«Creating a Mobile System for the functionality of a currency exchange office»

Anna Zhdanovich 06/24/20
GITHUB - https://github.com/aa-collab

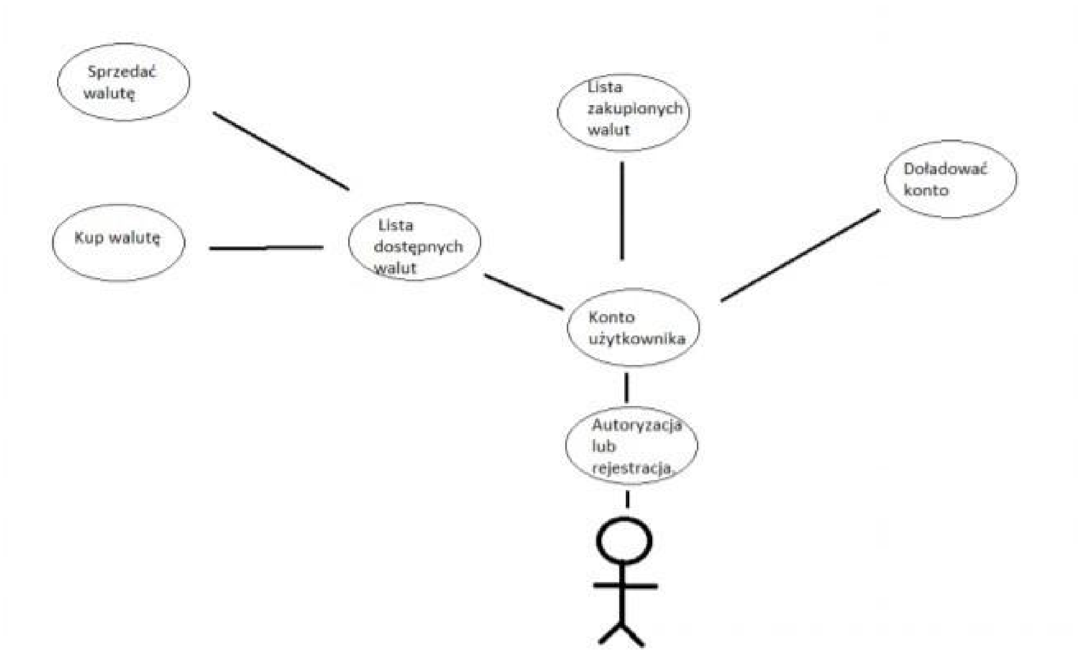
Project goal: Implementation of the office application. Project requirements:
1.Functional requirements:

- **possibility to create a foreign currency account;**
- **the ability to check current and archival exchange rates using the API of the National Bank of Poland; • the ability to check exchange rates, regardless of whether the user is logged in or not; • the ability to purchase / sell currency;**
- **possibility of topping up a foreign currency account;**

2. Non-functional requirements:

- **loading time from the website and database is no longer than 5 seconds;**
- **the application must be supported on devices with OS, Android, IOS.**

Case diagrams:

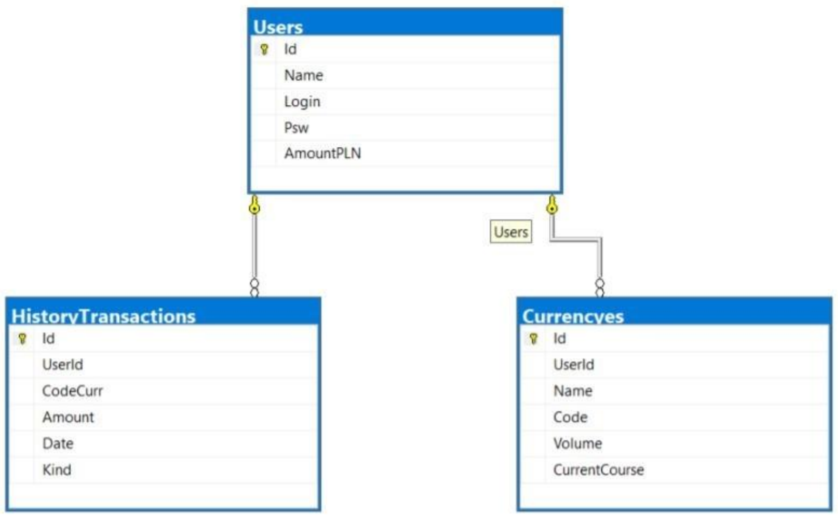


Pic.1 Use case diagrams

Use case diagram – graphic presentation of use cases, actors and relationships betweenthem occurring in a given subject area. Diagram UML use cases are used for modeling system functionality.

