НИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ И ПРОГРАММНОЙ ИНЖЕНЕРИИ

КУРСОВОЙ ПРОЕКТ ЗАЩИЩЕН С ОЦЕНКОЙ

РУКОВОДИТЕЛЬ

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Старший преподаватель |  |  |  |  | Е.О. Шумова |  |  |
|  |  |  |  |  |  |  |  |
| должность, уч. степень, звание | подпись, дата |  | инициалы, фамилия | |  |

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОМУ ПРОЕКТУ

**РАЗРАБОТКА ПРИЛОЖЕНИЯ ДЛЯ ОРГАНИЗАЦИИ ВЗАИМОДЕЙСТВИЯ ОБЬЕКТОВ ПРИ ЗАДАННЫХ КРИТЕРИЯХ**

по дисциплине:

ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ

|  |  |  |
| --- | --- | --- |
| РАБОТУ ВЫПОЛНИЛА |  |  |
| СТУДЕНТКА ГР. 4731 | А.В. Доценко инициалы, фамилия |  |
| подпись, дата |  |



Санкт-Петербург 2019

СОДЕРЖАНИЕ

ЦЕЛИ И ЗАДАЧИ КУРСОВОГО ПРОЕКТИРОВАНИЯ 3

ЗАДАНИЕ НА КУРСОВОЕ ПРОЕКТИРОВАНИЕ Разработка класса «Банковский счёт» 4

ВВЕДЕНИЕ 5

1. Постановка задачи

1.1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 6

1.2 ФОРМУЛИРОВКА ТЕХНИЧЕСКОГО ЗАДАНИЯ 7

2. Проектирование классов

2.1 КЛАССЫ СУЩНОСТЕЙ 8

2.2 УПРАВЛЯЮЩИЕ КЛАССЫ 8

2.3 ИЕРАРХИЯ КЛАССОВ 9

**3. Разработка приложения**

3.1 РАЗРАБОТКА ИНТЕРФЕЙСА ПРИЛОЖЕНИЯ 10

3.2 РЕАЛИЗАЦИЯ КЛАССОВ 12

**4. Тестирование**

4.1 СКРИНШОТЫ ВЫПОЛНЕНИЯ ПРОГРАММЫ1 [14](file:///C:\Users\настя\Downloads\OOP_Nash_D_2.docx#_Toc532307716)

5. Заключение

5.1 ВЫВОД 19

5.2 БИБЛИОГРАФИЧЕСКИЙ СПИСОК 20

5.3 ПРИЛОЖЕНИЕ [21](file:///C:\Users\настя\Downloads\OOP_Nash_D_2.docx#_Toc532307716)

5.3.1. App.config 21

5.3.2. DBcards.cs 22

5.3.3. DBusers.cs 23

5.3.4. FormCard.cs 25

5.3.5. FormUser.cs 27

5.3.6. MainMenu.cs 29

5.3.7. Program.cs 30

5.3.8. SQLConnector.cs 31

### **ЦЕЛИ И ЗАДАЧИ КУРСОВОГО ПРОЕКТИРОВАНИЯ**

*Целями курсового проекта* являются:

* Закрепление знаний, полученных в ходе изучения дисциплины «Объектно-ориентированное программирование»
* Приобретение навыков практического программирования с использованием объектно-ориентированной парадигмы
* Подготовка к выполнению выпускной квалификационной работы

*Задачей курсового проекта* является разработка иерархии типов в заданной предметной области, включающей в себя:

* Разработку иерархии классов
* Разработку приложения
* Подготовку презентации

### **ЗАДАНИЕ НА КУРСОВОЕ ПРОЕКТИРОВАНИЕ**

В ходе курсового проекта необходимо разработать приложение, позволяющее протестировать взаимодействие объектов классов, спроектированных и реализованных для решения конкретной задачи при заданных критериях. При разработке программного обеспечения следует использовать шаблоны проектирования. Также предпочтение должно быть отдано графическому приложению.

**Тема курсового проекта:** *Разработка класса «Банковский счет»*

Исходя из задания на курсовой проект и основных правил, можно выделить ряд обязательств, необходимых для разработки класса «Банковский счёт»:

* В системе должны поддерживаться ввод/вывод информации о держателе счета
* Приход и списание средств
* Привязка/отвязка банковской карточки к/от счёту/счёта

### **ВВЕДЕНИЕ**

Тенденции развития современных информационных технологий приводят к постоянному возрастанию сложности информационных систем (ИС), создаваемых в различных областях деятельности человека. Под термином информационные системы мы будем понимать среду, составляющими элементами которой являются компьютеры, компьютерные сети, программные продукты, базы данных, люди, различного рода технические и программные средства связи и т.д. Основная цель информационной системы **–** организация хранения, обработки и передачи информации. ИС представляет собой человеко-компьютерную систему обработки информации.

Современный финансовый мир не может работать без банков. Банки играют важную роль в управлении финансовыми потоками частных лиц, организаций и даже государства.

Данная работа посвящена разработке программного обеспечения (ПО) информационной системы банка, позволяющего освободить сотрудников от рутинных работ, связанных с подготовкой карточек, счёта, многоаспектного поиска пользователя по различным признакам, учета и ведения баланса.

Предметом исследования являются процессы, происходящие в банке, связанные с привязкой карт к пользователям и работой над их счётом.

Создание информационных систем банка, обладающих интуитивно-понятным интерфейсом, реализующих максимально возможное число банковских технологических процессов и операций, которые в принципе поддаются автоматизации, является актуальной и перспективной задачей, решение которой позволит значительно повысить эффективность работы сотрудников.

1. ПОСТАНОВКА ЗАДАЧИ

### **АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ**

Рассматриваемая предметная область проекта – банк. Взаимодействующие между собой, банк и пользователь являются субъектами, составляющими основу предметной области, а карта представляет собой используемый объект. Взаимодействие банка и пользователя заключается в том, что они могут узнать данные друг о друге.

Рассматриваемые субъекты могут использовать объект «Карта» одинаково, просматривая лишь данные об этом объекте. Однако субъекты независимо друг от друга могут выполнять и следующие действия: пользователь может зарегистрироваться в банке, положить и снять деньги со счёта, а банк может привязать карту к пользователю, работать со счётом.

Рассматриваемые субъекты и объект являются основными понятиями данной предметной области.

Объект «Карта» можно охарактеризовать следующими параметрами: label\_id, id пользователя, ФИО пользователя, счёт. Данные понятия представляют собой свойства объекта. В свою очередь субъекты «Банк» и «Пользователь» могут содержать в себе такие параметры как ФИО, id. Эти понятия представляют свойства данных субъектов.

### **ФОРМУЛИРОВКА ТЕХНИЧЕСКОГО ЗАДАНИЯ**

Исходя из задания на курсовой проект и основных правил, можно выделить ряд обязательств, необходимых для разработки класса «Банковский счёт»:

* В системе должны поддерживаться ввод/вывод информации о держателе счета
* Приход и списание средств
* Привязка/отвязка банковской карточки к/от счёту/счёта

Следуя вышеперечисленным правилам, можно выделить несколько основных классов, необходимых для разработки программного обеспечения:

* **FormUser**

Класс, необходимый для работы с пользователями;

Добавление, удаление, редактирование информации о пользователе (id, фамилия, имя, отчество).

* **FormCard**

Класс, необходимый для работы с картами;

Добавление, удаление, редактирование информации о карте (id, номер, счёт).

* **DBusers**

Получение информации с SQLConnector;

Организует заполнение таблицы данными.

* **DBcards**

Получение информации с SQLConnector;

Организует заполнение таблицы данными.

* **SQLConnector**

Обеспечивает подключение к базе данных;

Получение информации из базы данных.

1. ПРОЕКТИРОВАНИЕ КЛАССОВ
   1. **КЛАССЫ СУЩНОСТЕЙ**

К классам сущностей относится класс FormUser, описывающий основные свойства пользователя, и производныq от него класс DBusers, содержащие в себе информацию и карте и счёте. Также к классам сущностей относится класс FormCard, описывающий свойства карты, и производный от него класс DBcards, описывающий связь между пользователем и картой.

* 1. **УПРАВЛЯЮЩИЕ КЛАССЫ**

В приложении также были реализованы управляющий класс:

SQLConnector, обеспечивает подключение к базе данных, получение информации из базы данных.

