe astfel de validări pentru buna funcționare a programului și prevenirea erorilor majore. Pentru asta am implementat următorul comportament:



The screenshot shows a web browser window titled "Here! Have a password!". The page has a pink header with the text "Genereaza parole sigure acum!". Below the header, there are three rows of input fields for password requirements, each with a "MINIM" and "MAXIM" field. The first row is for "Caractere alfabetice:", the second for "Caractere numerice:", and the third for "Caractere speciale:". Below these is a "TOTAL CARACTERE" field showing the value "45". A large blue button labeled "GENEREAZA PAROLE" is positioned below the total field. A red error message at the bottom states "Sunt permise parole de maxim 30 de caractere!!". The background of the page is decorated with a pattern of pink and blue circles.

Atunci când sunt introduse date gresite de exemplu aici numărul total de caractere depășește 30. În momentul apăsării butonului lista de parole este eliberată dar nu este completată cu parole noi, sub butonul de generare apare un mesaj în care este explicat ce este gresit cu datele introduse iar campurile ce contin date eronate sunt colorate într-un roz deschis.

Alte validări implementate sunt:

- Se permite introducerea a maxim 2 cifre sau a spațiilor albe. Spațiile albe sunt permise deoarece programul accepta lasarea acestor libere, iar un spațiu alb este greu de remarcat iar utilizatorul nu va înțelege ce a completat gresit.

- Numarul maxim de caractere de un fel va fi obligatoriu mai mare sau egal cu numarul minim de caractere de acelasi fel (in cazul in care ambele valori sunt completate).
- Numarul minim de caractere dorit (cumulate din toate cele 3 campuri) nu poate fi mai mare decat numarul total de caractere setat.

The screenshot shows a web application titled "Here! Have a password!". The main heading is "Genereaza parole sigure acum!". There are three input sections: "Caractere alfabetice:", "Caractere numerice:", and "Caractere speciale:". Each section has "MINIM" and "MAXIM" input fields. The "Caractere alfabetice:" section has "MINIM" set to 4 and "MAXIM" set to 3. The "Caractere numerice:" and "Caractere speciale:" sections have empty "MINIM" and "MAXIM" fields. Below these is a "TOTAL CARACTERE" field set to 10. A "GENEREAZA PAROLE" button is at the bottom. A red error message at the bottom right states: "Numarul minim de caractere de un fel nu poate fi mai mare decat maximul de caractere de acelasi fel".

The screenshot shows the same web application. In this state, the "Caractere alfabetice:" section has "MINIM" set to 4 and "MAXIM" set to 10. The "Caractere numerice:" section has "MINIM" set to 10 and "MAXIM" set to 10. The "Caractere speciale:" section has empty "MINIM" and "MAXIM" fields. The "TOTAL CARACTERE" field is still set to 10. The "GENEREAZA PAROLE" button is at the bottom. A red error message at the bottom left states: "Numarul de caractere minim este mai mare decat numarul total de caractere dorit!!!".

- Numarul maxim de caractere dorit (cumulate din toate cele 3 campuri) nu poate fi mai mic decat numarul total de caractere setat. (Acest caz se intampla doar cand toate cele 3 campuri de maxim au valoare, altfel numarul de caractere ramas poate fi satisfacut din caracterele ce nu au maxim setat).

## Algoritmul de generare a parolelor

Pentru a genera parole pe baza datelor introduse folosesc un algoritm de generare aleatoare a caracterelor. Acesta incepe de la alegerea unui seed, seed ul ales este ora de rulare de la care folosim inclusiv milisecundele. Acest seed este ales o data per apasarea butonul de generare si este folosit pentru toate 50 de parole dintr o serie in modul descris in continuare.

Cream un string din datele ce formeaza ora noastra amestecate si undeva in acest string introducem path ul catre o poza nefolosita din proiectul nostru.

```
private String GenerateSeed()
{
    var currentTime = DateTime.Now;
    string scrambledDate = currentTime.Month.ToString() + currentTime.Day.ToString() + currentTime.Hour.ToString() + currentTime.Year.ToString() + currentTime.Minute.ToString();
    string fileName = "SpecialImage.png";
    scrambledDate += System.IO.Path.GetFullPath(fileName).ToString();
    scrambledDate += currentTime.Ticks.ToString();

    SHA256 mySHA256 = SHA256.Create();
    var hashedSeed = mySHA256.ComputeHash(Encoding.UTF8.GetBytes(scrambledDate));

    StringBuilder seed = new StringBuilder();
    foreach (byte b in hashedSeed)
    {
        seed.Append(b.ToString("x2"));
    }

    string resultedString = "";
    foreach (char ch in seed.ToString())
    {
        resultedString += CharToBinary((char)ch);
    }

    return resultedString;
}
```

Convertim acest string intr-un hash prin intermediul algoritmul de hashing Sha-256. Luam toate caracterele din acest hash si il convertim in binar. Acest sir va fi principala unealta de generat numere random.