Data base

LokalDb was used in the system

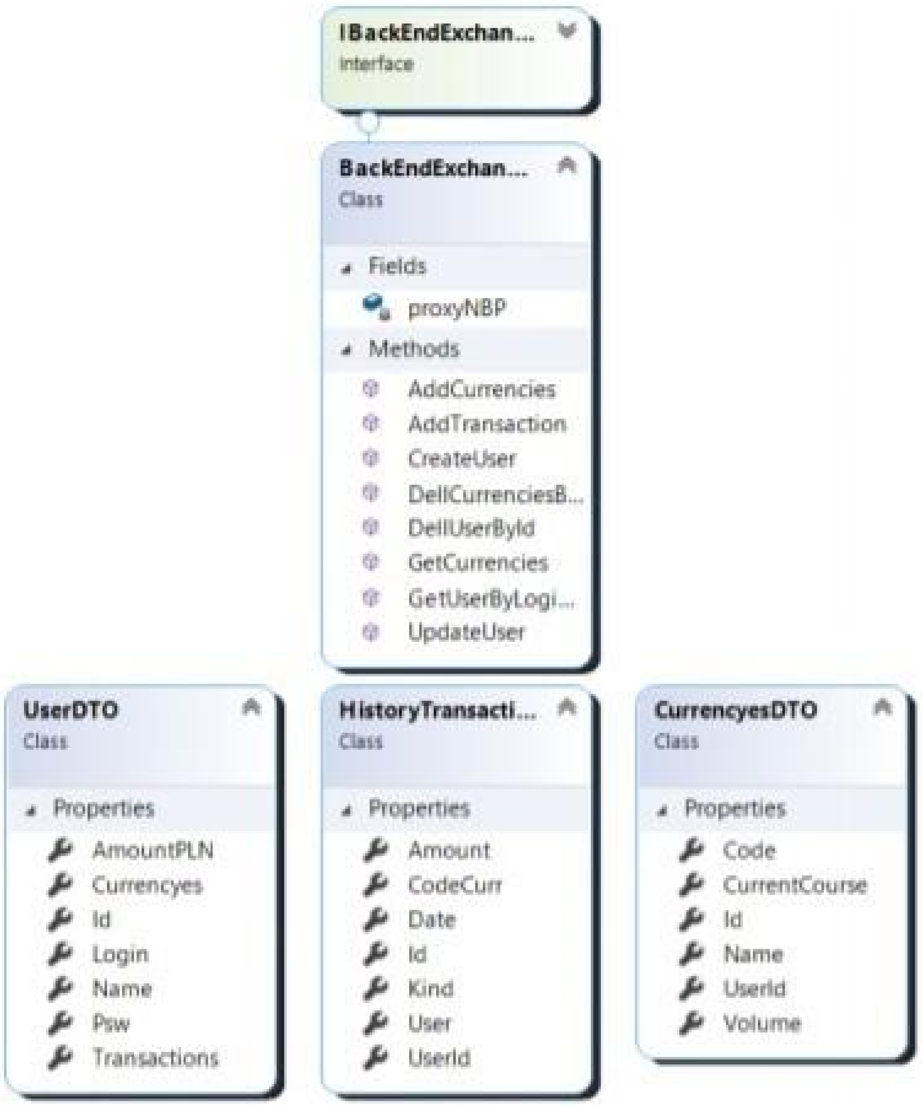


Pic.2 Database Diagrams

The project consists of modules

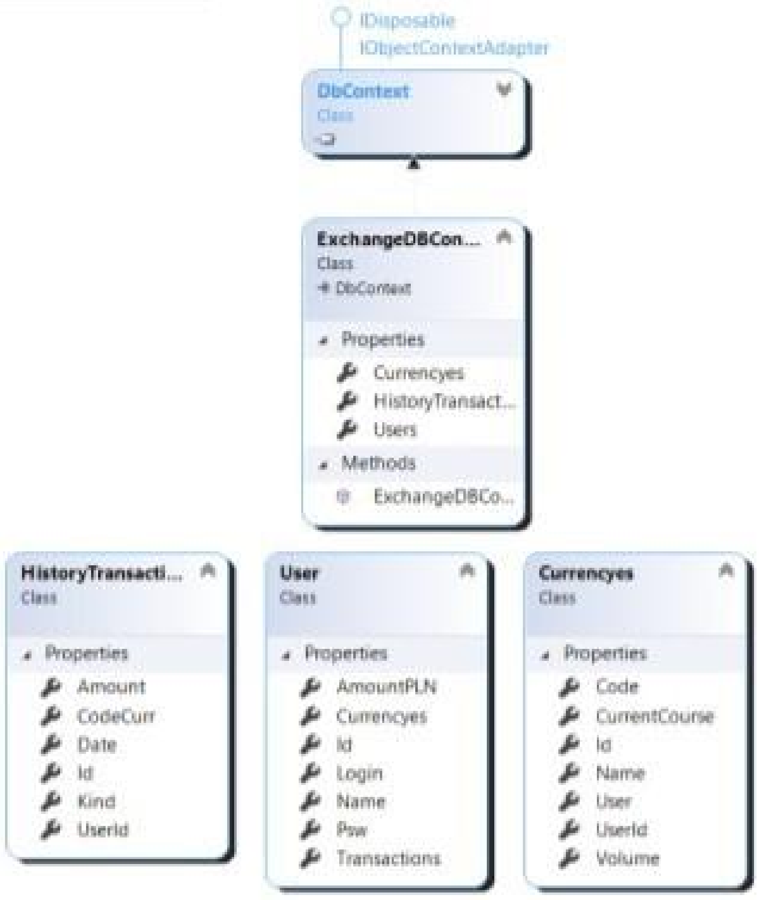
1. BackEndExchangeWCF
2. ExchengeMobileAppClient

ExchengeMobileAppClient



Pic.3 Diagram Class WCF Service and DTO

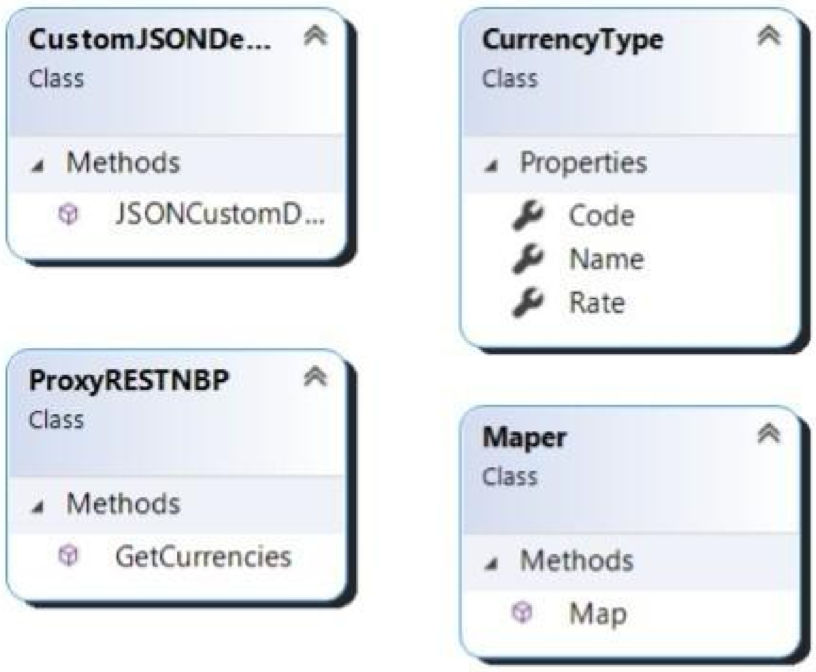
Diagram ORM



Pic.4 Diagram ORM

In the project I used Entity Framework, ORM.
For database management and creation.

Diagram klass working with NBP API



Pic.5 Diagram working with NBP API

To deserialize the JSON response received from the NBP REST API JSONCustomDeserialisator was used:

```
  public class CustomJSONDeserializator
    {

        public static List<CurrencyType> JSONCustomDeserialisator(string JSON)
        {
          List<CurrencyType> currencies = new List<CurrencyType>();
          int lenghtOfstring = JSON.IndexOf("]}]") - JSON.IndexOf(":[") - 3;
          string[] res = JSON.Substring(JSON.IndexOf(":[") + 3, lenghtOfstring).Split(new
string[] { "},{" }, StringSplitOptions.None);

          foreach(string str in res){
            string[] values = str.Split(','); // result like "currency":"yuan renminbi (Chiny)" x3
            var Name = values[0].Split(':')[1].Trim(new char[] { '"'});
            var Code = values[1].Split(':')[1].Trim(new char[] { '"' });
            var Rate = values[2].Split(':')[1].Trim(new char[] { '"' });

            double doubleRate = 0;
            double.TryParse(Rate, out doubleRate); // Parse double with dot
```

```
                currencies.Add(
                    new CurrencyType()
                    {
                        Name = Name,
                        Code = Code,
                        Rate = doubleRate
                    }); ;

            }

            return currencies;
        }
    }
```

Data collection from NBP API:

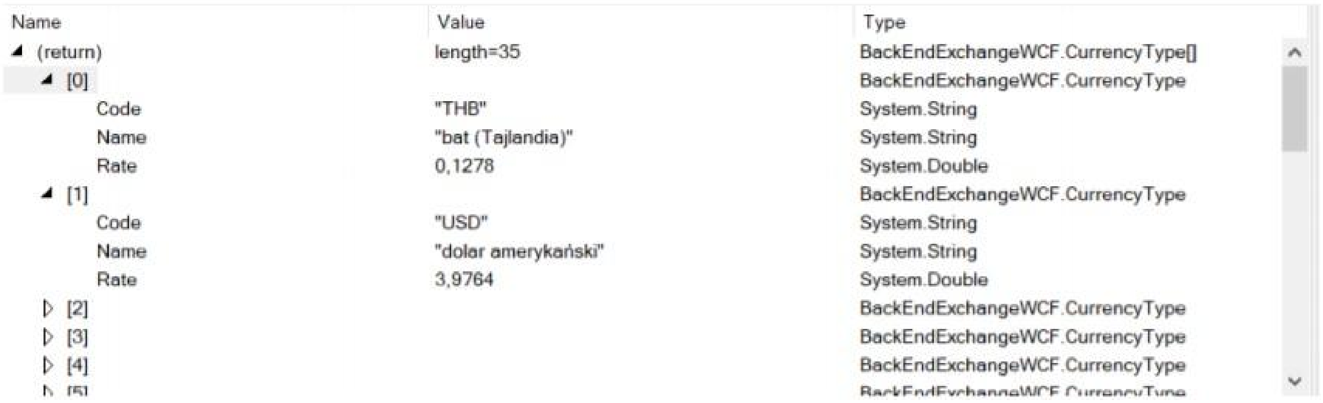
```
public class ProxyRESTNBP
    {
        public List<CurrencyType> GetCurrencies()
        {
            List<CurrencyType> currencies = new List<CurrencyType>();
            var client = new HttpClient();
            Task<HttpResponseMessage> task =
client.GetAsync("http://api.nbp.pl/api/exchangerates/tables/a/");
            if (task.Result.IsSuccessStatusCode)
            {
                var res = task.Result.Content.ReadAsStringAsync().Result;
                currencies = CustomJSONDeserializator.JSONCustomDeserialisator(res);
            }


            return currencies;
        }
    }
```

System testing with WCF Test Client programs embedded in Visual Studio:

WCF Service includes the method:
- GetCurrencies
- CreateUser
- UpdateUser
- DellUserById
- GetUserByLoginAndPsw
- AddCurrencies

-DellCurrenciesById
- AddTransaction

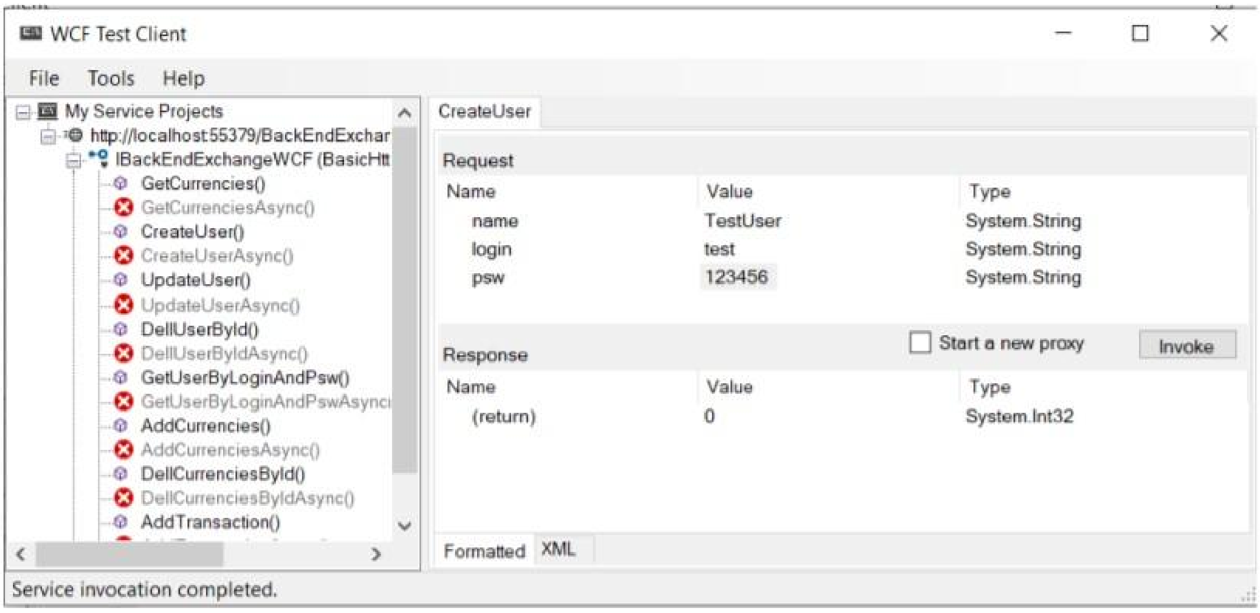
GetCurrencies



Name	Value	Type
◢ (return)	length=35	BackEndExchangeWCF.CurrencyType[]
◢ [0]		BackEndExchangeWCF.CurrencyType
Code	"THB"	System.String
Name	"bat (Tajlandia)"	System.String
Rate	0,1278	System.Double
◢ [1]		BackEndExchangeWCF.CurrencyType
Code	"USD"	System.String
Name	"dolar amerykański"	System.String
Rate	3,9764	System.Double
▷ [2]		BackEndExchangeWCF.CurrencyType
▷ [3]		BackEndExchangeWCF.CurrencyType
▷ [4]		BackEndExchangeWCF.CurrencyType
▷ [5]		BackEndExchangeWCF.CurrencyType

Pic. 6 Testing GetCurrencies

CreateUser



Request

Name	Value	Type
name	TestUser	System.String
login	test	System.String
psw	123456	System.String