* 1. **ИЕРАРХИЯ КЛАССОВ**

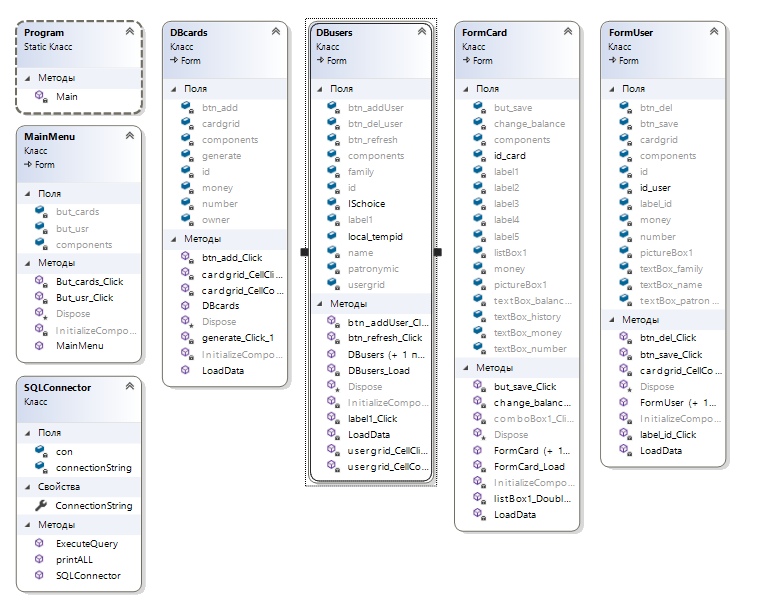
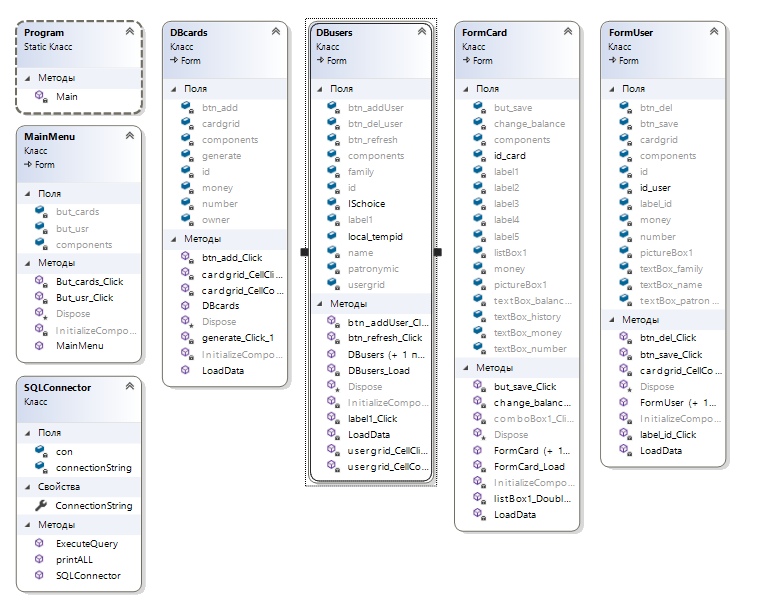
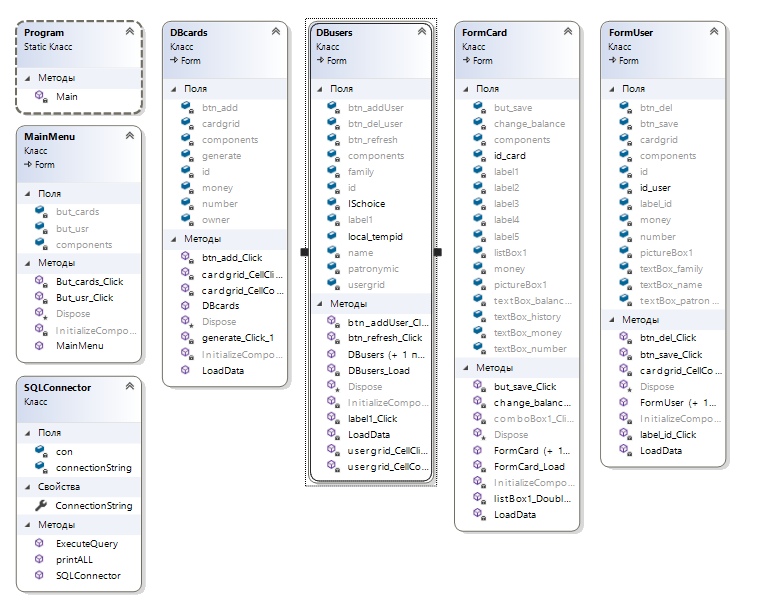
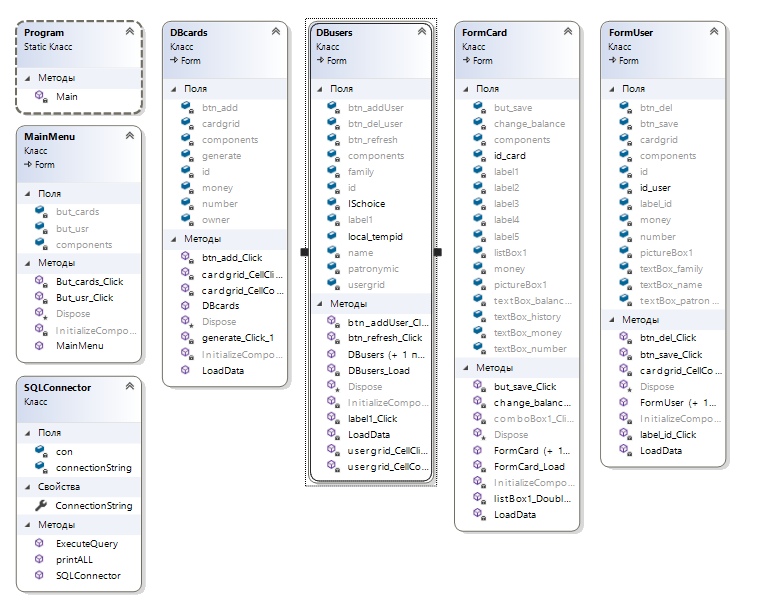
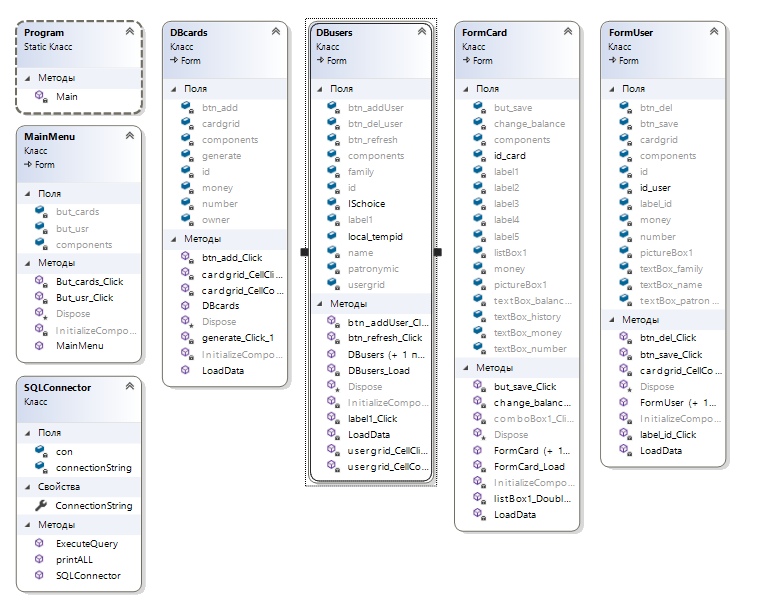
    

Рис.1. Диаграмма классов приложения.

Иерархия наследования была устроена данным образом по причине того, что на языке C# отсутствует возможность множественного наследования, однако классам проверок сущностей необходимо наследоваться от главного абстрактного класса проверки и от соответствующих им классов сущностей (чтобы сохранить свойства сущностей приватными, реализуя инкапсуляцию). Управляющие классы, используя идею полиморфизма реализуют задачи одной и той же идеи разными способами (идея поиска ключа для классов сущностей).

3.РАЗРАБОТКА ПРИЛОЖЕНИЯ

**3.1 РАЗРАБОТКА ИНТЕРФЕЙСА ПРИЛОЖЕНИЯ**

Пользовательский интерфейс приложения реализуется на языке C# в среде разработки Microsoft Visual Studio на основе сформулированного ТЗ. Используются два режима: Клиенты и Карты.

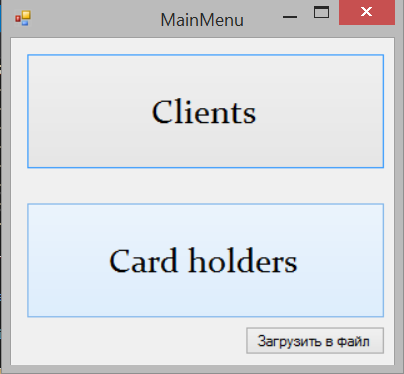


Рис.2. Меню.

Нажав на «Clients», перед нами откроется окно с id пользователя, Имя пользователя, Фамилия пользователя, Отчество пользователя и функция изменить данные.

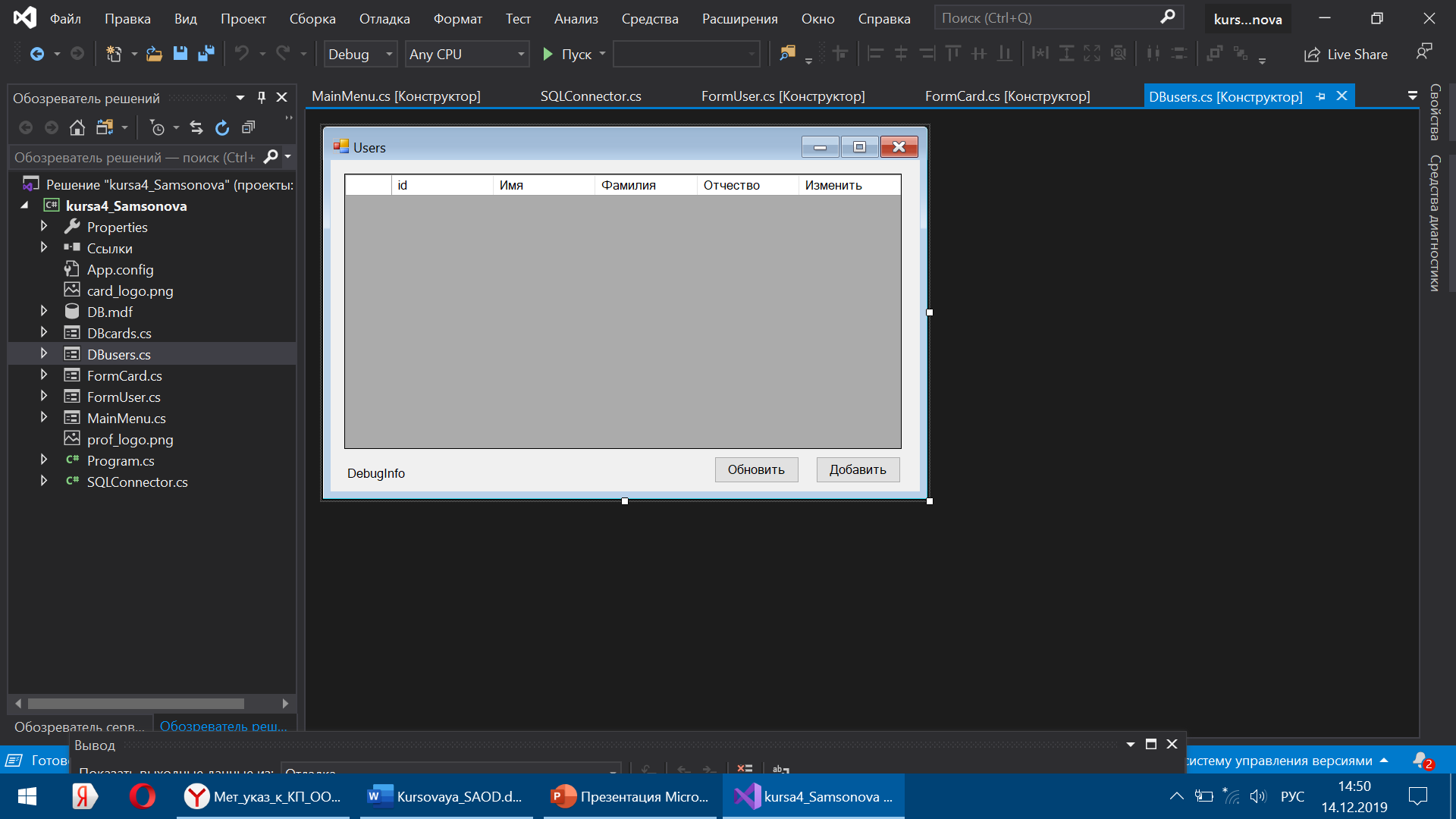


Рис.3. Пользователь.

После добавления пользователя в базу данных, будет возможность нажать на имя клиента и в отдельном окне будет отображаться профиль и его счёт.