Pentru fiecare numar aleator cerut aplicam lfsr de 22 de ori pe acest string si obtinem un sir de 22 de biti pe care il transformam intr-un numar. Sirul ramane modificat dupa fiecare aplicare a lfsr si este folosit mai departe pentru cand avem nevoie de alt numar aleator.

De 50 de ori generam o parola noua si le adaugam in lista ce va fi afisata in interfata noastra grafica.

```
public List<string> GetPasswords(int totalCharacters, int alphabetMin, int alphabetMax, int numericMin, int numericMax, int specialMin, int specialMax)
{
    var theLfsrString = GenerateSeed();
    List<string> passwordList = new List<string>();

    for (int i = 0; i < 50; i++)
    {
        var receivedTuple = GetOnePassword(totalCharacters, alphabetMin, alphabetMax, numericMin, numericMax, specialMin, specialMax, theLfsrString);
        var password = receivedTuple.Item1;
        theLfsrString = receivedTuple.Item2;
        passwordList.Add(password);
    }

    return passwordList;
}
```

Pentru fiecare parola adaugam numarul minim de caractere din fiecare categorie. Generam un numar random prin procedeul descris mai sus si facem modulo numarul de caractere disponibil din fiecare timp. Pentru caractere numerice generam un plus un numar ce va determina daca va fi litera mare sau mica. Dupa ce avem numarul de caractere minim deja general. Generam aleator caractere ce se vor incadra in maximele stabilite. Atunci cand generam aceste caractere aleator am ales sa am o probabilitate de 1/20 caracter

special, 9/20 caracter numeric si 10/20 caracter alphabetic. Aceasta metoda este foarte mare si nu am adaugat un screen shot.

Dupa ce am generat caracterele din care va fi formata metoda le voi amesteca tot prin intermediul generarii unui numar random.

```
1 reference
private Tuple<string, string> ShufflePassword(string password, string theLfsrString)
{
    var shuffledPassword = "";
    var initialLenght = password.Length;

    for (int i = 0; i < initialLenght; i++)
    {
        var receivedTuple = GetRandomNumber(theLfsrString);
        var number = receivedTuple.Item1 % password.Length;
        theLfsrString = receivedTuple.Item2;
        shuffledPassword += password[number];
        password = password.Remove(number, 1);
    }

    return Tuple.Create(shuffledPassword, theLfsrString);
}
```

Dupa dupa acest procedeu, aceasta este parola finala si o vom adauga in lista.

- Screen shot functia lfsr

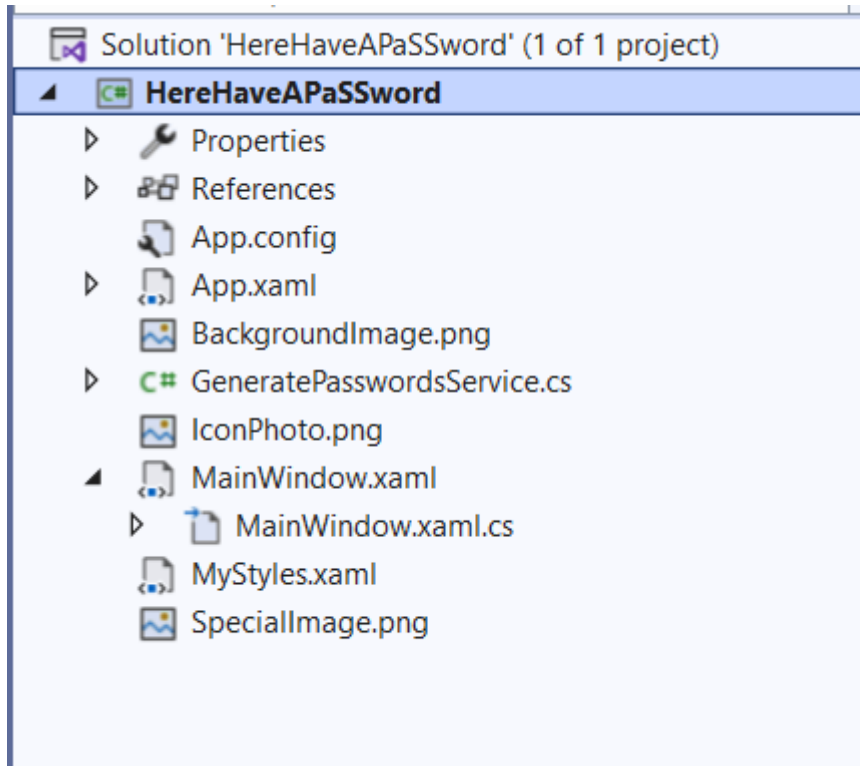
```
14 | 1 reference
15 | private Tuple<string, string> LFSR(string myArray)
16 | {
17 |     string generatedArray = "";
18 |     while (true)
19 |     {
20 |         generatedArray += myArray[myArray.Length - 1];
21 |
22 |         if (generatedArray.Length == 22)
23 |         {
24 |             break;
25 |         }
26 |
27 |         char rezXor = MultipleXOR(myArray);
28 |
29 |         myArray = myArray.Remove(myArray.Length - 1, 1);
30 |         myArray = rezXor + myArray;
31 |     }
32 |     return Tuple.Create(generatedArray, myArray);
33 | }
```