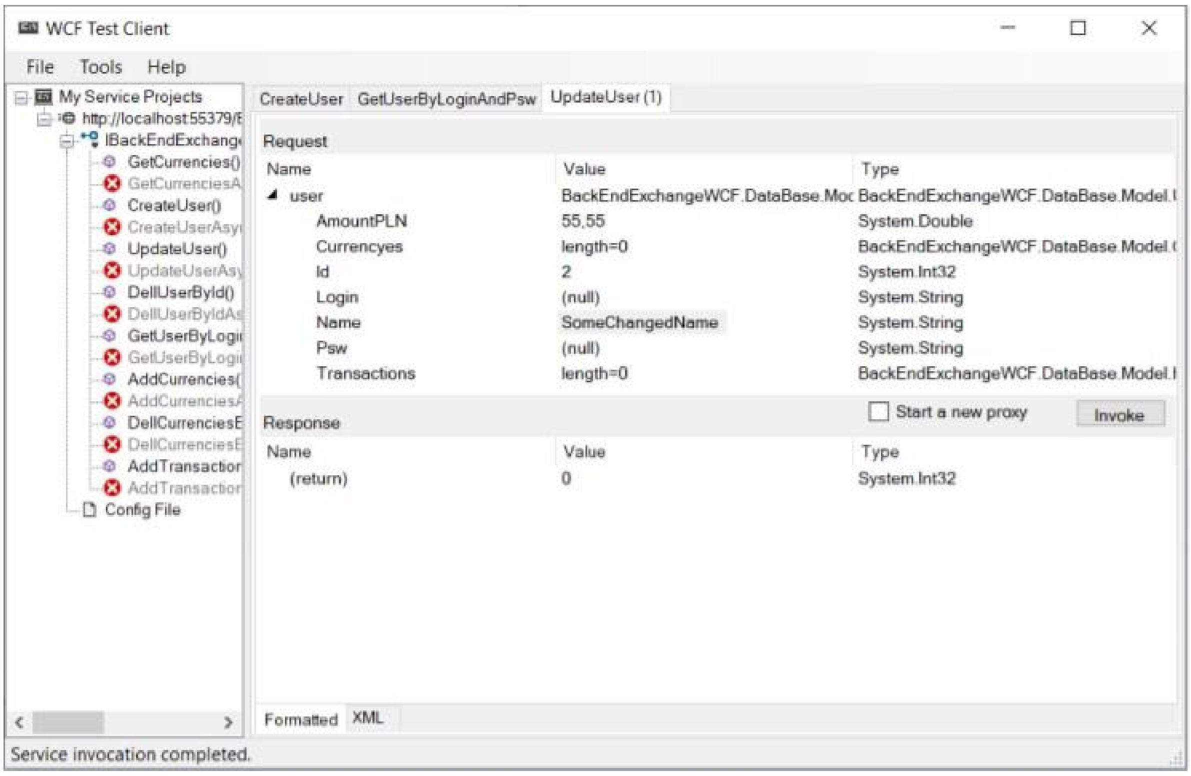
Response

Name	Value	Type
(return)	0	System.Int32

Service invocation completed.

Pic. 7 Testing CreateUser

UpdateUser



Request

Name	Value	Type
◢ user	BackEndExchangeWCF.DataBase.Moc	BackEndExchangeWCF.DataBase.Model.I
AmountPLN	55,55	System.Double
Currencyes	length=0	BackEndExchangeWCF.DataBase.Model.(
Id	2	System.Int32
Login	(null)	System.String
Name	SomeChangedName	System.String
Psw	(null)	System.String
Transactions	length=0	BackEndExchangeWCF.DataBase.Model.I

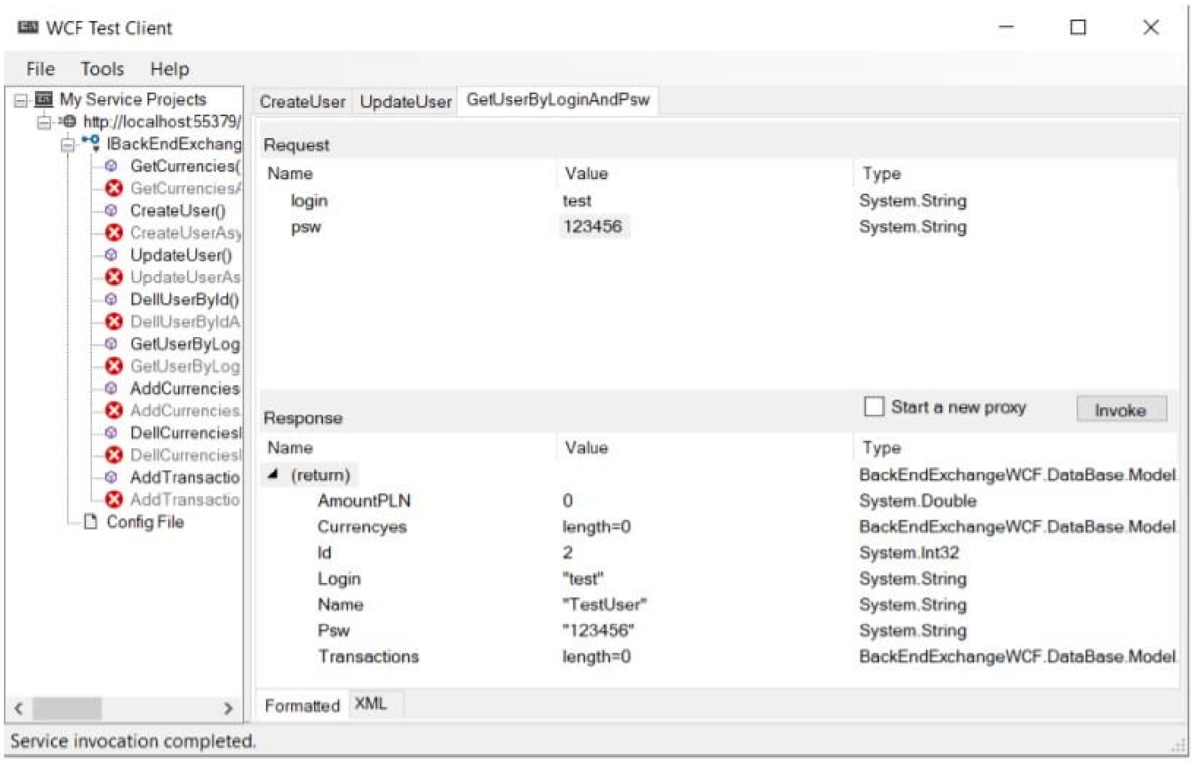
Response

Name	Value	Type
(return)	0	System.Int32

Service invocation completed.

