**МИНОБРНАУКИ РОССИИ  
 Федеральное государственное бюджетное   
образовательное учреждение высшего образования  
 «Ухтинский государственный технический университет»   
(УГТУ)**

Кафедра вычислительной техники, информационных систем и технологий

**РАСЧЁТНО-ГРАФИЧЕСКАЯ РАБОТА №1**Дисциплина: «Архитектура информационных систем»  
  
Шифр 191407 Группа ИCТ-2-19 Курс 2   
  
Морданов Егор Владимирович

Проверил:  
доцент кафедры ВТИСиТ Н. И. Григорьев

Ухта

2021

# Задание

Разработать систему банка пользователей криптовалют стиле: MVC (Model View Controller). Построить диаграмму классов бизнес-логики.

# ОПИСАНИЕ

Криптовалюты являются довольно популярными в наше время, многие пользователи используют её даже как способы оплаты услуг в интернете. Криптовалюты являются достаточно безопасными для оплаты поэтому их и используют для покупок. Поэтому было бы неплохо иметь приложение, в котором можно было бы свободно передавать/пополнять/открывать счета в разной криптовалюте, это все уже создано, но не было сделано в одной удобной оболочке.

# ДИАГРАММА КЛАССОВ

Диаграмма классов — структурная диаграмма языка моделирования UML, демонстрирующая общую структуру иерархии классов системы, их коопераций, атрибутов (полей), методов, интерфейсов и взаимосвязей между ними.

Построим модель классов стадии проектирования (см. Рисунок 1).

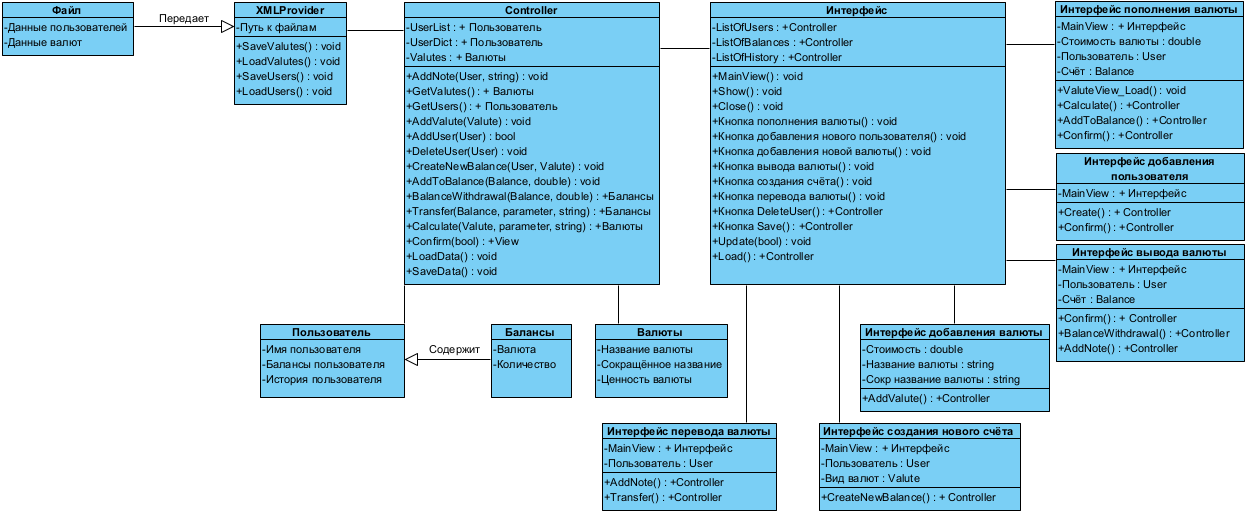


Рисунок 1 - Программная модель

# ДЕМОНСТРАЦИЯ РАБОТЫ ПРИЛОЖЕНИЯ

Главная форма приложения, на которой расположены кнопки для перевода валюты пользователю, создание счёта, пополнения валюты, вывода со счёта. А также меню действия, в котором есть такие функции как добавить пользователя, удалить пользователя, сохранить, добавить валюту. Главная форма включает в себя так же 3 списка которые хранят в себе пользователей, их счета и истории (см. Рисунок 3).

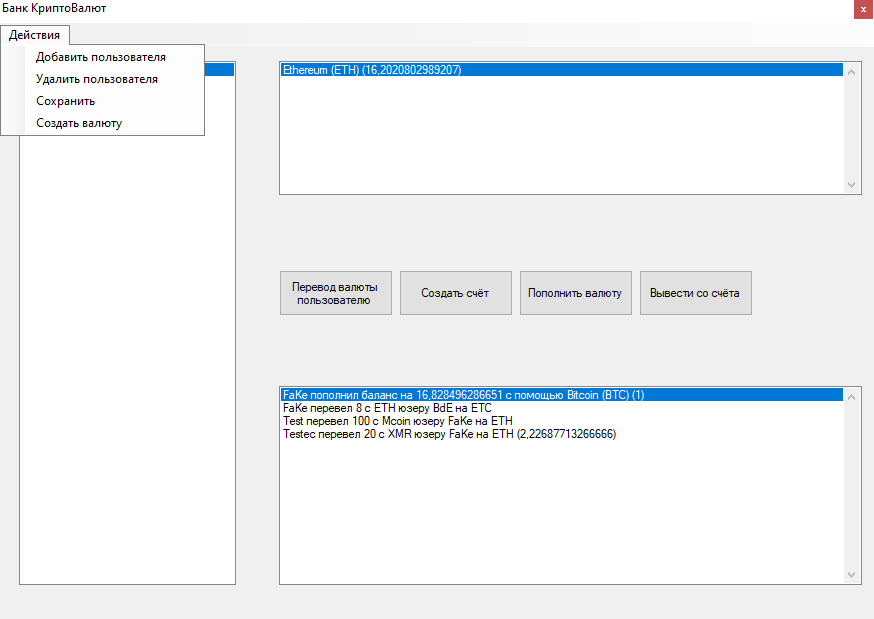


Рисунок 3 - Основное окно

Для добавления пользователя, нужно нажать соответствующую кнопку в меню «Добавить пользователя» и сразу откроется форма добавления пользователя (см. Рисунок 4).