- Screen shot generare numar random

```
168 |
169 | 10 references
170 | private Tuple<int, string> GetRandomNumber(string theLfsrString)
171 | {
172 |     var receivedTuple = LFSR(theLfsrString);
173 |     var generateRandomBytes = receivedTuple.Item1;
174 |     theLfsrString = receivedTuple.Item2;
175 |
176 |     int number = 0;
177 |     for (int i = 0; i < 22; i++)
178 |     {
179 |         number *= 2;
180 |         if (generateRandomBytes[i] == '1') { number++; }
181 |     }
182 |     return Tuple.Create(number, theLfsrString);
183 | }
184 | 1 reference
```

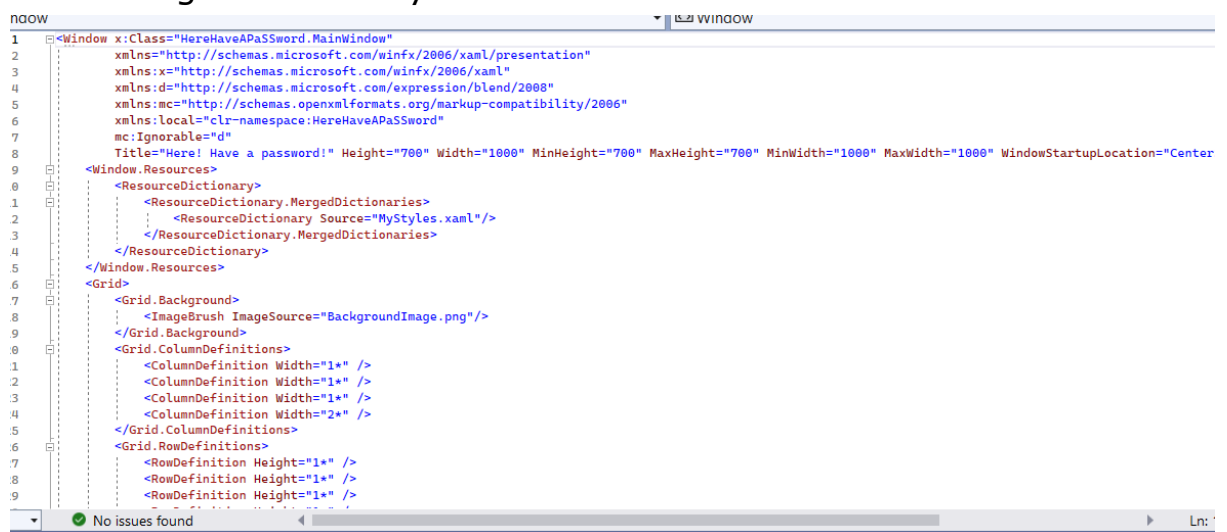
## Tehnologia si structura implementarii

Am ales pentru implementarea proiectului o aplicatie WPF ce foloseste C# ca limbaj de backend si XAML ca limbaj de front end. Acestea apartin tehnologiei .NET de la Microsoft.