Pic. 8 Testing UpdateUser

GetUserByLoginAndPsw



Request

Name	Value	Type
login	test	System.String
psw	123456	System.String

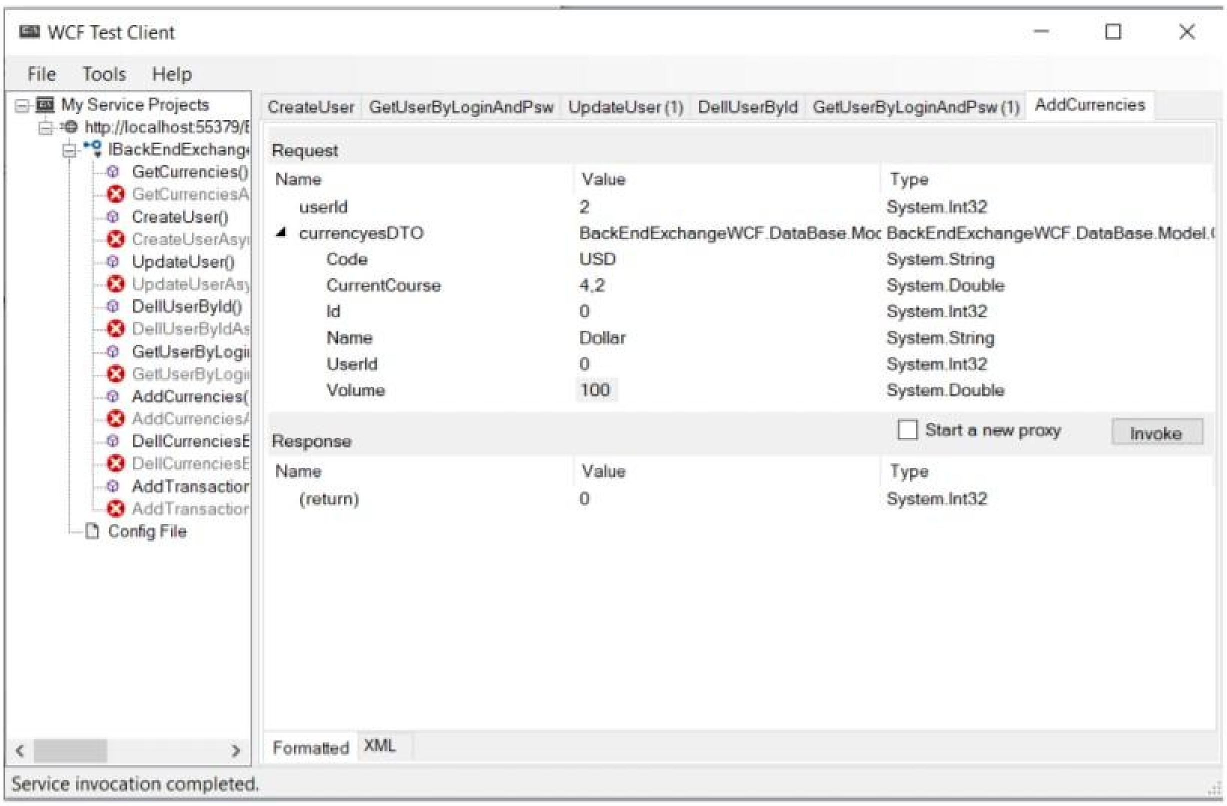
Response

Name	Value	Type
◢ (return)		BackEndExchangeWCF.DataBase.Model.
AmountPLN	0	System.Double
Currencyes	length=0	BackEndExchangeWCF.DataBase.Model.
Id	2	System.Int32
Login	"test"	System.String
Name	"TestUser"	System.String
Psw	"123456"	System.String
Transactions	length=0	BackEndExchangeWCF.DataBase.Model.

Service invocation completed.

Pic. 9 Testing GetUserByLoginAndPsw

AddCurrencies



Request

Name	Value	Type
userId	2	System.Int32
◢ currencyesDTO	BackEndExchangeWCF.DataBase.Moc	BackEndExchangeWCF.DataBase.Model.(
Code	USD	System.String
CurrentCourse	4,2	System.Double
Id	0	System.Int32
Name	Dollar	System.String
UserId	0	System.Int32
Volume	100	System.Double

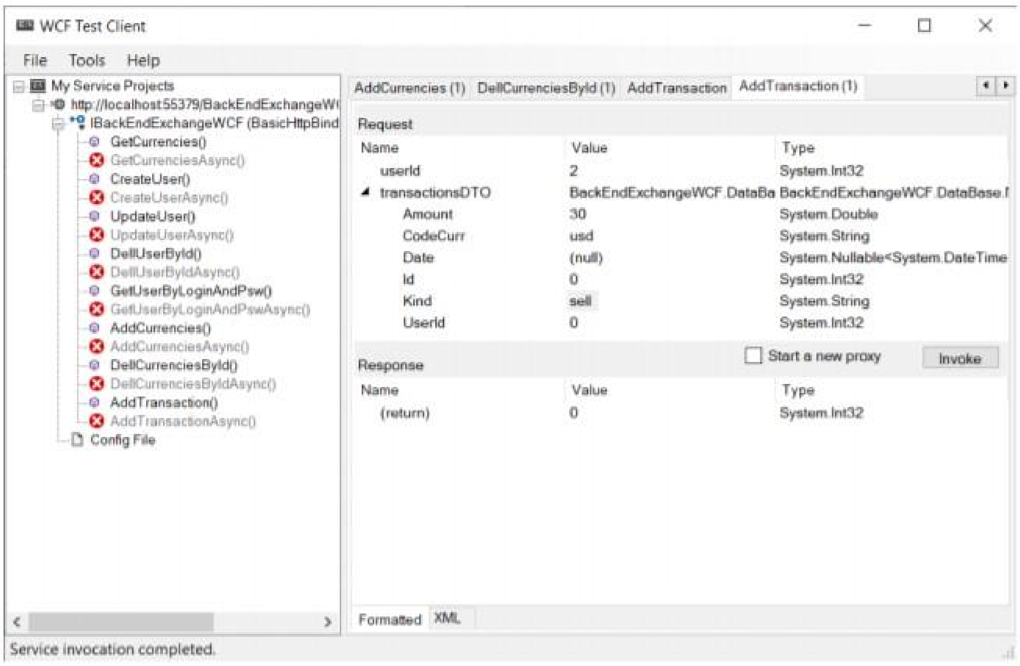
Response

Name	Value	Type
(return)	0	System.Int32

Service invocation completed.

Pic. 10 Testing AddCurrencies

AddTransaction



Request

Name	Value	Type
userId	2	System.Int32
◢ transactionsDTO	BackEndExchangeWCF.DataBa	BackEndExchangeWCF.DataBase.I
Amount	30	System.Double
CodeCurr	usd	System.String
Date	(null)	System.Nullable<System.DateTime
Id	0	System.Int32
Kind	sell	System.String
UserId	0	System.Int32

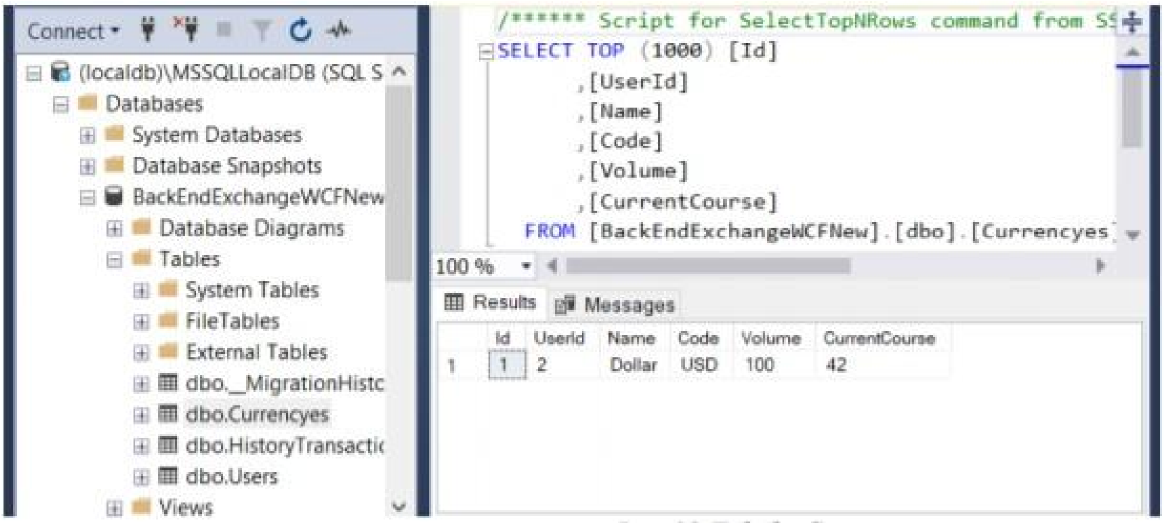
Response

Name	Value	Type
(return)	0	System.Int32

Service invocation completed.