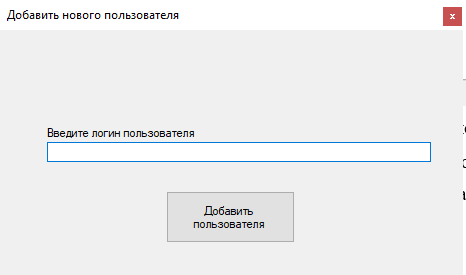


Рисунок 4 – Форма добавления пользователя

Добавим пользователя deleted чтобы он отобразился в списке наших пользователей (см. Рисунок 5).

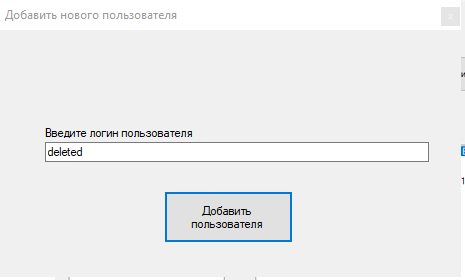


Рисунок 5 – Добавлением пользователя deleted

При нажатии добавить пользователя, выскочит всплывающее окно с подтверждением операции, данное окно будет открываться практически во всех операциях, которые мы проделываем (см. Рисунок 6)

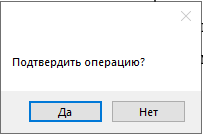


Рисунок 6 – Окно подтверждения действия

Нажав на кнопку «Нет», программа просто вернет вас в предыдущее окно с добавлением пользователя, нажав на кнопку «Да» программа исполнит операцию, таким образом мы добавили пользователя deleted (см. Рисунок 7)

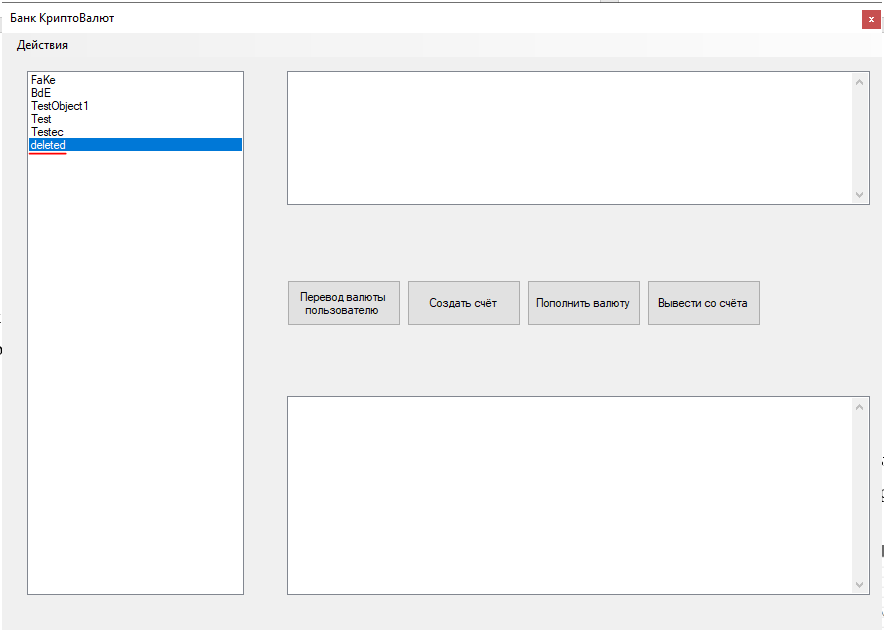


Рисунок 7 – Окно пользователей

Создадим новую валюту благодаря меню «Действия» и выбрав пункт «Создать валюту» нажав на этот пункт откроется меню добавления валюты (Рисунок 8).

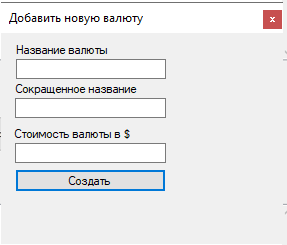


Рисунок 8 - Добавление новой валюты

Назовём новую валюту, дадим её название UseCoin, сокращённое название UCoin и стоимость в 100$ (см. Рисунок 9).

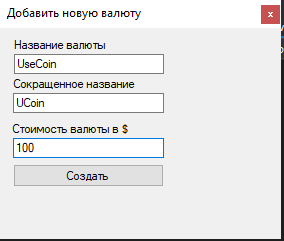


Рисунок 9 – Добавление новой валюты UseCoint

После подтверждения операции на создание новой валюты мы можем теперь открыть новый счёт для нашего нового пользователя, для этого выделить deleted пользователя и нажать на кнопку «Создать счёт» (Рисунок 10, Рисунок 11).