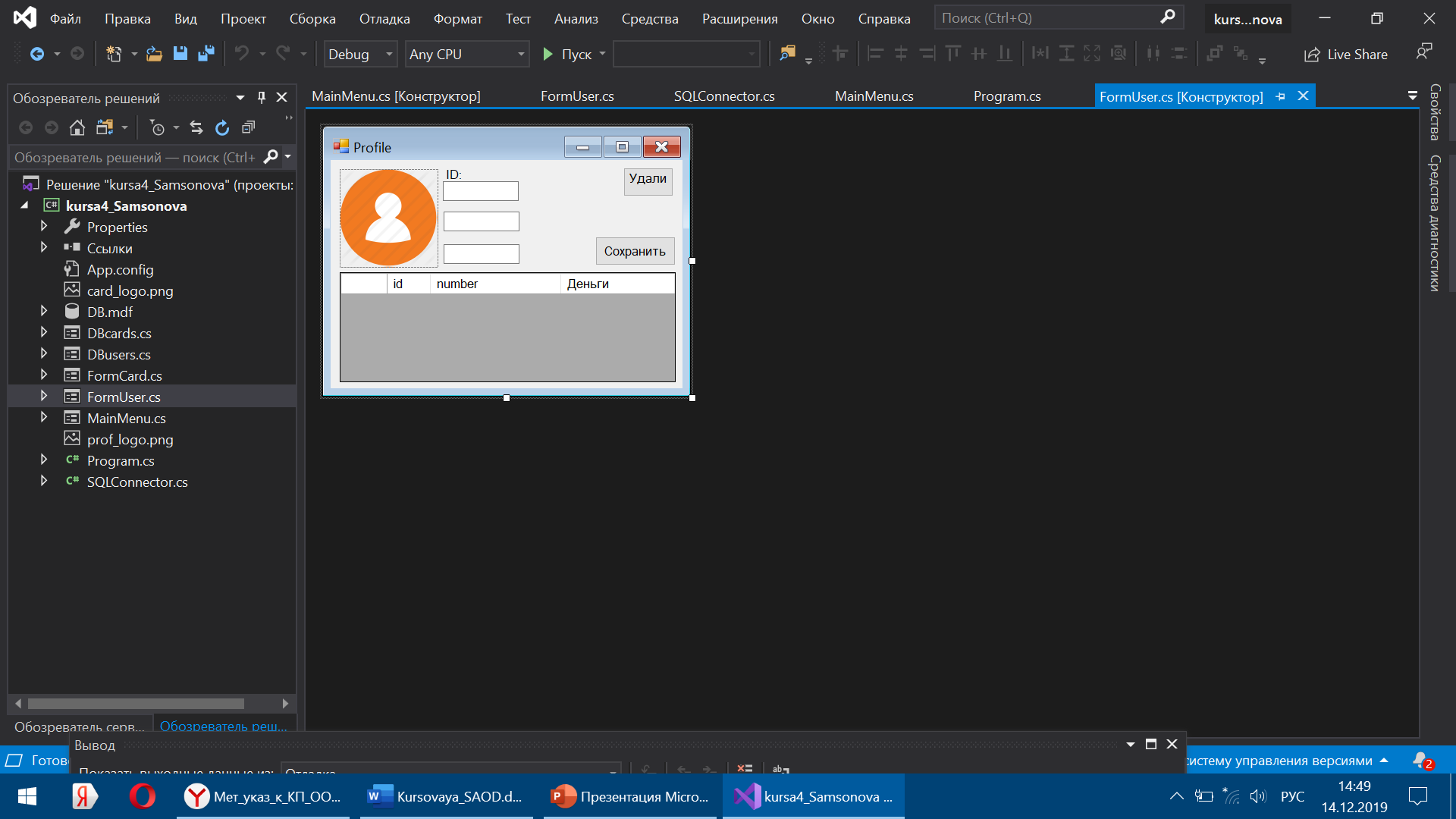


Рис.4. Профиль пользователя.

Нажав на «Карты», перед нами откроется окно с id карты, номером пользователя, баланс пользователя.

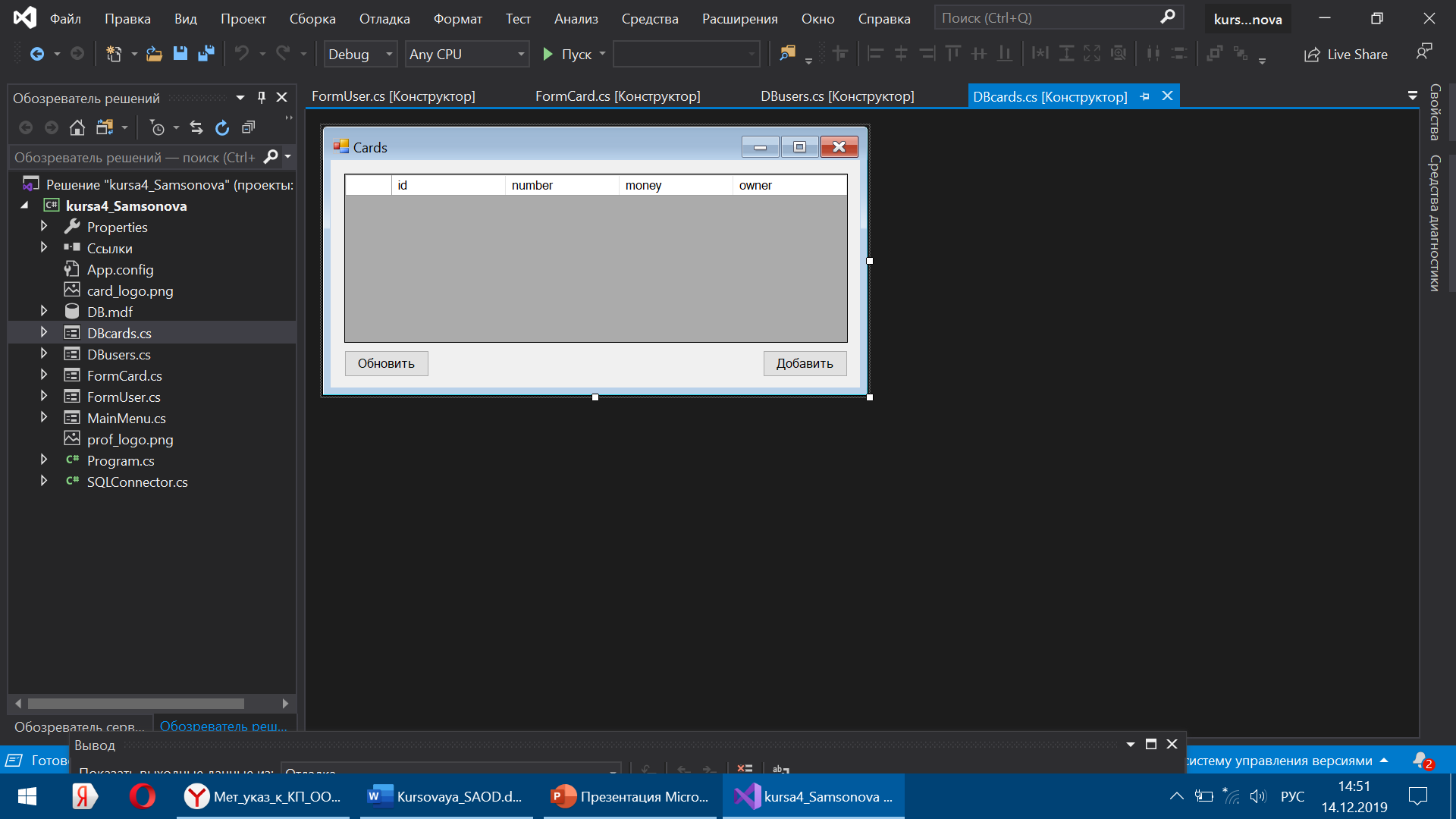


Рис.5. Карты.

После добавления карты в базу данных, будет возможность нажать на карту и в отдельном окне будет отображаться окно с возможностью добавления пользователя и работы с его счётом.

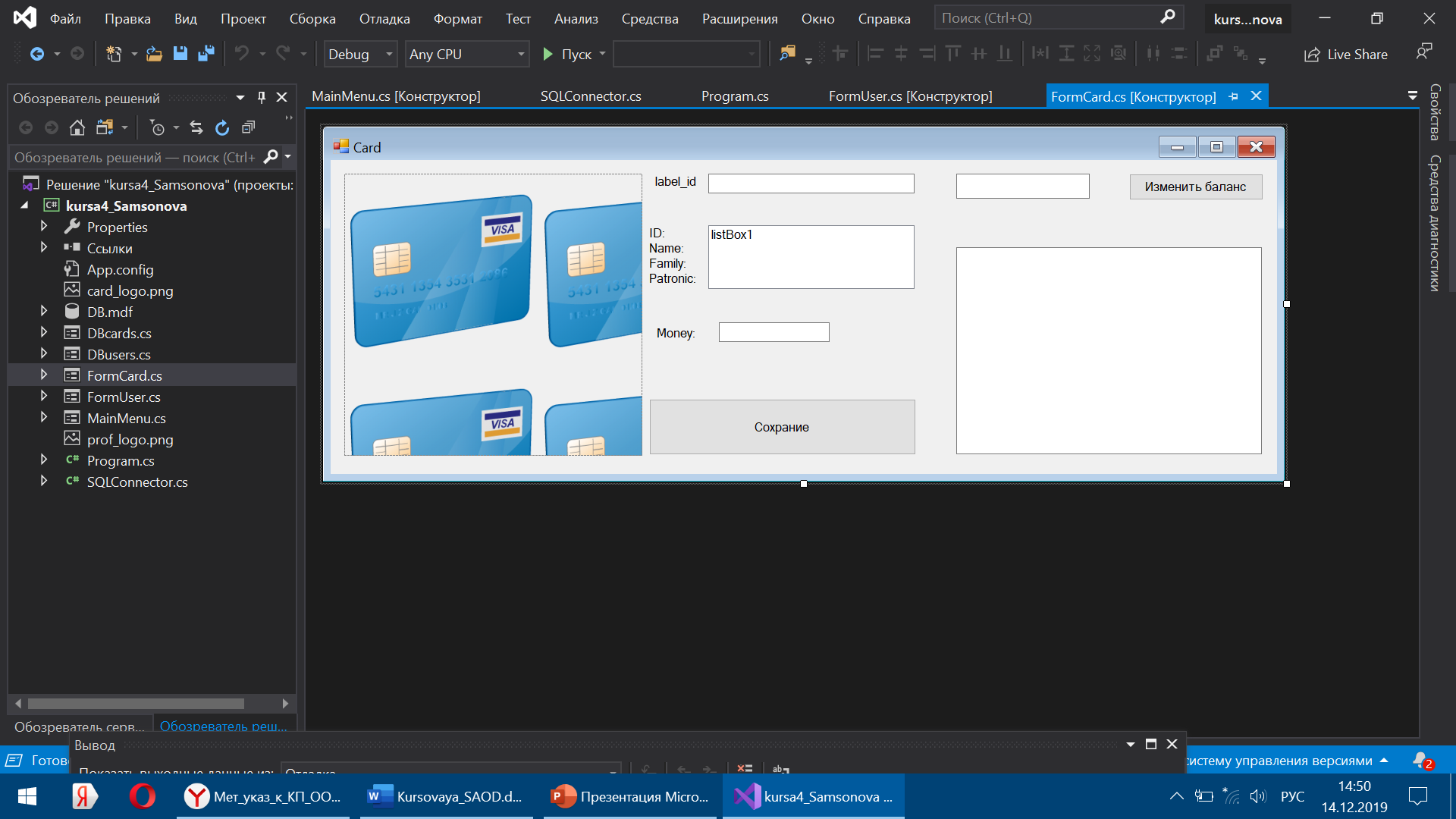


Рис.5. Информация о картах.

**3.2. РЕАЛИЗАЦИЯ КЛАССОВ**

**DBcards.cs**

1. Конструктор, в котором происходит инициализация компонентов и загрузка данных из Базы Данных:

public DBcards()

1. Загрузка данных из Базы Данных:

public async void LoadData()

1. Обновление данных из Базы Данных:

private void generate\_Click\_1(object sender, EventArgs e)

1. Вызов формы для добавления карты:

private void btn\_add\_Click(object sender, EventArgs e)

1. Вызов окна с информацией о карте пользователя с дальнейшим изменением и редактированием данных:

private void cardgrid\_CellClick(object sender, DataGridViewCellEventArgs e)

**DBusers.cs**

1. Конструктор для инициализации компонентов:

public DBusers()

1. Конструктор с параметром для инициализации компонентов:

public DBusers(out uint tempid)

1. Обработчик событий нажатия на ячейку таблицы.