Pic. 11 Testing AddCurrencies

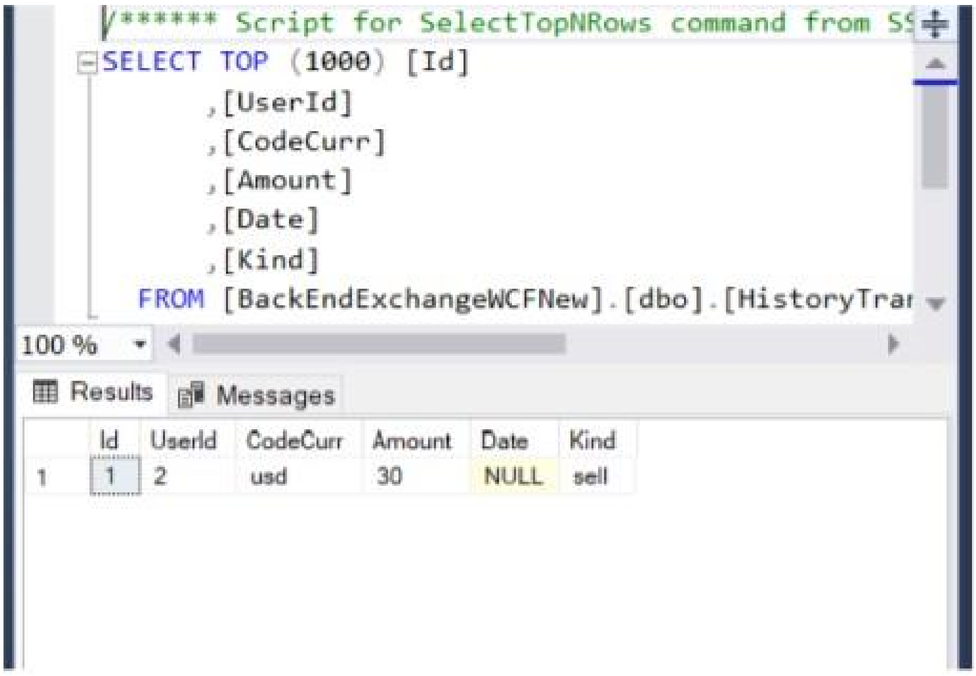
Checking entries in the Database:



```
/****** Script for SelectTopNRows command from SS
SELECT TOP (1000) [Id]
      ,[UserId]
      ,[Name]
      ,[Code]
      ,[Volume]
      ,[CurrentCourse]
  FROM [BackEndExchangeWCFNew].[dbo].[Currencyes]
```

	Id	UserId	Name	Code	Volume	CurrentCourse
1	1	2	Dollar	USD	100	42

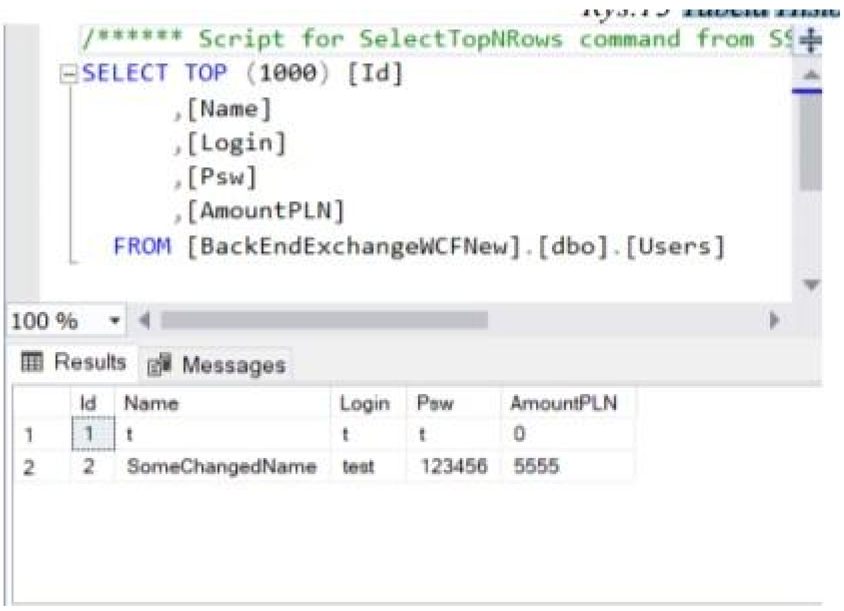
Pic. 12 Currencies table



```
/****** Script for SelectTopNRows command from SS
SELECT TOP (1000) [Id]
      ,[UserId]
      ,[CodeCurr]
      ,[Amount]
      ,[Date]
      ,[Kind]
  FROM [BackEndExchangeWCFNew].[dbo].[HistoryTrar
```

	Id	UserId	CodeCurr	Amount	Date	Kind
1	1	2	usd	30	NULL	sell

Pic. 13 History Transactions table

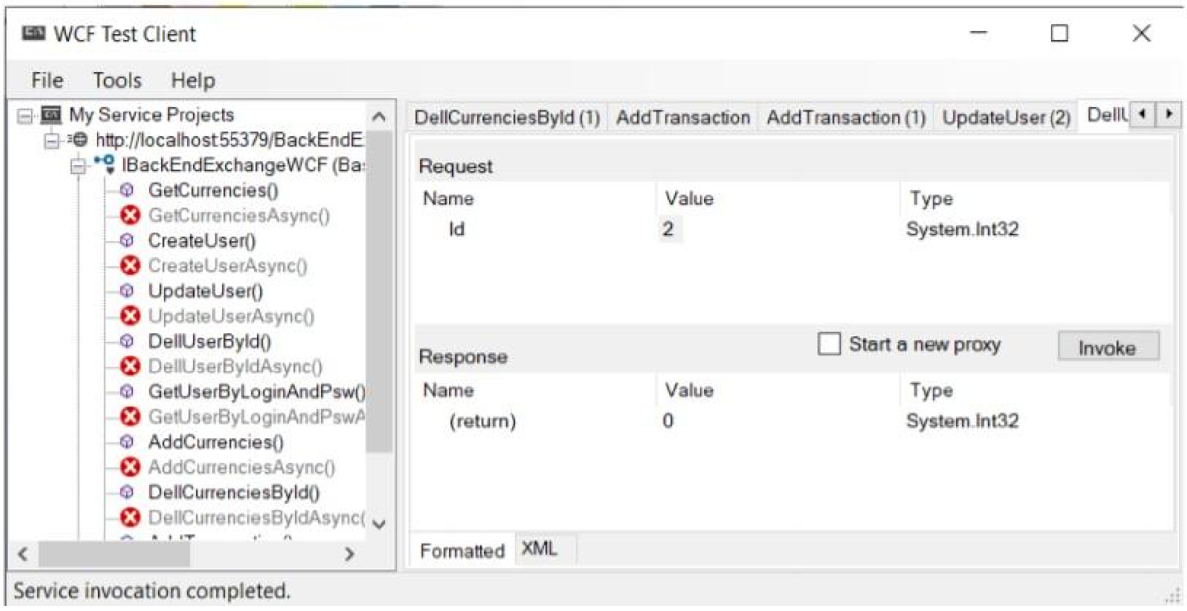


```sql
/****** Script for SelectTopNRows command from SS
SELECT TOP (1000) [Id]
      ,[Name]
      ,[Login]
      ,[Psw]
      ,[AmountPLN]
  FROM [BackEndExchangeWCFNew].[dbo].[Users]
```