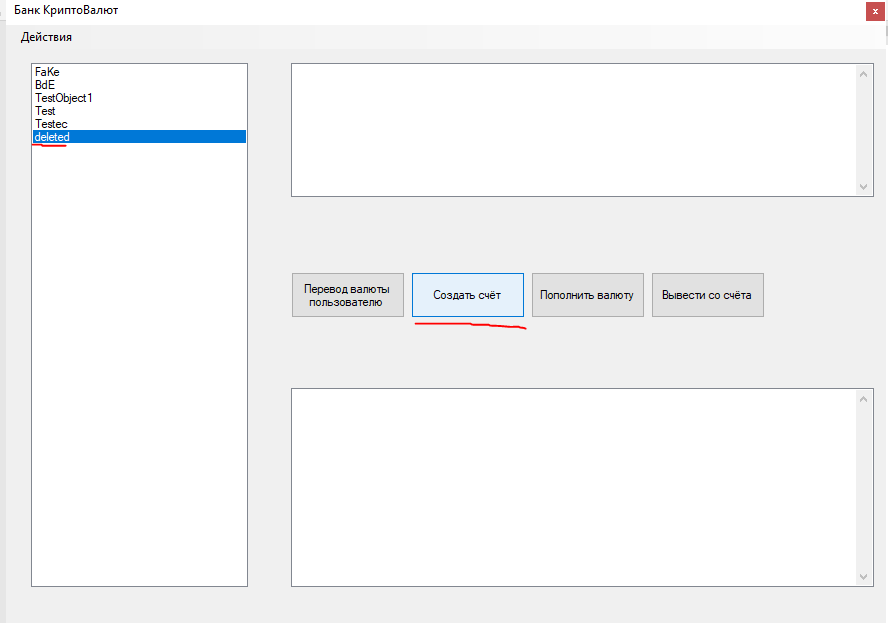


Рисунок 10 – Создание счёта для пользователя

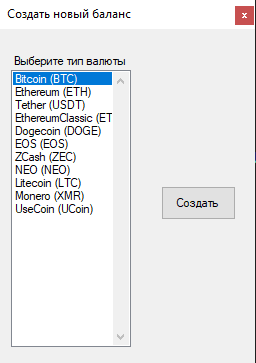


Рисунок 11 – Меню для создания нового счёта пользователю

Создав новый счёт в UseCoin пользователю deleted, его счёт появляется в основной форме, после чего можно выбрать этот баланс и нажать на кнопку «Пополнить валюту» (см. Рисунок 12)

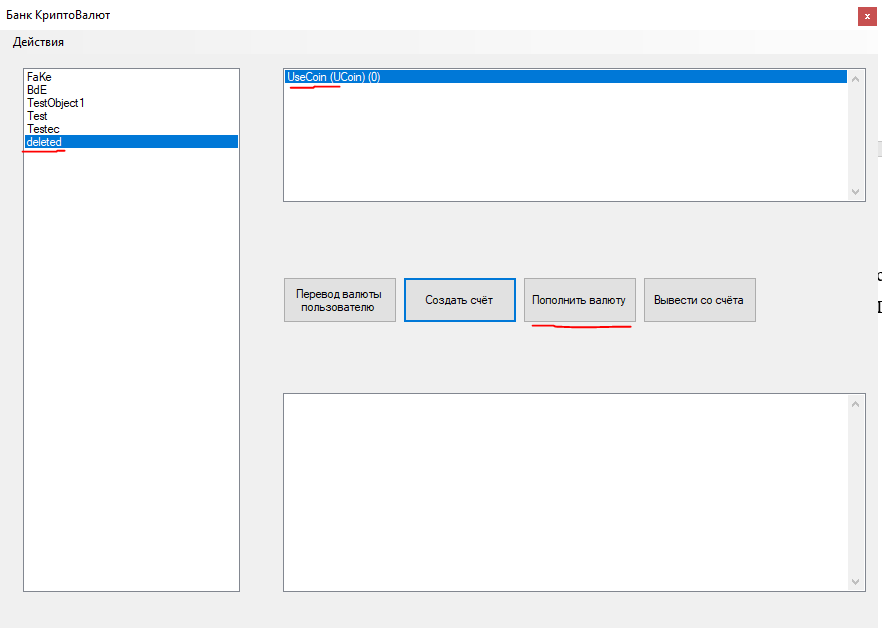


Рисунок 12 – Основное меню после добавление нового счёта пользователю

Нажав на кнопку «Пополнить валюту» откроется окно пополнения валюты (см. Рисунок 13)

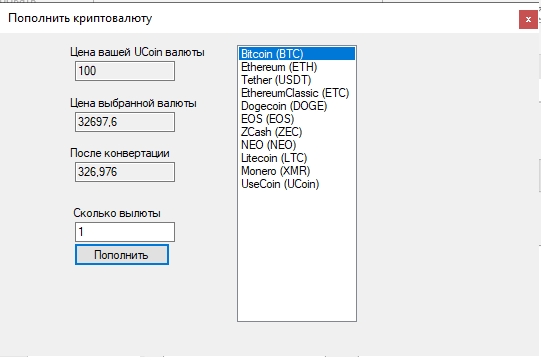


Рисунок 13 – Меню пополнения валюты

Валюту можно выбрать любую, которая присутствует в банке валют, она автоматически сделает конвертацию и запишет это в историю. Пополним наш счёт с помощью одного биткойна. (см. Рисунок 14).