Отвечает за вызов формы пользователя:

private void usergrid\_CellClick\_1(object sender, DataGridViewCellEventArgs

1. Загрузка данных из Базы Данных:

private async void LoadData()

1. Вызывает форму при добавлении пользователя:  
   private void btn\_addUser\_Click(object sender, EventArgs e)
2. Обновляет данные таблицы:

private void btn\_refresh\_Click(object sender, EventArgs e)

**FormCard.cs**

1. Конструктор для инициализации компонентов:

public FormCard()

1. Конструктор с параметрами для инициализации компонентов и загрузка данных с Базой Данных:

public FormCard(uint id)

1. Отправка информации о карте в Базе Данных:

private async void but\_save\_Click(object sender, EventArgs e)

1. Вызов окна для смены пользователя:

private async void listBox1\_DoubleClick(object sender, EventArgs e)

1. Получение данных о карте и её пользователе:

private async void LoadData()

1. Изменение баланса:

private void change\_balance\_Click(object sender, EventArgs e)

**FormUser.cs**

1. Конструктор для инициализации компонентов:

public FormUser(uint id)

1. Конструктор с параметрами для инициализации компонентов:

public FormUser()

1. Загрузка данных с БД о пользователе и его картах:

private async void LoadData()

1. Обновление данных о пользователе:

private async void btn\_save\_Click(object sender, EventArgs e)

1. Удаление пользователя и его карт:

private async void btn\_del\_Click(object sender, EventArgs e)

1. Вызов окна для редактирования информации о картах:

private void cardgrid\_CellContentClick(object sender, DataGridViewCellEventArgs e)

**SQLConnector.cs**

1. Строка для подключения к Базе Данных:

static string connectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\алексей\Desktop\алексей курсач\kursa4\_Dotsenko\kursa4\_Dotsenko\DB.mdf;Integrated Security=True;Connect Timeout=30";

1. Выполнение sql запроса и возврат полученных данных:

public async Task<SqlDataReader> ExecuteQuery(string query)

1. ТЕСТИРОВАНИЕ

**4.1 СКРИНШОТЫ ВЫПОЛНЕНИЯ ПРОГРАММЫ**

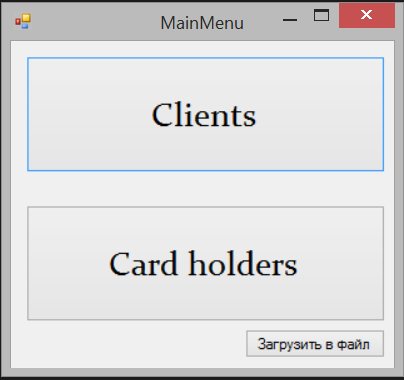


Рис.1. Меню.

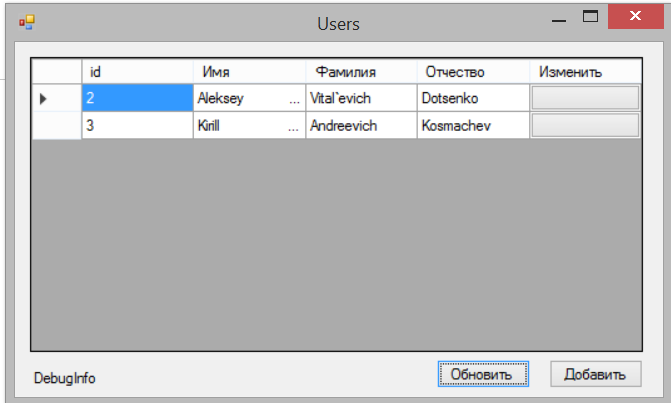


Рис.2. База пользователей.

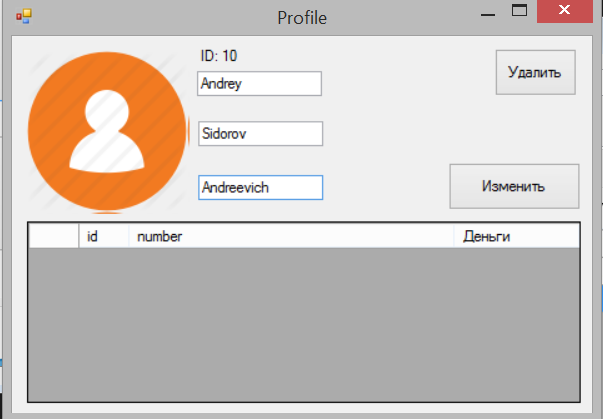


Рис.3. Добавление нового пользователя.

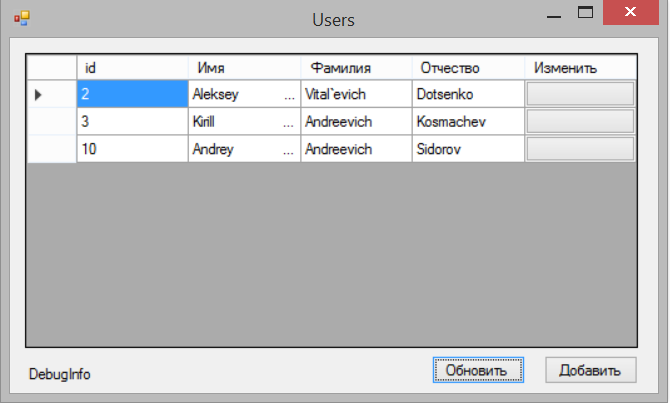


Рис.4. База данных после добавления пользователя. Обновление списка.

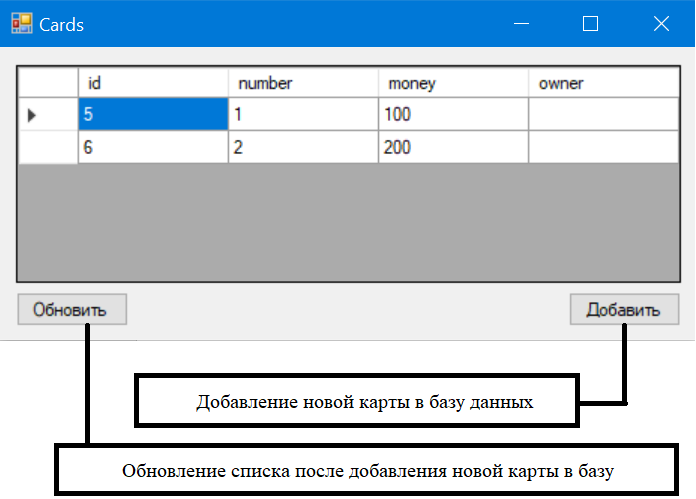


Рис.5. База карт.

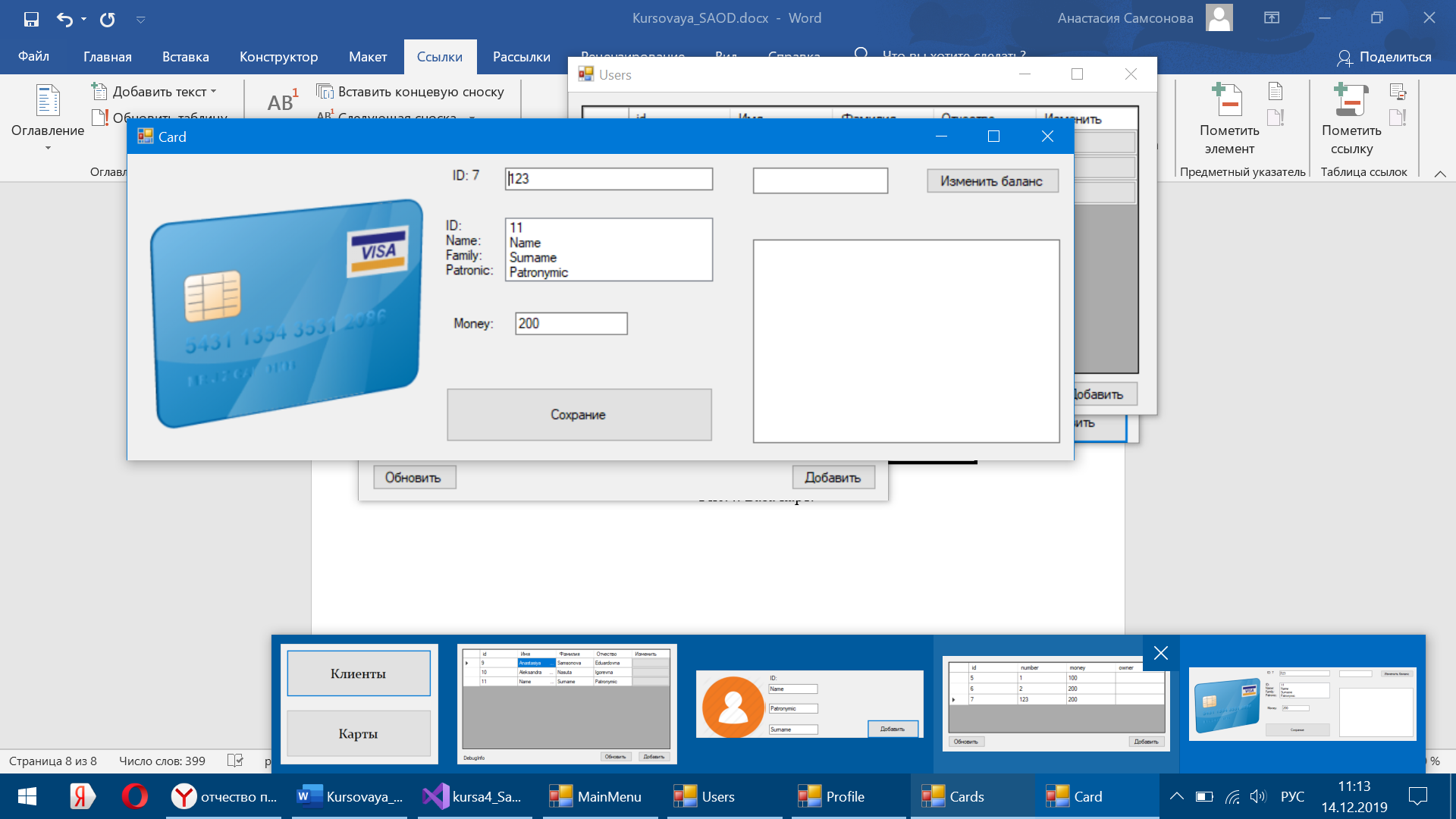


Рис.6. Добавление новой карты.