	Id	Name	Login	Psw	AmountPLN
1	1	t	t	t	0
2	2	SomeChangedName	test	123456	5555

Pic. 14 The Users table

Testing the DellUserById method



Request

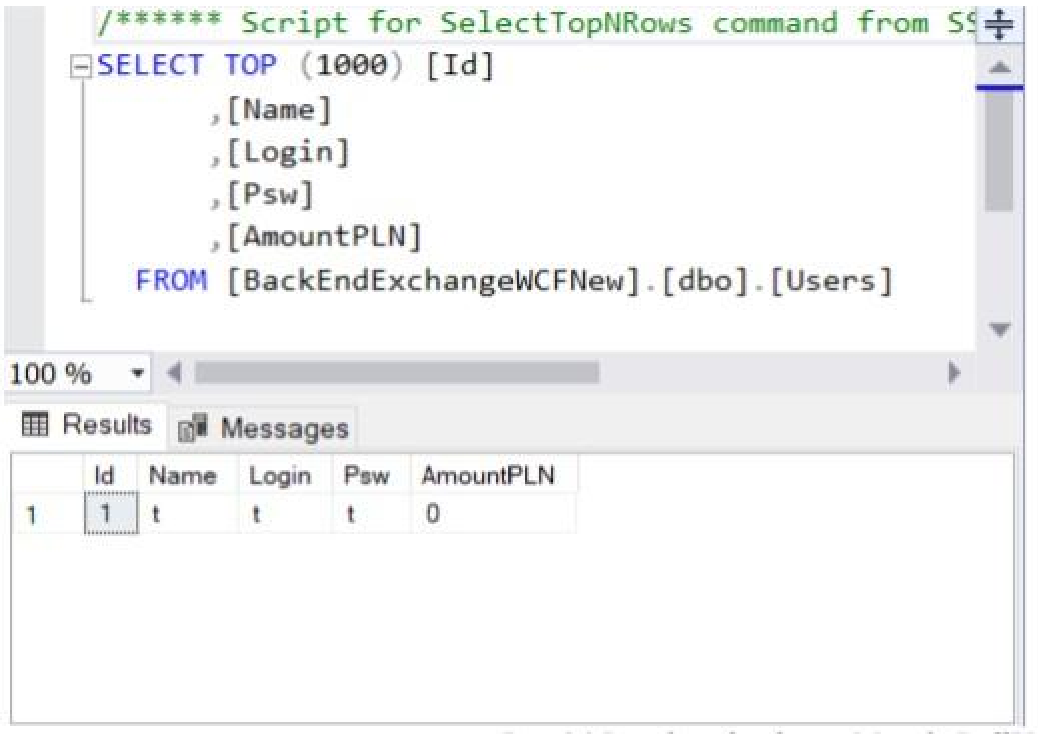
Name	Value	Type
Id	2	System.Int32

Response

Name	Value	Type
(return)	0	System.Int32

Service invocation completed.

Pic. 15 DellUserById method



```
/****** Script for SelectTopNRows command from SS
SELECT TOP (1000) [Id]
      ,[Name]
      ,[Login]
      ,[Psw]
      ,[AmountPLN]
  FROM [BackEndExchangeWCFNew].[dbo].[Users]
```

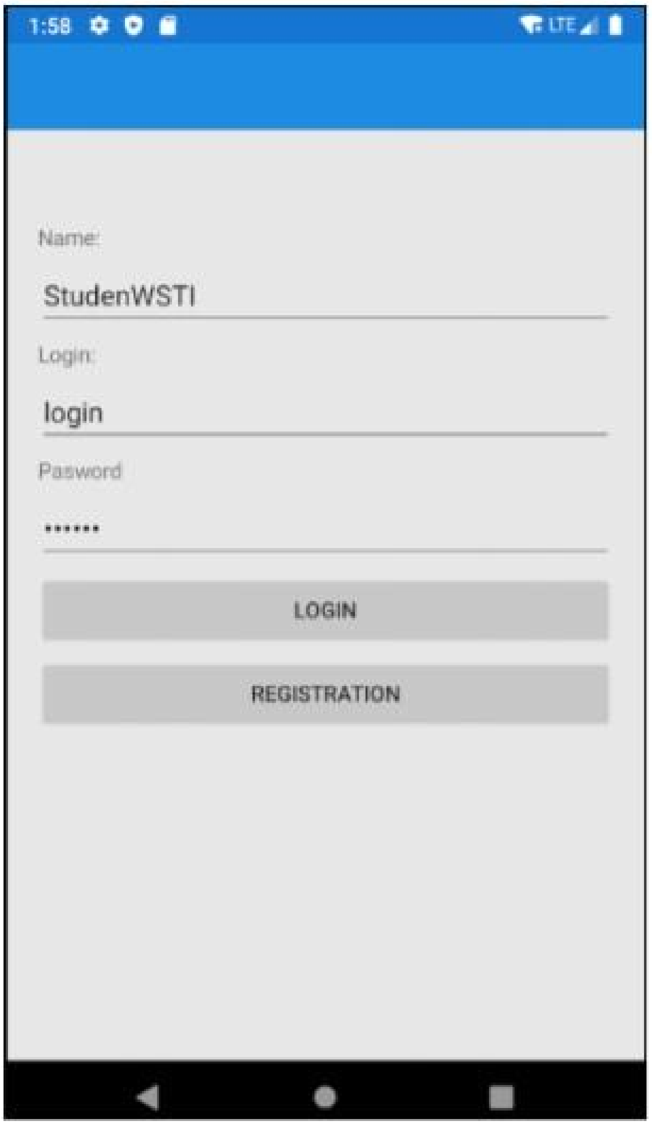
100 %

Results Messages

	Id	Name	Login	Psw	AmountPLN
1	1	t	t	t	0

Pic.16 The result of the DellUserById method

Testing and presentation of the mobile application


Name:

StudenWSTI

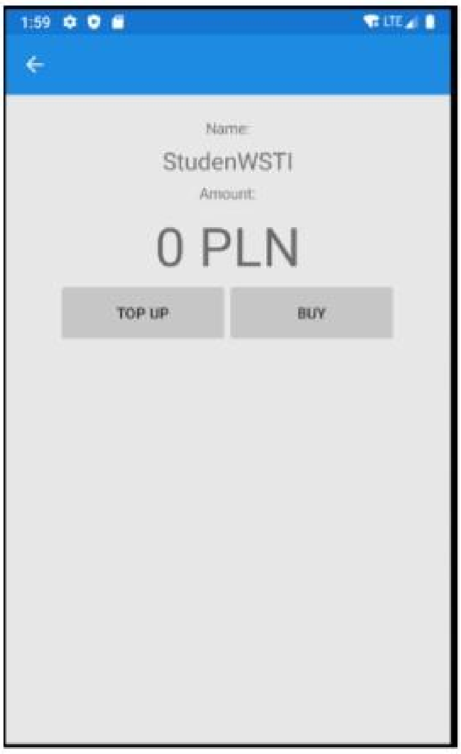
Login:

login

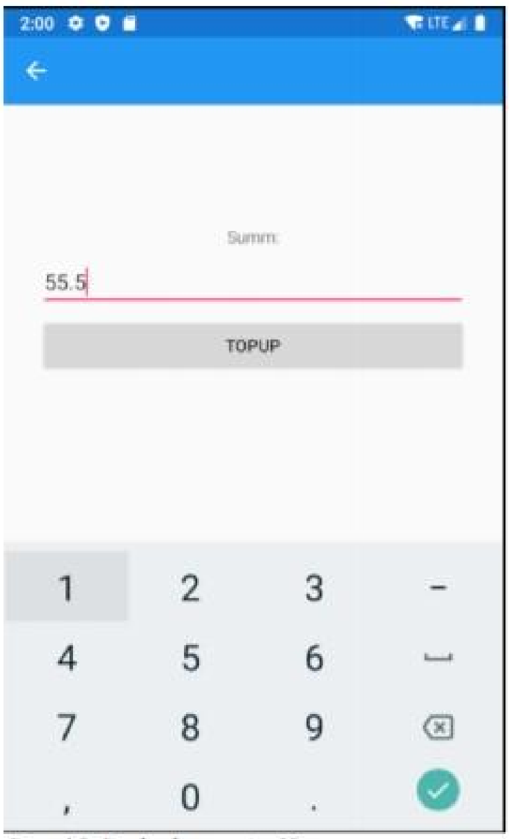
Pasword

••••••

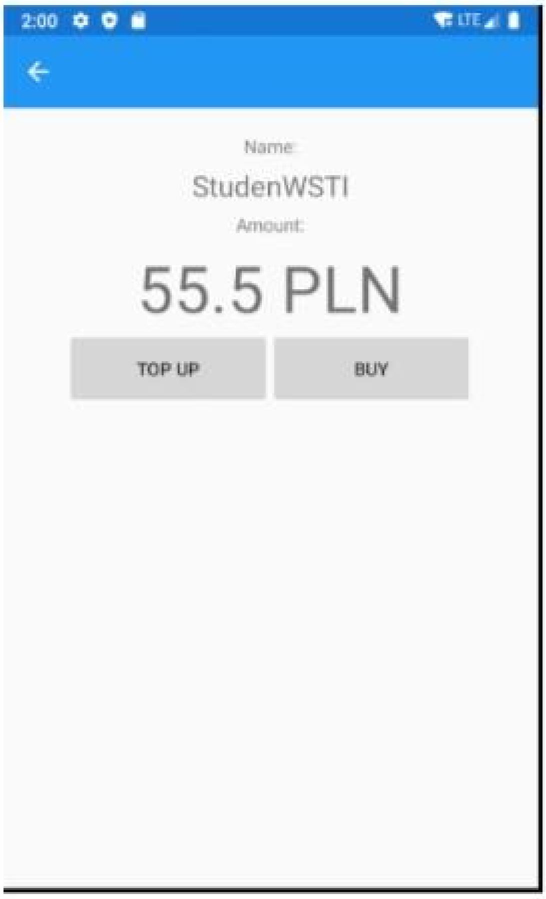
Pic. 17 Authorization window



Pic.18 User's account



Pic.19 Account top-up


2:00

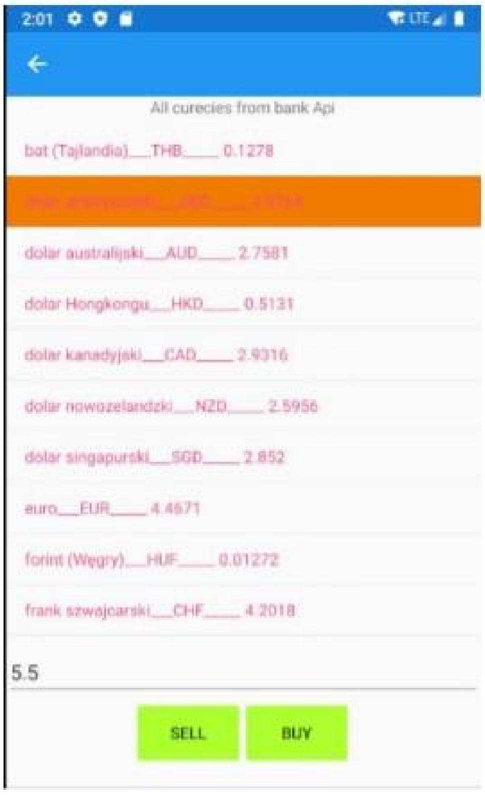
Name:

StudenWSTI

Amount:

55.5 PLN

Pic. 20 Account Top-up result


2:01

All curecies from bank Api

bat (Tajlandia)__THB____ 0.1278

[illegible]

dolar australijski__AUD____ 2.7581

dolar Hongkongu__HKD____ 0.5131

dolar kanadyjski__CAD____ 2.9316

dolar nowozelandzki__NZD____ 2.5956

dolar singapurski__SGD____ 2.852

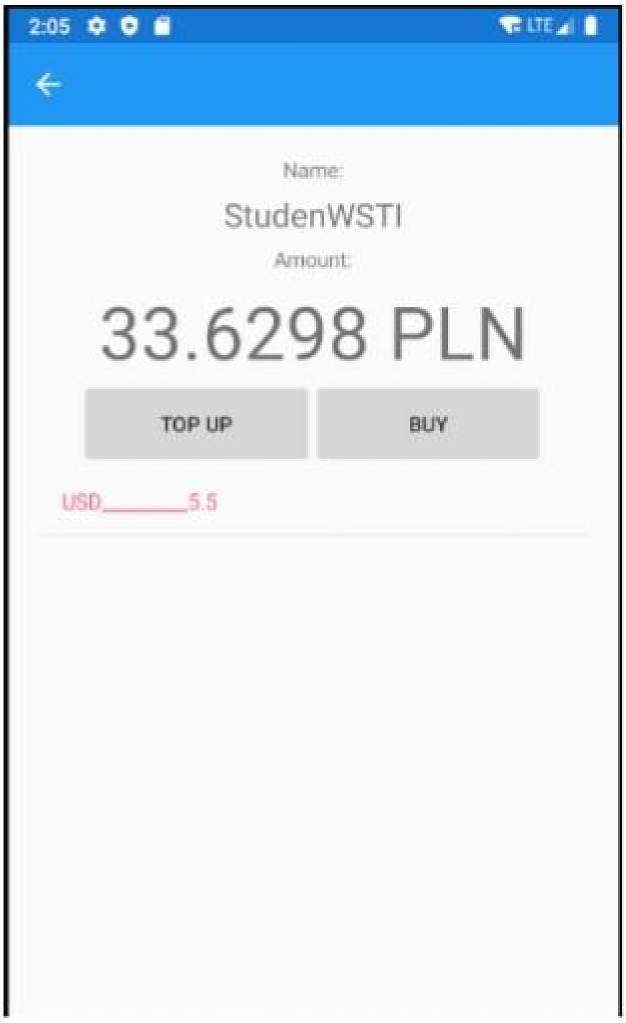
euro__EUR____ 4.4671

forint (Węgry)__HUF____ 0.01272

frank szwajcarski__CHF____ 4.2018

5.5

Pic. 21 List of available currencies. 1st process of currency purchase



Pic.22 Result of currency purchase

Conclusions

During the project development, all requirements were implemented and the results tested. I learned to create service-oriented systems in order to use functionalities in different systems. For example: websites and various mobile operating systems.