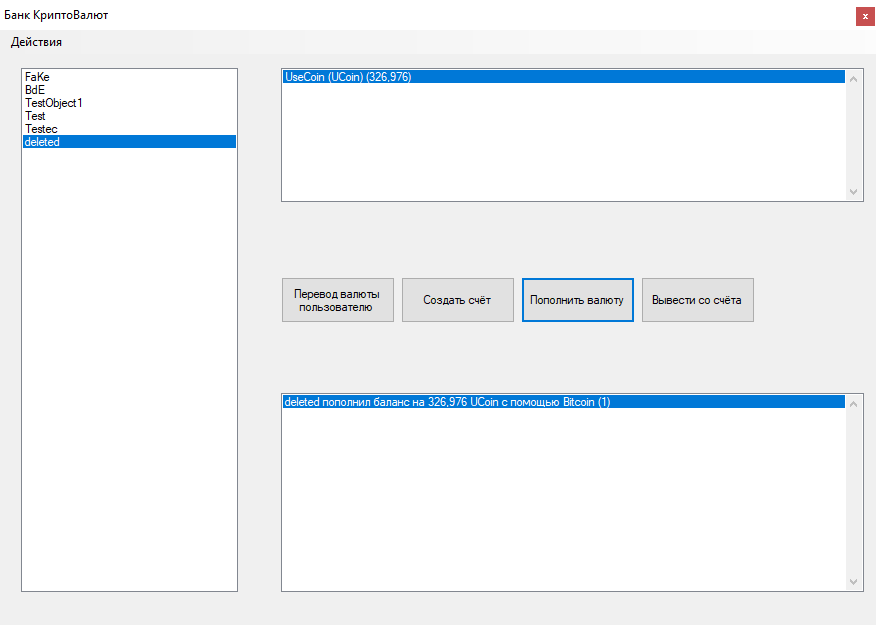


Рисунок 14 – Главная форма после пополнения баланса

Пополнив баланс, мы сразу же записали это в историю операций, что и отображается на главной форме. Переведём часть валюты другому пользователю для этого надо нажать на кнопку «Перевод валюты пользователю» и откроется данное меню (см. Рисунок 15).

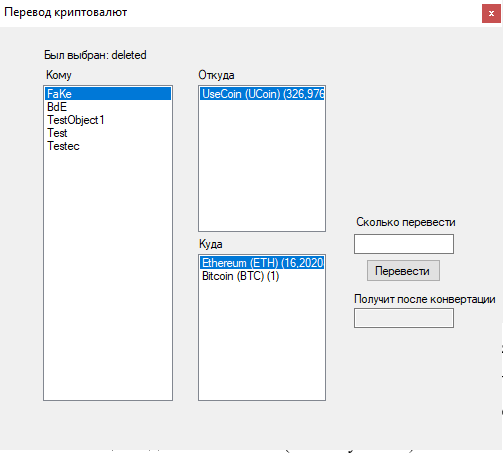


Рисунок 15 – Перевод криптовалют другим пользователям

В форме отображается кого выбрали в качестве объекта для перевода, и мы можем выбрать кому мы можем отправить валюту, надо чтобы у пользователя был обязательно открыт счёт в любой из доступных валют, банк выполнит конвертацию при переводе на другие виды валют, а также отобразит сколько получит пользователь до того, как сама операция еще будет выполнена. (см. Рисунок 16)

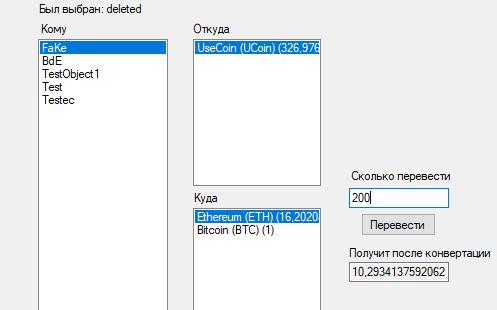


Рисунок 16 – Отображение перевода другому пользователю

Нажав кнопку перевести у нас, спишется со счёта наша валюту и отправится другом пользователю, которого мы выбрали, это также отобразится в истории пользователя (см. Рисунок 17).

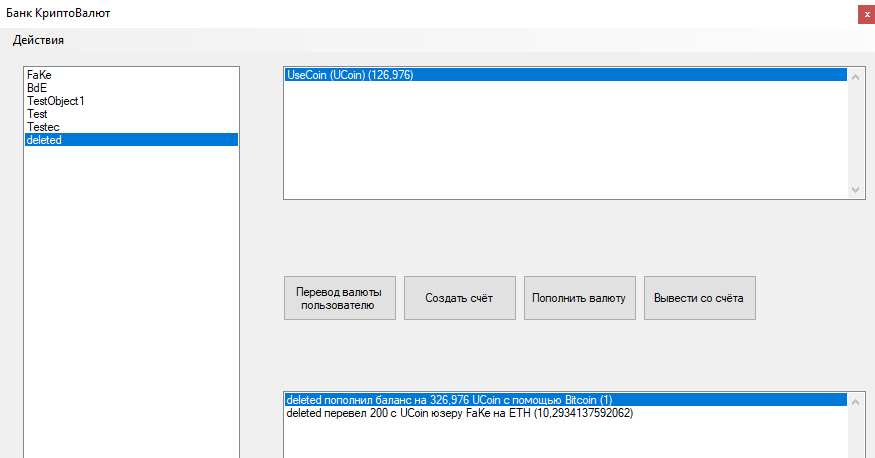


Рисунок 17 – Отображение перевода в истории

Теперь можно вывести валюту со счёта, для этого надо нажать на кнопку «Вывести со счёта» и откроется меню вывода валюты со счёта (см. Рисунок 18)