In proiectul mereu regasim urmatoare fisiere scrise de mine:

- MainWindow.xaml in care este implementata structura interfeței grafice si a layoutului.





- MainWindow.xaml.cs fisierul c# aferent in care este implementata logica de UI a proiectului. Aici este implementata toata logica de validare a datelor din interfata grafica si modul de functionare a interactiunii cu userul.

```

76 var numericMin = string.IsNullOrEmpty(NumericMin.Text) ? -1 : Int32.Parse(NumericMin.Text.Replace(" ", ""));
77 var numericMax = string.IsNullOrEmpty(NumericMax.Text) ? -1 : Int32.Parse(NumericMax.Text.Replace(" ", ""));
78 var specialMin = string.IsNullOrEmpty(SpecialMin.Text) ? -1 : Int32.Parse(SpecialMin.Text.Replace(" ", ""));
79 var specialMax = string.IsNullOrEmpty(SpecialMax.Text) ? -1 : Int32.Parse(SpecialMax.Text.Replace(" ", ""));
80 var totalCharacters = Int32.Parse(TotalCharacters.Text.Replace(" ", ""));
81
82 if (!ValidateNumbers(totalCharacters, alphabetMin, alphabetMax, numericMin, numericMax, specialMin, specialMax))
83 {
84     return;
85 }
86
87 PasswordList.ItemsSource = _generatePasswordsService.GetPasswords(totalCharacters, alphabetMin, alphabetMax, numericMin, numericMax, specialMin, specialMax);
88
89 }
90
91 private bool ValidateNumbers(int totalCharacters, int alphabetMin, int alphabetMax, int numericMin, int numericMax, int specialMin, int specialMax)
92 {
93     if (totalCharacters > 30 || totalCharacters < 6)
94     {
95         TotalCharacters.Background = Brushes.LightPink;
96         ErrorLabel.Background = Brushes.White;
97         ErrorTextBlock.Text = "Sunt permise parole de minim 6 si maxim 30 de caractere!!!";
98         return false;
99     }
100
101     if (specialMax != -1 && alphabetMax != -1 && numericMax != -1 && specialMax + alphabetMax + numericMax < totalCharacters)
102     {
103         AlphabetMax.Background = Brushes.LightPink;
104         NumericMax.Background = Brushes.LightPink;
105         SpecialMax.Background = Brushes.LightPink;
106         ErrorLabel.Background = Brushes.White;
107         ErrorTextBlock.Text = "Numarul de caractere maxim este mai mic decat numarul total de caractere dorite!!!";
108         return false;
109     }
110
111     bool caractereMinime = true;
112     if (specialMin != -1 && alphabetMin != -1 && numericMin != -1 && specialMin + alphabetMin + numericMin > totalCharacters)