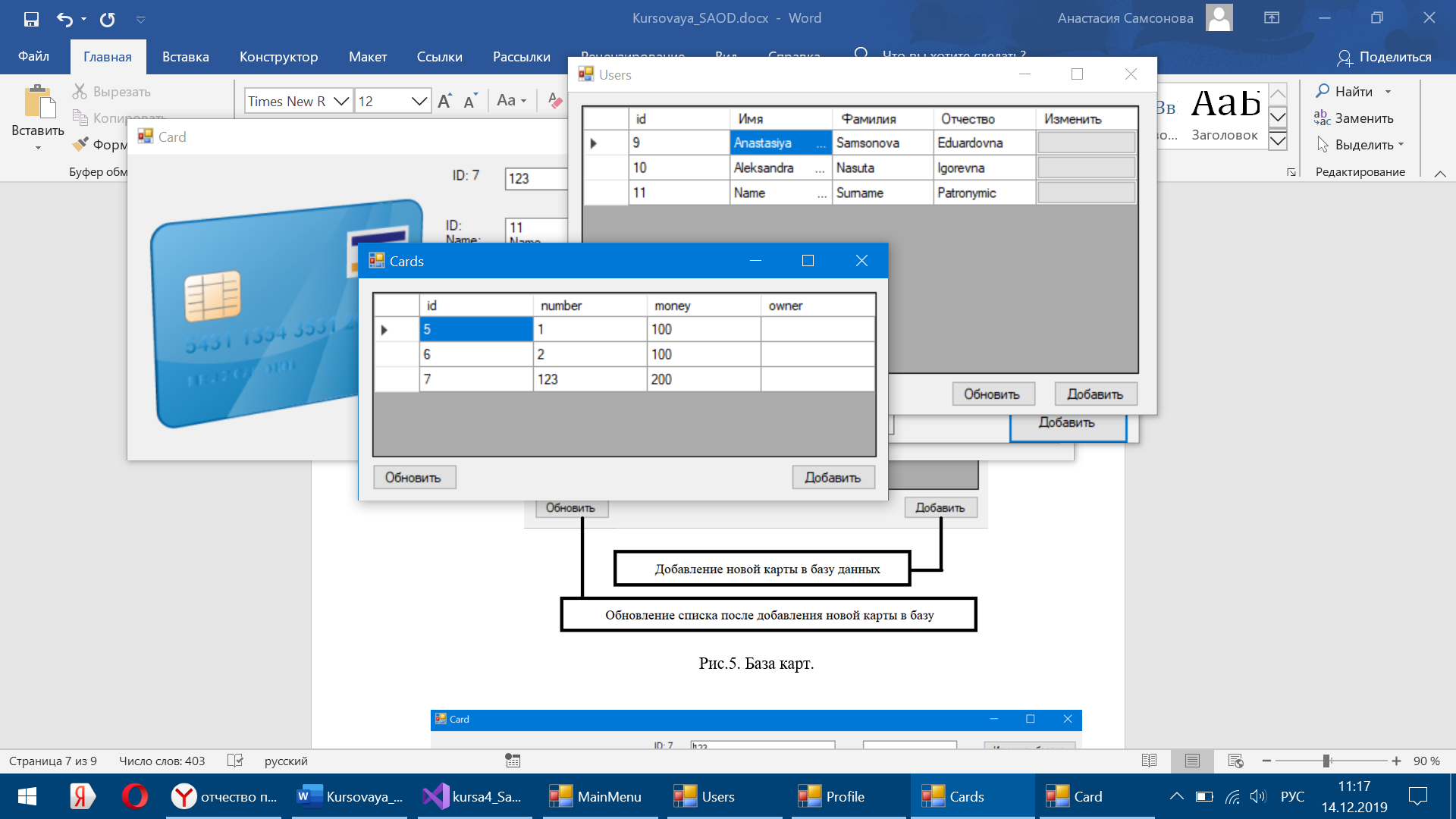


Рис.7. База данных после добавления карты. Обновление списка.

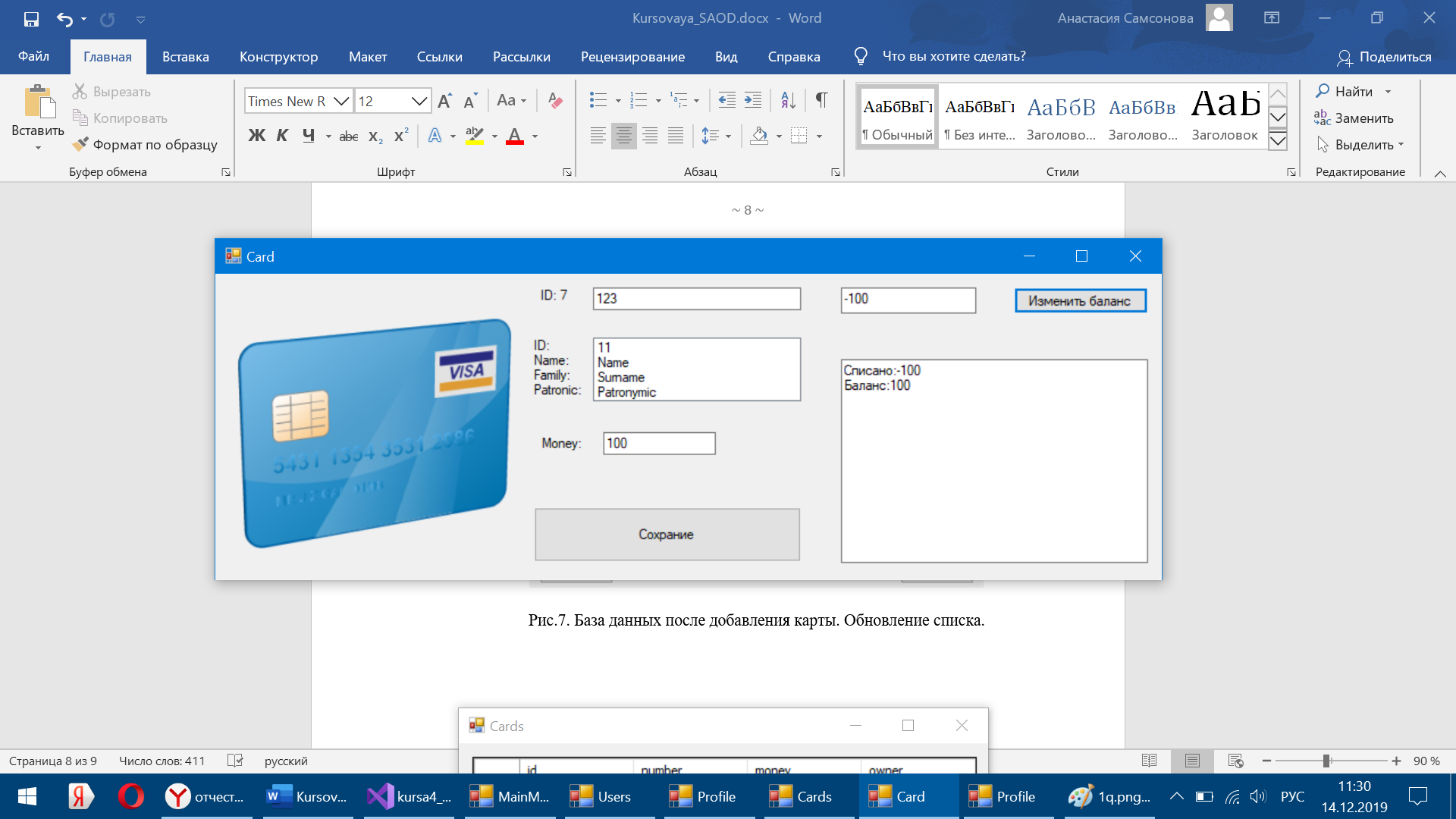


Рис.6. Вычитание денег со счёта пользователя.

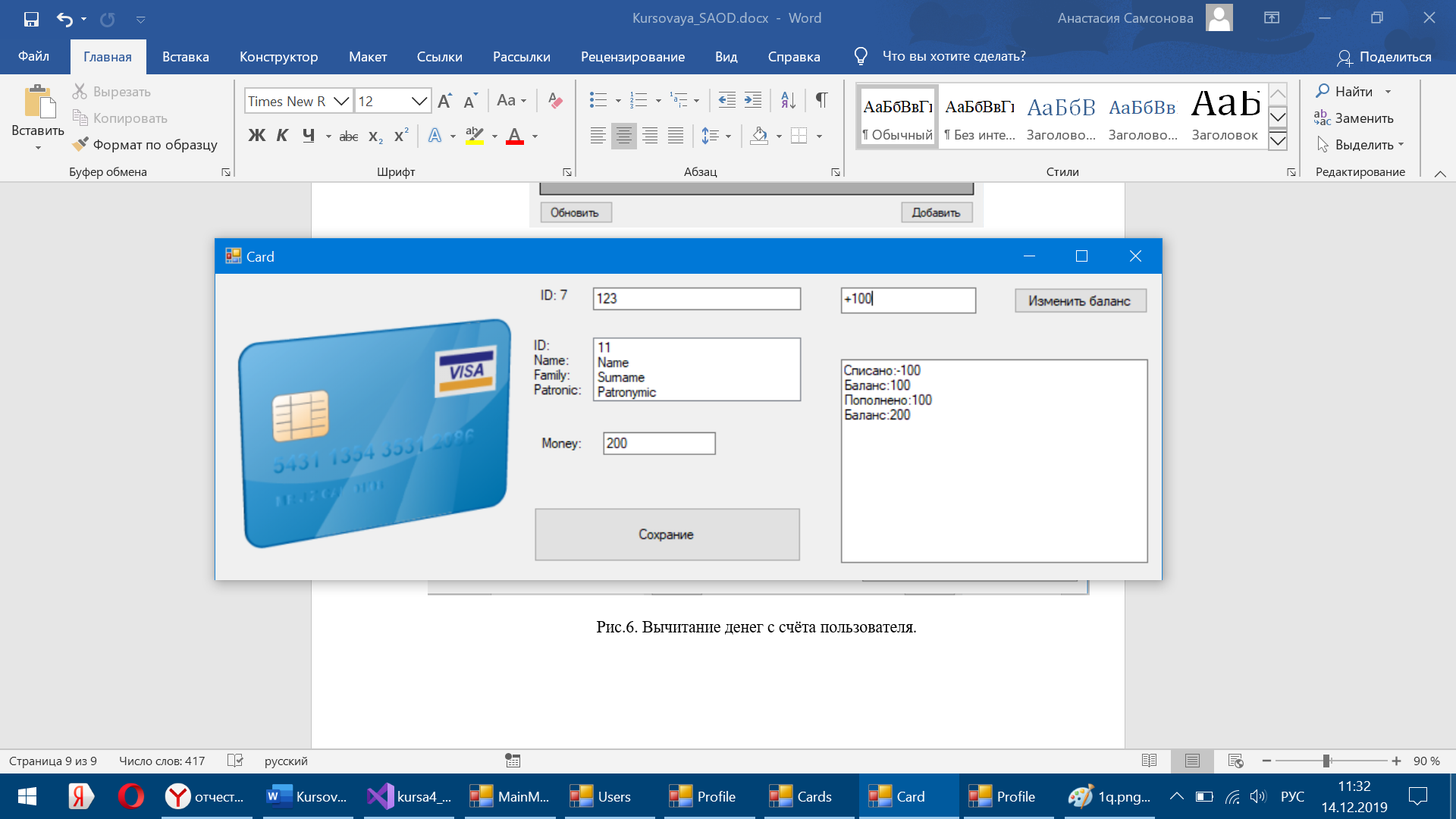


Рис.7.1. Добавление денег на счёт пользователя.