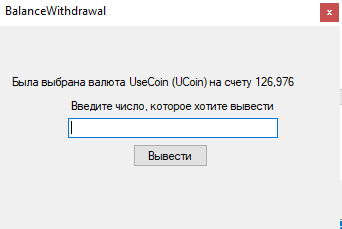


Рисунок 18 – Меню вывода валюты со счёта

Выведем со счёта 100 UCoin`s, для этого надо ввести в строке число 100 и нажать кнопку «Вывести», это также отобразится в истории операций пользователя (см. Рисунок 19)

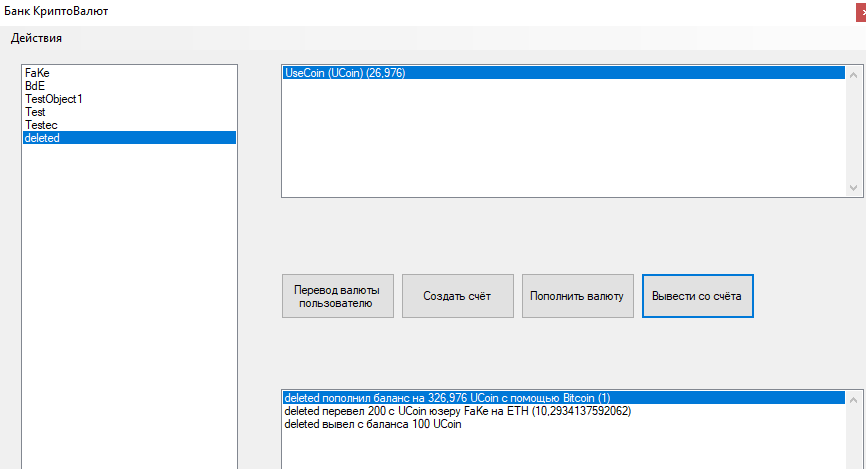


Рисунок 19 – Отображение операции вывода в истории операций

Теперь мы можем удалить данного пользователя, вместе с этим удалятся и все его счета, но останется история операций у тех пользователей, с кем он выполнял транзакции. Для удаления пользователя нужно нажать на меню «Действия» и нажать «Удалить пользователя» (см. Рисунок 20)

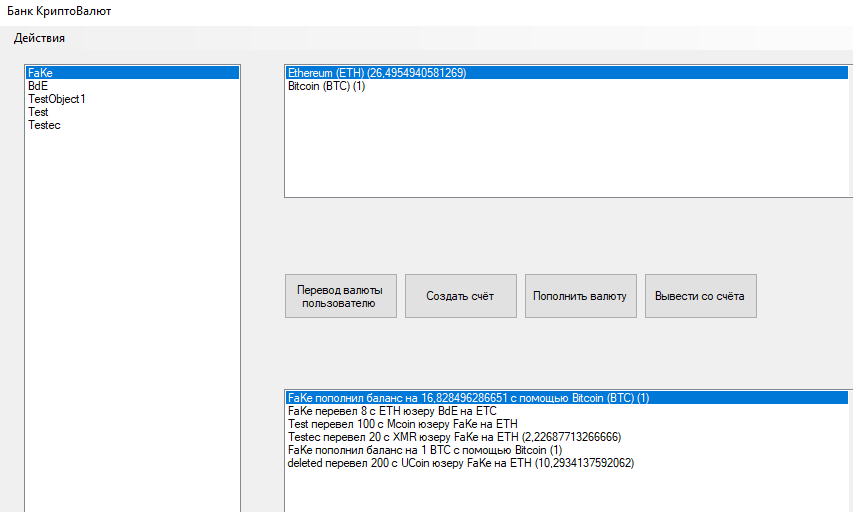


Рисунок 20 – Результат удаления пользователя

Как видно пользователь был удалён, но он все еще остался в истории операций у другого пользователя.

# Приложение А

Листинг 1 - Класс Balance

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace valutebank.DOMAIN  {  public class Balance  {  public Valute valute { get; set; }  public double amount { get; set; }  public Balance(double n, Valute u)  {  amount = n;  valute = u;  }  public Balance() {    }  public override string ToString()  {  return string.Format("{0} ({1})", valute.ToString(), amount);  }  }  } |

Листинг - класс User

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace valutebank.DOMAIN  {  public class User  {  public string Name { get; set; }    public List<Balance> BalanceList;  public List<String> History;  private User()  {  }  public User(string n) {  Name = n;  BalanceList = new List<Balance>();  History = new List<String>();  }  public override string ToString()  {  return string.Format("{0}", Name);  }  }  } |

Листинг - Класс Valute

|  |
| --- |
| using System.Xml.Serialization;  namespace valutebank.DOMAIN  {  [XmlRoot("ValCurs")]  public class Valute  {  public string CharCode { get; set; }  public string Name { get; set; }  public double Rate { get; set; }  private Valute()  {  }  public Valute(string name, string ccode, string rate)  {  CharCode = ccode;  Name = name;  Rate = double.Parse(rate);  }  public override string ToString()  {  return string.Format("{0} ({1})", Name, CharCode);  }  }  } |