```

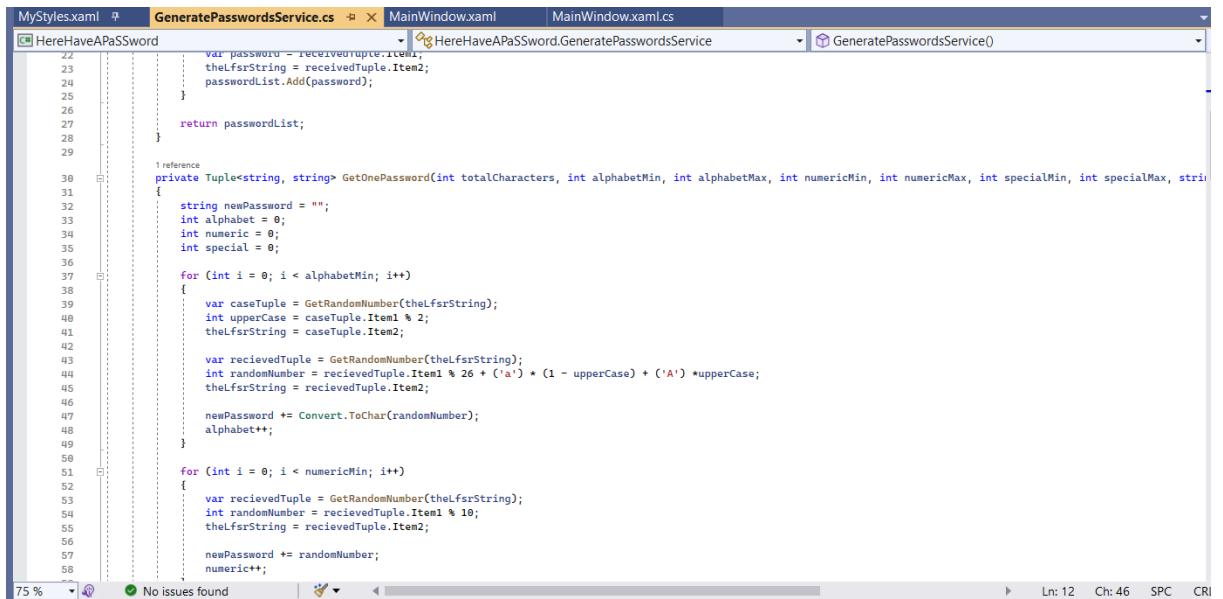
- MyStyles.xaml folosit pentru a configura designul aplicatiei si aspectul fiecarei componente, dar si cateva setari de comportament a acestora.

```

1 <ResourceDictionary xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
2     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">
3
4     <Style TargetType="TextBlock" x:Key="DescriptionTextBlock">
5         <Setter Property="Margin" Value="10 0 10 0"/>
6         <Setter Property="FontSize" Value="18"/>
7         <Setter Property="Background" Value="#fff1f1" />
8         <Setter Property="TextBlock.TextAlignment" Value="Center"/>
9     </Style>
10
11     <Style TargetType="{x:Type Border}">
12         <Setter Property="BorderThickness" Value="8"/>
13     </Style>
14
15     <Style TargetType="Label" x:Key="DescriptionLabel">
16         <Setter Property="Margin" Value="10 0 10 0"/>
17         <Setter Property="FontSize" Value="18"/>
18         <Setter Property="Background" Value="#fff1f1" />
19         <Setter Property="TextBlock.TextAlignment" Value="Center"/>
20     </Style>
21
22     <Style TargetType="{x:Type Border}">
23         <Setter Property="CornerRadius" Value="8"/>
24     </Style>
25
26     <Style TargetType="Label" x:Key="TitleLabel">
27         <Setter Property="FontSize" Value="24"/>
28         <Setter Property="Background" Value="#fff1f1" />
29         <Setter Property="TextBlock.TextAlignment" Value="Center" />
30         <Setter Property="Margin" Value="10 2 10 2"/>
31     </Style>
32
33     <Style TargetType="{x:Type Border}">
34         <Setter Property="CornerRadius" Value="8"/>
35     </Style>
36

```

- GeneratePasswordsService.cs clasa unde este creata lista de parole



```

22     var password = receivedTuple.Item1;
23     theLfsrString = receivedTuple.Item2;
24     passwordList.Add(password);
25 }
26
27     return passwordList;
28 }
29
30     1 reference
31     private Tuple<string, string> GetOnePassword(int totalCharacters, int alphabetMin, int alphabetMax, int numericMin, int numericMax, int specialMin, int specialMax, string theLfsrString)
32     {
33         string newPassword = "";
34         int alphabet = 0;
35         int numeric = 0;
36         int special = 0;
37
38         for (int i = 0; i < alphabetMin; i++)
39         {
40             var caseTuple = GetRandomNumber(theLfsrString);
41             int upperCase = caseTuple.Item1 % 2;
42             theLfsrString = caseTuple.Item2;
43
44             var recievedTuple = GetRandomNumber(theLfsrString);
45             int randomNumber = recievedTuple.Item1 % 26 + ('a') * (1 - upperCase) + ('A') * upperCase;
46             theLfsrString = recievedTuple.Item2;
47
48             newPassword += Convert.ToChar(randomNumber);
49             alphabet++;
50         }
51
52         for (int i = 0; i < numericMin; i++)
53         {
54             var recievedTuple = GetRandomNumber(theLfsrString);
55             int randomNumber = recievedTuple.Item1 % 10;
56             theLfsrString = recievedTuple.Item2;
57
58             newPassword += randomNumber;
59             numeric++;
60         }
61
62         for (int i = 0; i < specialMin; i++)
63         {
64             var recievedTuple = GetRandomNumber(theLfsrString);
65             int randomNumber = recievedTuple.Item1 % 32;
66             theLfsrString = recievedTuple.Item2;
67
68             newPassword += Convert.ToChar(randomNumber);
69             special++;
70         }
71
72         return Tuple.Create(newPassword, theLfsrString);
73     }
74 }

```

- BackgroundImage.png imaginea cu buline de pe fundalul ferestrei.
- SpecialImage.png o imagine alba a carei absolute path este folosita in crearea seedului.