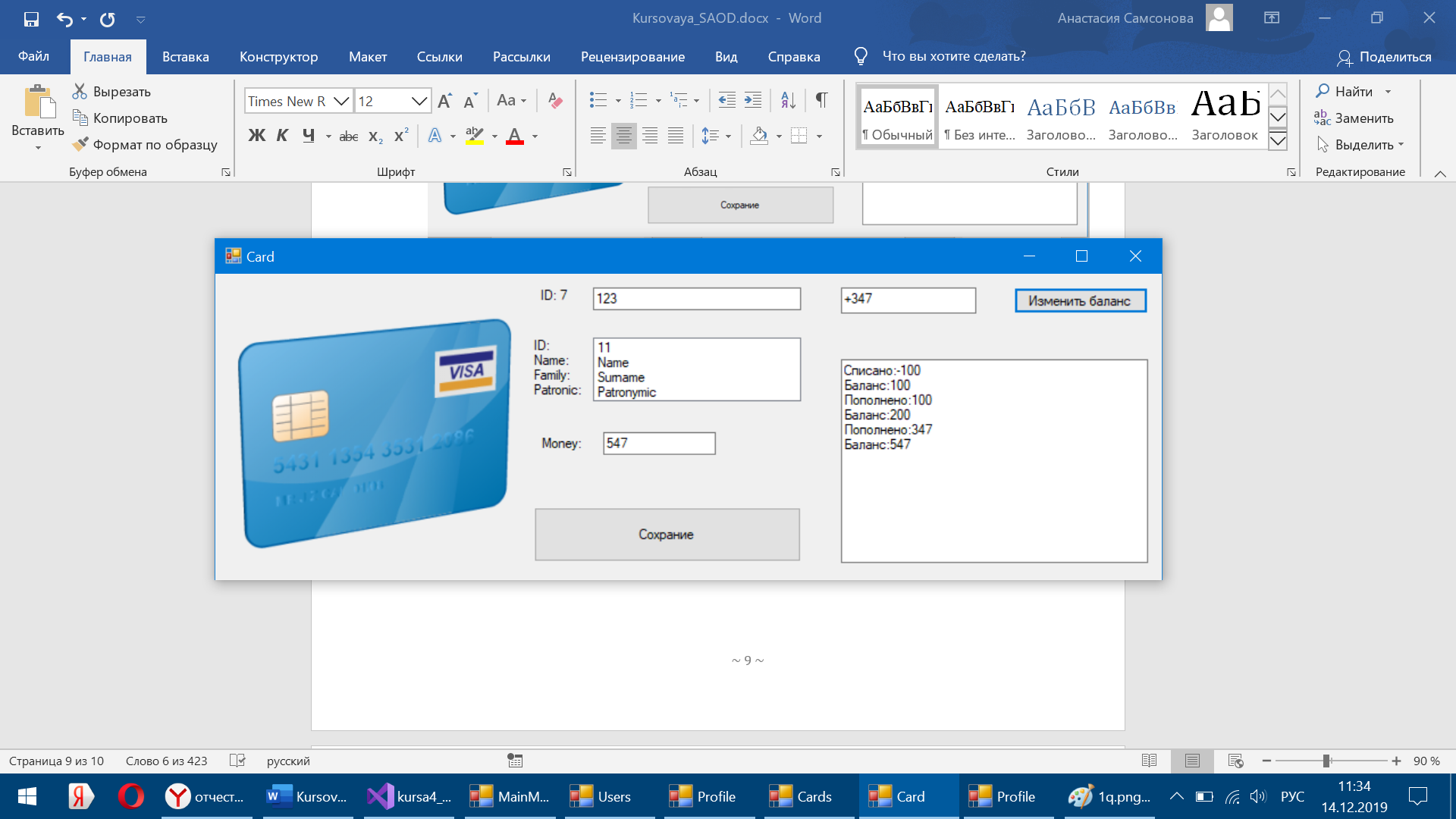


Рис.7.2. Добавление денег на счёт пользователя.

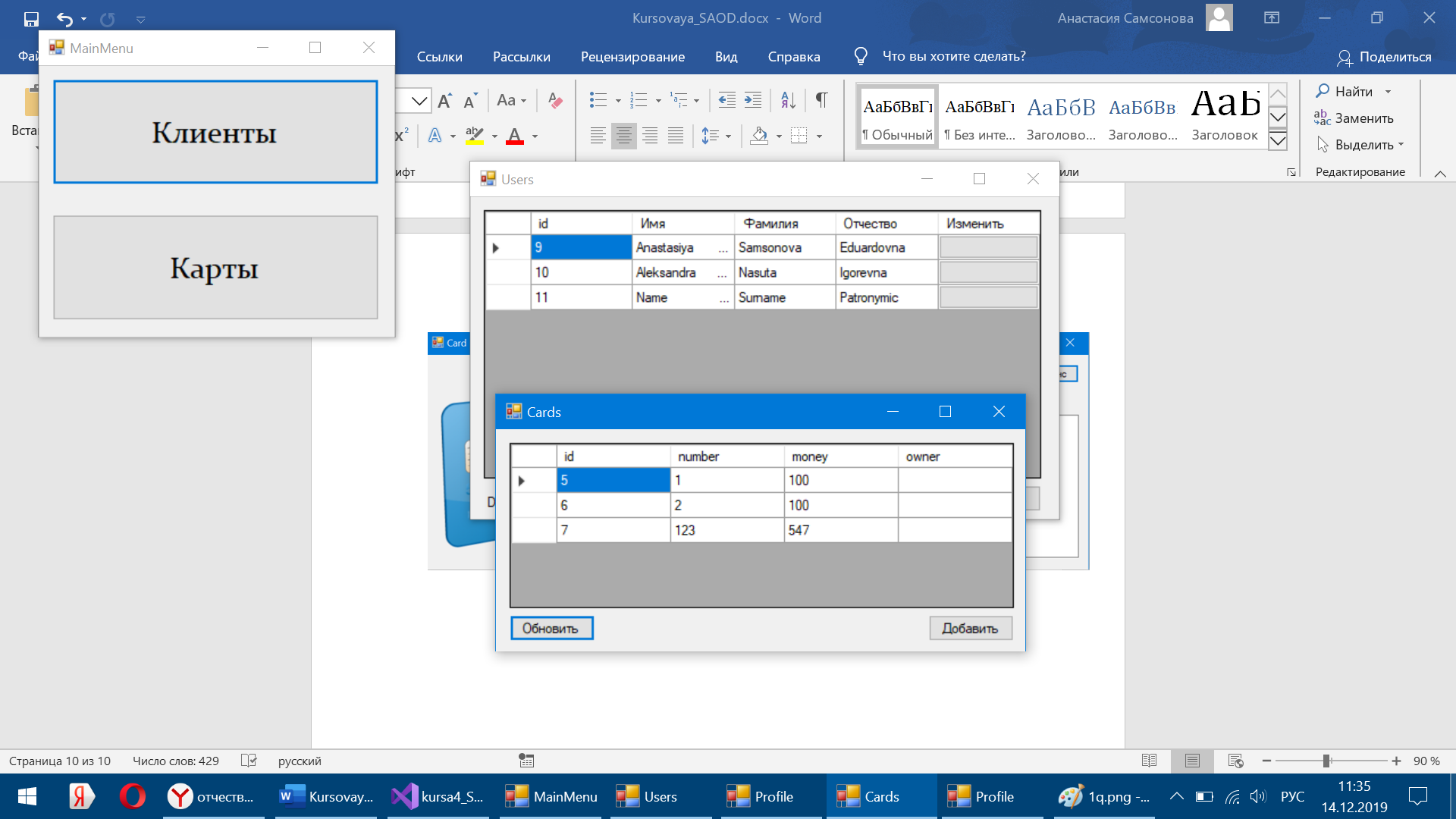


Рис.8. База данных после изменения баланса на карте. Обновление списка.

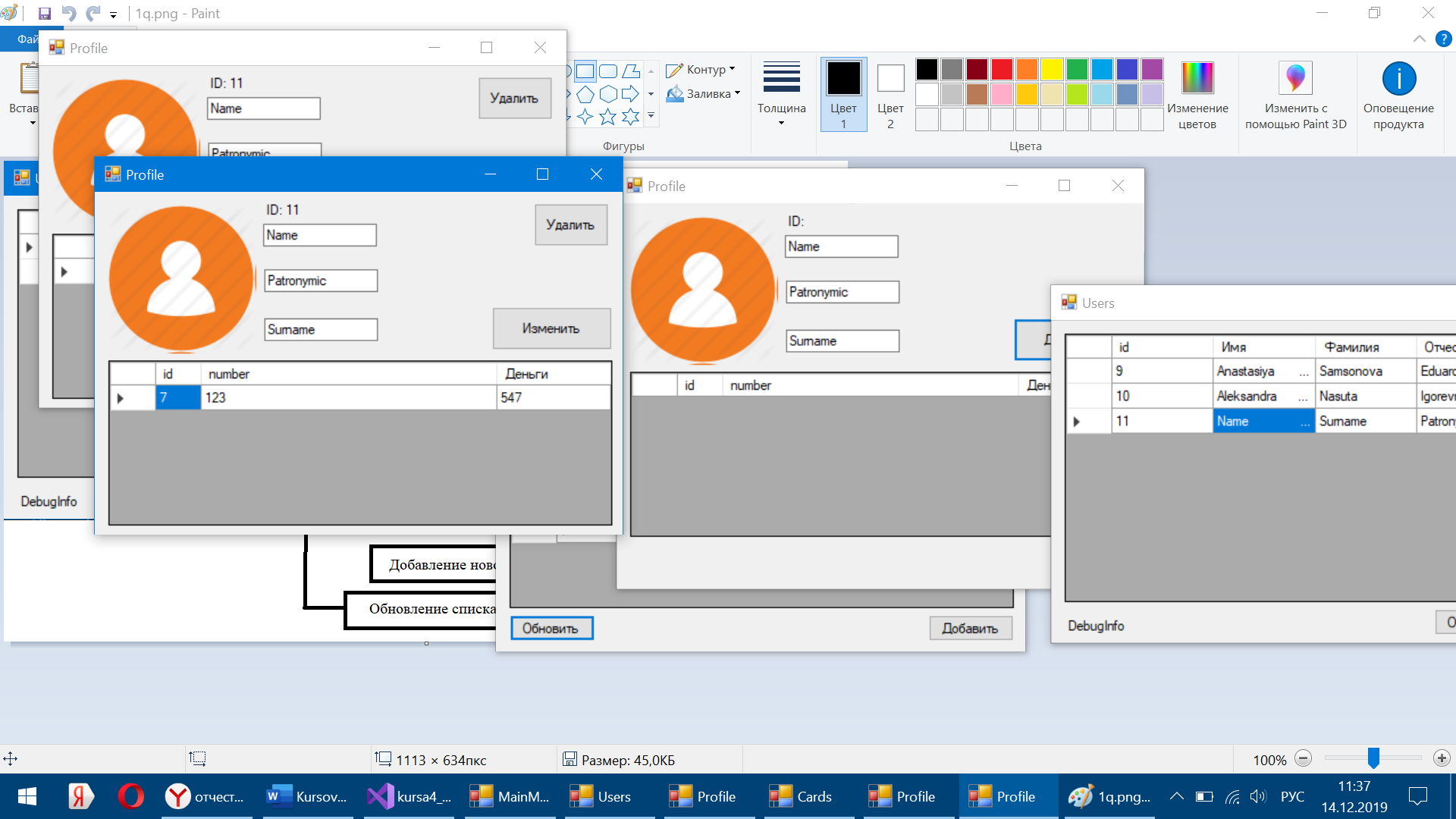


Рис.9. Информация о пользователе после работы с его счётом. Обновление данных.

### ЗАКЛЮЧЕНИЕ

**5.1 ВЫВОД**

В ходе выполнения курсового проекта было разработано приложение для организации взаимодействия объектов при заданных критериях «Банковский счёт».

В качестве среды разработки была выбрана Microsoft Visual Studio 2019. Приложение полностью написано на высокоуровневом языке программирования C# с использованием структурированного языка запросов SQL. Все операции с базой данных выполняются с использованием языка SQL. В ходе написания проекта использовался объектно-ориентированный подход к программированию, что позволило качественно структурировать код и в значительной мере облегчить процесс разработки и поддержки программного продукта.

Во время реализации программного обеспечения освоен общий методологический подход, используемый при проектировании и программной реализации системы классов, соответствующей объектно-ориентированной парадигме программирования. Спроектированная система классов реализована в виде программы и протестирована. Разработано приложение, функциональные возможности которой демонстрируют различные варианты использования объектов созданных классов.

### **5.2 БИБЛИОГРАФИЧЕСКИЙ СПИСОК**

1. Шпаргалка по шаблонам проектирования [Электронный ресурс]: Хабрахабр. – TM, 2006-2018.
2. Объектно-ориентированный анализ и проектирование с примерами приложений / Грэди Б. – 3-е изд. – М.: ИД Вильямс, 2010. – 720 с.: с ил.
3. Приемы объектно-ориентированного проектирования. Паттерны проектирования. / Гамма Э. – отдельное изд. – П.: ИД Питер, 2016. – 366с.: с ил.
4. Е. О. Шумова / Объектно-ориентированное программирование: Методические указания к выполнению курсового проекта - ГУАП. -СПб., 2019. 14 с.
5. [Электронный ресурс]: Документация Microsoft Docs / URL: https://docs.microsoft.com/ru-ru/dotnet/csharp/
   1. **ПРИЛОЖЕНИЕ**

**КОД ПРОГРАММЫ**

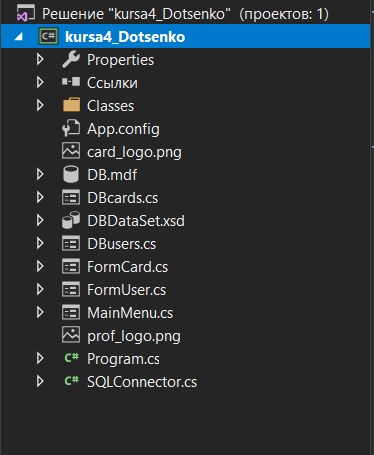


Рис.10. Обозреватель решений. Расширения.