Листинг 4 - класс XMLProvider для работы с файлами

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Xml.Serialization;  using System.Xml.Linq;  using System.IO;  using valutebank.DOMAIN;  namespace valutebank.DAL  {  public static class XMLProvider  {  const string ValutePath = "C:\\Users\\FaKe\\source\\repos\\valutebank\\valutebank\\Content\\valutes.xml";  const string UsersPath = "C:\\Users\\FaKe\\source\\repos\\valutebank\\valutebank\\Content\\users.xml";  public static List<Valute> LoadValutes()  {  if (!File.Exists(ValutePath)) return null;  using (var aFile = new FileStream(ValutePath, FileMode.Open))  {  XmlSerializer formatter = new XmlSerializer(typeof(List<Valute>));  byte[] buffer = new byte[aFile.Length];  aFile.Read(buffer, 0, (int)aFile.Length);  using (var stream = new MemoryStream(buffer))  {  List<Valute> data;  try  {  data = (List<Valute>)formatter.Deserialize(stream);  }  catch  {  data = null;  }  return data;  }  }  }  public static void SaveValutes(List<Valute> valutes)  {  using (var outFile = File.Create(ValutePath))  {  XmlSerializer formatter = new XmlSerializer(typeof(List<Valute>));  formatter.Serialize(outFile, valutes);  }  }  public static List<User> LoadUsers()  {  if (!File.Exists(UsersPath)) return null;  using (var aFile = new FileStream(UsersPath, FileMode.Open))  {  XmlSerializer formatter = new XmlSerializer(typeof(List<User>));  byte[] buffer = new byte[aFile.Length];  aFile.Read(buffer, 0, (int)aFile.Length);  using (var stream = new MemoryStream(buffer))  {  List<User> data;  try  {  data = (List<User>)formatter.Deserialize(stream);  }  catch  {  data = null;  }  return data;  }  }  }  public static void SaveUsers(List<User> users)  {  using (var outFile = File.Create(UsersPath))  {  XmlSerializer formatter = new XmlSerializer(typeof(List<User>));  formatter.Serialize(outFile, users);  }  }  }  } |

Листинг 5 – класс Controller

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Globalization;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Windows.Forms;  using valutebank.DAL;  using valutebank.DOMAIN;  using valutebank.View;  public static class MainController  {  private static List<User> UserList = new List<User>();  private static List<Valute> Valutes = new List<Valute>();  private static Dictionary<String, User> UserDict = new Dictionary<String, User>();  public static void LoadData()  {  var dataValutes = XMLProvider.LoadValutes();  var dataUsers = XMLProvider.LoadUsers();  if (dataValutes != null)  Valutes = dataValutes;  if (dataUsers != null) {  UserList = dataUsers;  foreach (User u in UserList)  UserDict.Add(u.Name, u);  }  }  public static void SaveData()  {  XMLProvider.SaveValutes(Valutes);  XMLProvider.SaveUsers(UserList);  }  public static void AddNote(User user, string data)  {  user.History.Add(data);  }  public static List<Valute> GetValutes()  {  return Valutes == null ? null : Valutes;  }  public static List<User> GetUsers()  {  return UserList == null ? null : UserList;  }  public static void AddValute(Valute val)  {  Valutes.Add(val);  }  public static bool AddUser(User user)  {  if (UserDict.ContainsKey(user.Name))  return false;  UserDict.Add(user.Name, user);  UserList.Add(user);  return true;  }  public static void DeleteUser(User user)  {  UserDict.Remove(user.Name);  UserList.Remove(user);  }  public static void CreateNewBalance(User user, Valute k)  {  user.BalanceList.Add(new Balance(0, k));  }  public static void AddToBalance(Balance balance, double amount)  {  balance.amount += amount;  }  public static bool BalanceWithdrawal(Balance balance, double amount)  {  try  {  if (balance.amount < amount)  {  MessageBox.Show("Недостаточно средств");  return false;  }  balance.amount -= amount;  return true;  }  catch  {  MessageBox.Show("Введено некорректное значение");  return false;  }  }  public static double Transfer(Balance from, Balance to, String textAmount)  {  try  {  double amount = double.Parse(textAmount);  if (from.amount < amount)  {  MessageBox.Show("Недостаточно средств");  return -1;  }  double received = Calculate(from.valute, to.valute, textAmount);  to.amount += received;  from.amount -= amount;  return amount;  }  catch  {  MessageBox.Show("Введено некорректное значение");  return -1;  }  }  public static double Calculate(Valute from, Valute to, string amountStr, TextBox tbox = null)  {  if (tbox != null)  tbox.Text = to.Rate.ToString();  double amount = double.Parse(amountStr);  return (from.Rate \* amount) / to.Rate;  }  public static bool Confirm()  {  DialogResult result = MessageBox.Show("Подтвердить операцию?","", MessageBoxButtons.YesNo);  if (result == DialogResult.Yes)  {  return true;  }  else  {  return false;  }  }  } |

Листинг - интерфейс MainView

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.ComponentModel;  using System.Data;  using System.Drawing;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Windows.Forms;  using valutebank.DOMAIN;  namespace valutebank.View  {  public partial class MainView : Form  {  public MainView()  {  InitializeComponent();  MainController.SaveData();  Update(false);  }  public void Update(bool ignore = true)  {  if (!ignore) {  listBoxOfUsers.DataSource = null;  listBoxOfUsers.DataSource = MainController.GetUsers();  }  listBoxAccount.DataSource = (listBoxOfUsers.SelectedItem as User)?.BalanceList?.ToList();  listBoxHistory.DataSource = (listBoxOfUsers.SelectedItem as User)?.History?.ToList();    }  private void button4\_Click(object sender, EventArgs e)  {  var cur = listBoxAccount.SelectedItem as Balance;  if (cur == null)  {  MessageBox.Show("Валюта не выбрана");  return;  }  var user = listBoxOfUsers.SelectedItem as User;  if (user == null)  {  MessageBox.Show("Пользователь не выбран");  return;  }  var bal = listBoxAccount.SelectedItem as Balance;  var cryptoval = cur.valute.Rate;  AddCryptoValute addform = new AddCryptoValute(this, cryptoval, user, bal);  addform.Show();  }  private void button1\_Click(object sender, EventArgs e)  {  var user = listBoxOfUsers.SelectedItem as User;  if (user == null)  {  MessageBox.Show("Пользователь не выбран");  return;  }  var bal = listBoxAccount.SelectedItem as Balance;  if (bal == null)  {  MessageBox.Show("Валюта не выбрана");  return;  }  BalanceWithdrawal addform = new BalanceWithdrawal(this, user, bal);  addform.Show();  }  public void button6\_Click(object sender, EventArgs e)  {  var user = listBoxOfUsers.SelectedItem as User;  if (user == null)  {  MessageBox.Show("Пользователь не выбран");  return;  }  CreateBalance addform = new CreateBalance(this, user);  addform.Show();  }  private void button2\_Click(object sender, EventArgs e)  {  var user = listBoxOfUsers.SelectedItem as User;  if (user == null)  {  MessageBox.Show("Пользователь не выбран");  return;  }  CryptoCurrencyTransfer addform = new CryptoCurrencyTransfer(this, user);  addform.Show();  }  private void добавитьПользователяToolStripMenuItem\_Click(object sender, EventArgs e)  {  AddNewUser addform = new AddNewUser(this);  addform.Show();  }  private void создатьВалютуToolStripMenuItem\_Click(object sender, EventArgs e)  {  AddNewValute addform = new AddNewValute();  addform.Show();  }  private void сохранитьToolStripMenuItem\_Click(object sender, EventArgs e)  {  MainController.SaveData();  }  private void удалитьПользователяToolStripMenuItem\_Click(object sender, EventArgs e)  {  var user = listBoxOfUsers.SelectedItem as User;  if (user == null)  {  MessageBox.Show("Пользователь не выбран");  return;  }  if (MainController.Confirm() == true)  {  MainController.DeleteUser(user);  }  Update(false);  }  private void listBoxOfUsers\_SelectedIndexChanged(object sender, EventArgs e)  {  Update();  }  }  } |

Листинг - интерфейс CryptoCurrencyTransfer

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using valutebank.DOMAIN;

using valutebank.View;

namespace valutebank.View

{

public partial class CryptoCurrencyTransfer : Form

{

private MainView mV = null;

private User user;

public CryptoCurrencyTransfer(Form callingForm, User u)

{

mV = callingForm as MainView;

user = u;

InitializeComponent();

LabelSelected.Text = "Был выбран: " + user.Name;

}

private void CryptoCurrencyTransfer\_Load(object sender, EventArgs e)

{

ListFrom.DataSource = user?.BalanceList?.ToList();

var userList = MainController.GetUsers().Where(x => x != user).ToList();

ListUsers.DataSource = userList;

}

private void listBoxListToTrans\_SelectedIndexChanged(object sender, EventArgs e)

{

var user = ListUsers.SelectedItem as User;

ListTo.DataSource = user?.BalanceList?.ToList();

}

private void button1\_Click(object sender, EventArgs e)

{

if (MainController.Confirm() == true)

{

Balance from = ListFrom.SelectedItem as Balance, to = ListTo.SelectedItem as Balance;

double amount = MainController.Transfer(from, to, TextAmount.Text);

if (amount != -1)

{

string note = user.Name + " перевел " + amount.ToString() +

" с " + from.valute.CharCode +

" юзеру " + (ListUsers.SelectedItem as User).ToString() + " на " + to.valute.CharCode +" (" + textBox3.Text + ")";

MainController.AddNote(user, note);

MainController.AddNote(ListUsers.SelectedItem as User, note);

mV.Update();

Close();

}

}

}

private void TextAmount\_TextChanged(object sender, EventArgs e)

{

Balance from = ListFrom.SelectedItem as Balance, to = ListTo.SelectedItem as Balance;

if (TextAmount.Text != null && TextAmount.Text != "")

{

double transaction = MainController.Calculate(from.valute, to.valute, TextAmount.Text);

textBox3.Text = transaction.ToString();

}

}

}

}

Листинг - интерфейс BalanceWithdrawal

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.ComponentModel;  using System.Data;  using System.Drawing;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Windows.Forms;  using valutebank.DOMAIN;  using valutebank.View;  namespace valutebank.View  {  public partial class BalanceWithdrawal : Form  {  private MainView mV = null;  private User user;  private Balance balance;  public BalanceWithdrawal(Form callingForm, User u ,Balance b)  {  mV = callingForm as MainView;  user = u;  balance = b;  InitializeComponent();  }  private void BalanceWithdrawal\_Load(object sender, EventArgs e)  {  label2.Text = "Была выбрана валюта "+ balance.valute.ToString() + " на счету " + balance.amount.ToString();  }  private void button1\_Click(object sender, EventArgs e)  {  if (MainController.Confirm() == true)  {  double amount = double.Parse(textBox1.Text);  if ((MainController.BalanceWithdrawal(balance, amount)) == true)  {  string note = user.Name + " вывел с баланса " + amount.ToString() + " " + balance.valute.CharCode.ToString();  MainController.AddNote(user, note);  mV.Update();  Close();  }  else  return;  }  }  }  } |