**App.config**

<?xml version="1.0" encoding="utf-8" ?>

<configuration>

<configSections>

</configSections>

<connectionStrings>

<add name="kursa4\_Dotsenko.Properties.Settings.masterConnectionString"

connectionString="Data Source=DESKTOP-1ATNRMB\SQLEXPRESS;Initial Catalog=master;Integrated Security=True"

providerName="System.Data.SqlClient" />

<add name="kursa4\_Dotsenko.Properties.Settings.DbCardConnectionString"

connectionString="Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\DbCard.mdf;Integrated Security=True;Connect Timeout=30"

providerName="System.Data.SqlClient" />

<add name="kursa4\_Dotsenko.Properties.Settings.DBConnectionString"

connectionString="Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\DB.mdf;Integrated Security=True;Connect Timeout=30"

providerName="System.Data.SqlClient" />

</connectionStrings>

<startup>

<supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2" />

</startup>

</configuration>

**DBcards.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace kursa4\_Dotsenko

{

public partial class DBcards : Form

{

public DBcards()

{

InitializeComponent();

LoadData();

}

public async void LoadData()

{

cardgrid.Rows.Clear();

string q = "SELECT \* FROM [Cards]";

SqlDataReader sqlReader = await new SQLConnector().ExecuteQuery(q);

while (await sqlReader.ReadAsync())

{

Console.WriteLine(Convert.ToString(sqlReader["Id"]) + " " + Convert.ToString(sqlReader["number"]) +" "+ Convert.ToString(sqlReader["money"]));

cardgrid.Rows.Add(Convert.ToString(sqlReader["Id"]), Convert.ToString(sqlReader["number"]), Convert.ToString(sqlReader["money"]));

}

}

private void generate\_Click\_1(object sender, EventArgs e)

{

LoadData();

}

private void btn\_add\_Click(object sender, EventArgs e)

{

FormCard newForm = new FormCard();

// this.Hide();

newForm.Show();

}

private void cardgrid\_CellClick(object sender, DataGridViewCellEventArgs e)

{

if (e.RowIndex == -1) { }

else

{

FormCard newForm = new FormCard(Convert.ToUInt32(cardgrid[0, e.RowIndex].Value));

// this.Hide();

newForm.Show();

}

}

private void cardgrid\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

}

}

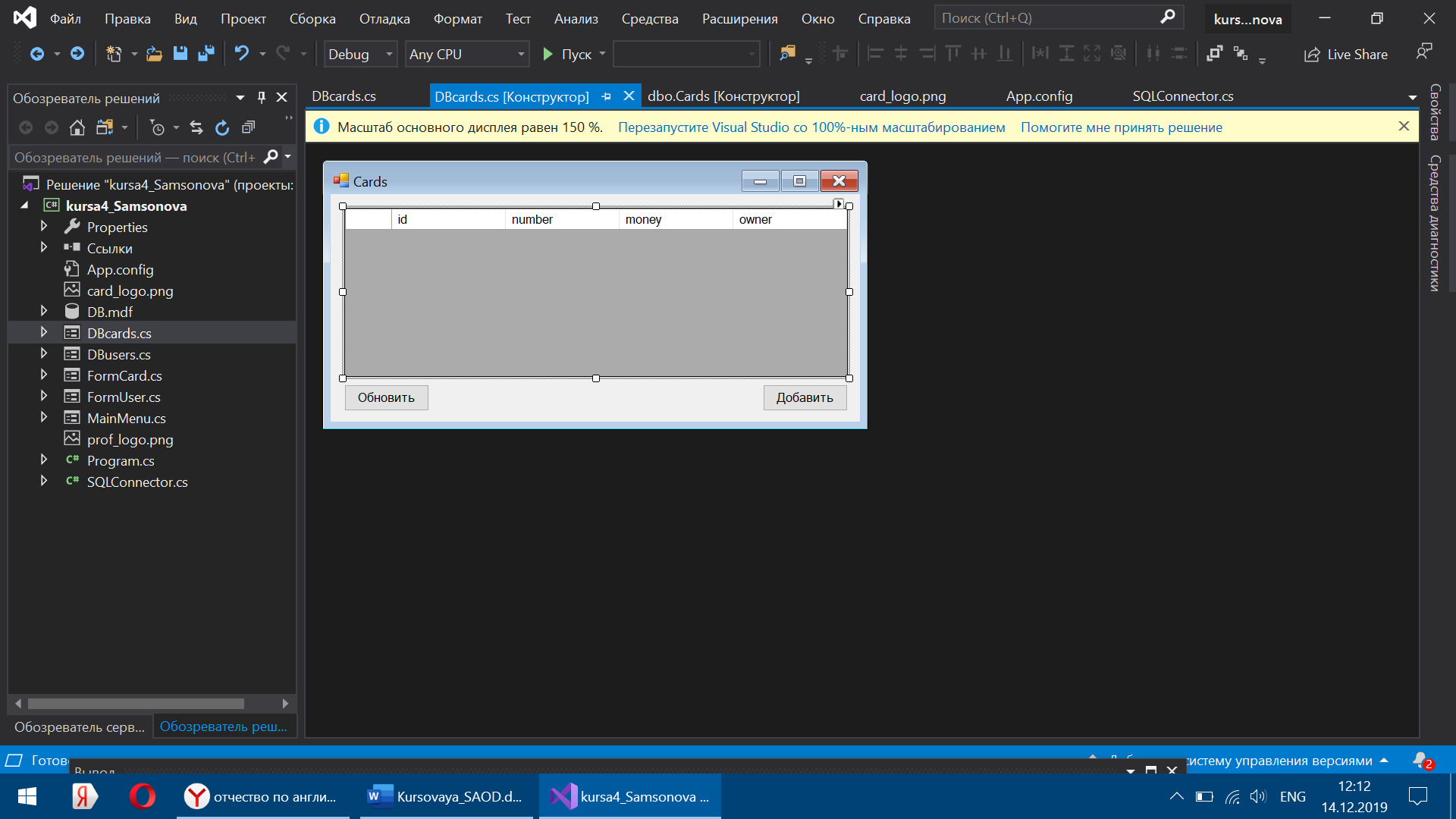


Рис.11. DBcards.cs.

**DBusers.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static kursa4\_Dotsenko.DBcards;

using static kursa4\_Dotsenko.FormUser;

namespace kursa4\_Dotsenko

{

public partial class DBusers : Form

{

bool ISchoice = false;

public DBusers()

{

InitializeComponent();

}

public uint local\_tempid=0;

public DBusers(out uint tempid)

{

tempid= local\_tempid;

ISchoice = true;

InitializeComponent();

}

private void label1\_Click(object sender, EventArgs e)

{

}

private void usergrid\_CellClick\_1(object sender, DataGridViewCellEventArgs e)

{

if (ISchoice)

{

local\_tempid = Convert.ToUInt32(usergrid[0, e.RowIndex].Value);

this.Close();

}

else

{

FormUser newForm = new FormUser(Convert.ToUInt32(usergrid[0, e.RowIndex].Value));

// this.Hide();

newForm.Show();

}

}

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

{

LoadData();

}

private async void LoadData()

{

usergrid.Rows.Clear();

string q = "SELECT \* FROM [USERS]";

SqlDataReader sqlReader = await new SQLConnector().ExecuteQuery(q);

while (await sqlReader.ReadAsync())

{

Console.WriteLine(Convert.ToString(sqlReader["Id"]) + " " + Convert.ToString(sqlReader["name"]) + " " + Convert.ToString(sqlReader["family"]) + " " + Convert.ToString(sqlReader["patronic"]));

usergrid.Rows.Add(Convert.ToString(sqlReader["Id"]), Convert.ToString(sqlReader["name"]), Convert.ToString(sqlReader["family"]), Convert.ToString(sqlReader["patronic"]));

//userList.Add(new User(Convert.ToUInt32(sqlReader["Id"]), Convert.ToString(sqlReader["name"]), Convert.ToString(sqlReader["family"]), Convert.ToString(sqlReader["patronic"])));

}

}

private void btn\_addUser\_Click(object sender, EventArgs e)

{

FormUser newForm = new FormUser();

// this.Hide();

newForm.Show();

}

private void btn\_refresh\_Click(object sender, EventArgs e)

{

LoadData();

}

private void usergrid\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

}

}

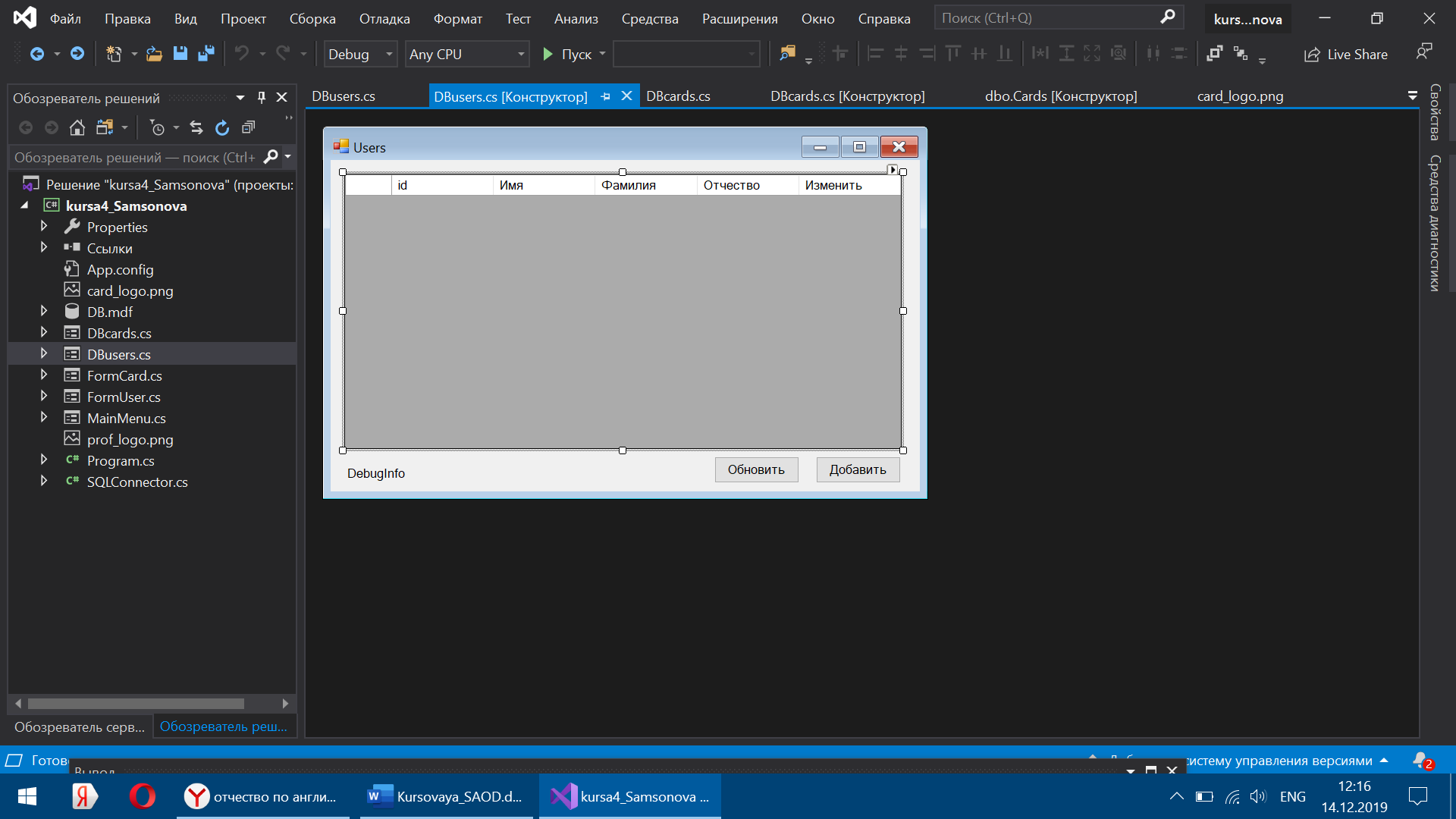


Рис.11. DBusers.cs.