Листинг - интерфейс CreateBalance

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.ComponentModel;  using System.Data;  using System.Drawing;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Windows.Forms;  using valutebank.DOMAIN;  namespace valutebank.View  {  public partial class CreateBalance : Form  {  private MainView mV = null;  private User user;  public CreateBalance(Form callingForm, User u)  {  mV = callingForm as MainView;  user = u;  InitializeComponent();  }  private void ValuteView\_Load(object sender, EventArgs e)  {  listBoxOfCryptoValutes.DataSource = MainController.GetValutes();  }  private void button1\_Click(object sender, EventArgs e)  {  if (MainController.Confirm() == true)  {  MainController.CreateNewBalance(user, listBoxOfCryptoValutes.SelectedItem as Valute);  mV.Update();  Close();  }  }  }  } |

Листинг 10 - интерфейс AddNewValute

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.ComponentModel;  using System.Data;  using System.Drawing;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Text.RegularExpressions;  using System.Windows.Forms;  using valutebank.DOMAIN;  namespace valutebank.View  {  public partial class AddNewValute : Form  {  public AddNewValute()  {  InitializeComponent();  }  private void button1\_Click(object sender, EventArgs e)  {  if (MainController.Confirm() == true)  {  double value;  Regex regex = new Regex(@"^[a-zA-Z]\*$");  if (regex.IsMatch(textBox1.Text) == false || regex.IsMatch(textBox3.Text) == false)  {  MessageBox.Show("Название валюты может содержать только латинские буквы");  return;  }  if (textBox1.Text.Length < textBox3.Text.Length)  {  MessageBox.Show("Сокращенное название должно быть короче оригинального");  return;  }  try  {  value = double.Parse(textBox2.Text);  Valute valute = new Valute(textBox1.Text, textBox3.Text, textBox2.Text);  MainController.AddValute(valute);  Close();  }  catch  {  MessageBox.Show("Введено некорректное число");  }  }  }  }  } |

Листинг 11 - интерфейс AddNewUser

|  |
| --- |
| using System;  using System.Text.RegularExpressions;  using System.Windows.Forms;  using valutebank.DOMAIN;  namespace valutebank.View  {  public partial class AddNewUser : Form  {  private MainView mV = null;  public AddNewUser(Form callingForm)  {  mV = callingForm as MainView;  InitializeComponent();  ActiveControl = textBox1;  }  private void Create()  {  Regex regex = new Regex(@"^[a-zA-Z0-9]+([.\_]?[a-zA-Z0-9]+)\*$");  User u = new User(textBox1.Text);  if (regex.IsMatch(u.Name) == true)  {  if (MainController.AddUser(u) == true)  {  mV.Update(false);  Close();  }  else  MessageBox.Show("Пользователь с таким именем уже существует");  }  else  MessageBox.Show("Имя пользователя не удовлетворяет регулярному выражению", "Введены запрещенные символы");  }  private void button1\_Click(object sender, EventArgs e)  {  if (MainController.Confirm() == true)  {  Create();  }  }  private void textBox1\_KeyDown(object sender, KeyEventArgs e)  {  if (e.KeyCode == Keys.Enter)  if (MainController.Confirm() == true)  {  Create();  }  }  }  } |

Листинг 12 - интерфейс AddCryptoValute

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.ComponentModel;  using System.Data;  using System.Drawing;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.IO;  using System.Globalization;  using System.Windows.Forms;  using valutebank.DOMAIN;  using valutebank.View;  namespace valutebank.View  {  public partial class AddCryptoValute : Form  {  private MainView mV = null;  private double rate;  private User user;  private Balance balance;  public AddCryptoValute(Form callingForm, double cv, User u, Balance b)  {  mV = callingForm as MainView;  rate = cv;  user = u;  balance = b;  InitializeComponent();  }  private double Calculate()  {  return MainController.Calculate(ListValutes.SelectedItem as Valute, balance.valute, textBox1.Text, textBox2);  }  private void ValuteView\_Load(object sender, EventArgs e)  {  ListValutes.DataSource = MainController.GetValutes();  textBox1.Text = "1";  textBox2.Text = "1";  label1.Text = "Цена вашей " + balance.valute.CharCode + " валюты";  Calculate();  }  private void button1\_Click(object sender, EventArgs e)  {  if (MainController.Confirm() == true)  {  double amount = Calculate();  MainController.AddToBalance(balance, amount);  string note = user.Name + " пополнил баланс на " + amount.ToString()+ " " + balance.valute.CharCode + " с помощью " + (ListValutes.SelectedItem as Valute).Name.ToString() + " (" + textBox1.Text + ")";  MainController.AddNote(user, note);  mV.Update();  Close();  }  }  private void textBox1\_TextChanged(object sender, EventArgs e)  {  textBox3.Text = Convert.ToString(Calculate());  textBox4.Text = (ListValutes.SelectedItem as Valute).Rate.ToString();  }  private void listBoxOfCryptoIndValute\_SelectedIndexChanged(object sender, EventArgs e)  {  textBox3.Text = Convert.ToString(Calculate());  textBox4.Text = (ListValutes.SelectedItem as Valute).Rate.ToString();  }  }  } |