**FormCard.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace kursa4\_Dotsenko

{

public partial class FormCard : Form

{

uint id\_card;

public FormCard()

{

InitializeComponent();

}

public FormCard(uint id)

{

id\_card = id;

InitializeComponent();

LoadData();

}

private async void but\_save\_Click(object sender, EventArgs e)

{

if (id\_card == 0)

{

string q = "INSERT INTO [Cards] (number,money,history,user\_id)VALUES(";

q += '\'' + textBox\_number.Text + "\',";

q += '\'' + textBox\_money.Text + "\',";

q += '\'' + textBox\_history.Text.ToString() + "\',";

if (listBox1.Items.Count > 0)

q += '\'' + listBox1.Items[0].ToString() + "\')";

await new SQLConnector().ExecuteQuery(q);

}

else

{

string q = "UPDATE[Cards] SET number = \'" +

Convert.ToString(textBox\_number.Text) +

"\', user\_id = \'" + listBox1.Items[0].ToString() +

"\', money = \'" + textBox\_money.Text +

"\', history = \'" + textBox\_history.Text.ToString() +

"\' WHERE id = " + id\_card;

await new SQLConnector().ExecuteQuery(q);

}

this.Close();

}

private async void listBox1\_DoubleClick(object sender, EventArgs e)

{

DBusers newForm = new DBusers(out uint id);

newForm.ShowDialog();

id = newForm.local\_tempid;

listBox1.Text = "";

string q = "SELECT \* FROM [USERS] WHERE [id]=" + id;

SqlDataReader sqlReader = await new SQLConnector().ExecuteQuery(q);

while (await sqlReader.ReadAsync())

{

listBox1.Items.Add(Convert.ToString(sqlReader["id"]));

listBox1.Items.Add(Convert.ToString(sqlReader["name"]));

listBox1.Items.Add(Convert.ToString(sqlReader["family"]));

listBox1.Items.Add(Convert.ToString(sqlReader["patronic"]));

}

}

private async void LoadData()

{

string user\_id = "";

string q = "SELECT \* FROM [CARDS] WHERE id=" + id\_card.ToString();

Console.WriteLine("Run Query " + q);

using (SqlDataReader sqlReader = await new SQLConnector().ExecuteQuery(q))

{

while (await sqlReader.ReadAsync())

{

user\_id = Convert.ToString(sqlReader["user\_id"]);

label1.Text = "ID: " + id\_card.ToString();

textBox\_number.Text = Convert.ToString(sqlReader["number"]);

textBox\_money.Text = Convert.ToString(sqlReader["money"]);

textBox\_history.Text = Convert.ToString(sqlReader["history"]);

}

}

q = "SELECT \* FROM [USERS] WHERE id = " + user\_id;

Console.WriteLine("Run Query " + q);

using (SqlDataReader sqlReader = await new SQLConnector().ExecuteQuery(q))

{

while (await sqlReader.ReadAsync())

{

listBox1.Items.Add(Convert.ToString(sqlReader["id"]));

listBox1.Items.Add(Convert.ToString(sqlReader["name"]));

listBox1.Items.Add(Convert.ToString(sqlReader["family"]));

listBox1.Items.Add(Convert.ToString(sqlReader["patronic"]));

}

}

}

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

{

if (id\_card > 0)

{

}

}

private void change\_balance\_Click(object sender, EventArgs e)

{

int money = 0;

if(Int32.TryParse(textBox\_balance\_change.Text.ToString(),out money))

{

int balance = Convert.ToInt32(textBox\_money.Text);

balance += money;

if (money > 0)

textBox\_history.Text += "Пополнено:" + money + "\r\n";

if (money < 0)

textBox\_history.Text += "Списано:" + money + "\r\n";

textBox\_history.Text += "Баланс:" + balance + "\r\n";

textBox\_money.Text = balance.ToString();

}

}

}

}

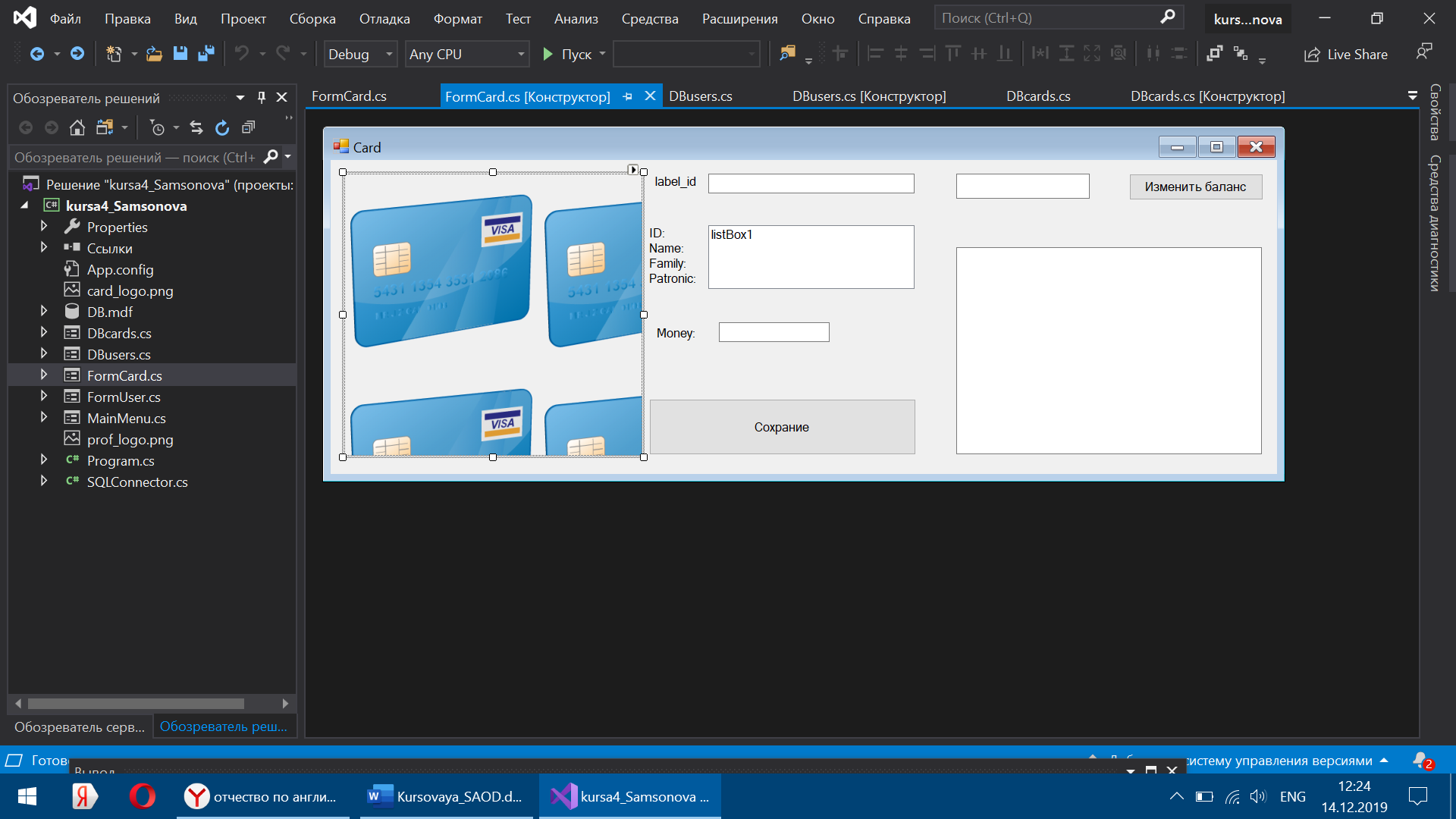


Рис.12. FormCard.cs.

**FormUser.cs**

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 static kursa4\_Dotsenko.DBusers;

using static kursa4\_Dotsenko.FormUser;

using static kursa4\_Dotsenko.DBcards;

using System.Data.SqlClient;

namespace kursa4\_Dotsenko

{

public partial class FormUser : Form

{

uint id\_user;

public FormUser(uint id)

{

id\_user = id;

InitializeComponent();

LoadData();

btn\_save.Text = "Изменить";

}

public FormUser()

{

InitializeComponent();

btn\_save.Text = "Добавить";

}

private async void LoadData()

{

string q = "SELECT \* FROM [USERS] WHERE id=" + id\_user.ToString();

Console.WriteLine("Run Query " + q);

using (SqlDataReader sqlReader = await new SQLConnector().ExecuteQuery(q))

{

while (await sqlReader.ReadAsync())

{

label\_id.Text = "ID: " + id\_user.ToString();

Console.WriteLine(Convert.ToString(sqlReader["name"]));

textBox\_name.Text = Convert.ToString(sqlReader["name"]);

textBox\_family.Text = Convert.ToString(sqlReader["family"]);

textBox\_patronymic.Text = Convert.ToString(sqlReader["patronic"]);

btn\_del.Visible = true;

}

}

q = "SELECT \* FROM [CARDS] WHERE user\_id = " + id\_user.ToString();

Console.WriteLine("Run Query " + q);

using (SqlDataReader sqlReader = await new SQLConnector().ExecuteQuery(q))

{

while (await sqlReader.ReadAsync())

{

cardgrid.Rows.Add(Convert.ToString(sqlReader["id"]), Convert.ToString(sqlReader["number"]), Convert.ToString(sqlReader["money"]));

}

}

}

private async void btn\_save\_Click(object sender, EventArgs e)

{

if (id\_user == 0)

{

string q = "INSERT INTO [USERS] (name,family,patronic)VALUES(";

q += '\'' + textBox\_name.Text + "\',";

q += '\'' + textBox\_family.Text + "\',";

q += '\'' + textBox\_patronymic.Text + "\')";

await new SQLConnector().ExecuteQuery(q);

}

else

{

string q = "UPDATE[USERS] SET name = \'" +

Convert.ToString(textBox\_name.Text) +

"\', family = \'" + Convert.ToString(textBox\_family.Text) +

"\', patronic = \'" + Convert.ToString(textBox\_patronymic.Text) +

"\' WHERE id = " + id\_user;

await new SQLConnector().ExecuteQuery(q);

}

}

private async void btn\_del\_Click(object sender, EventArgs e)

{

string q = "DELETE FROM [CARDS] WHERE [user\_id]=";

q += id\_user;

Console.WriteLine(q);

await new SQLConnector().ExecuteQuery(q);

q = "DELETE FROM [USERS] WHERE [Id]=";

q += id\_user;

Console.WriteLine(q);

await new SQLConnector().ExecuteQuery(q);

this.Close();

}

private void label\_id\_Click(object sender, EventArgs e)

{

}

private void cardgrid\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

FormCard newForm = new FormCard(Convert.ToUInt32(cardgrid[0, e.RowIndex].Value));

// this.Hide();

newForm.Show();

}

}

}

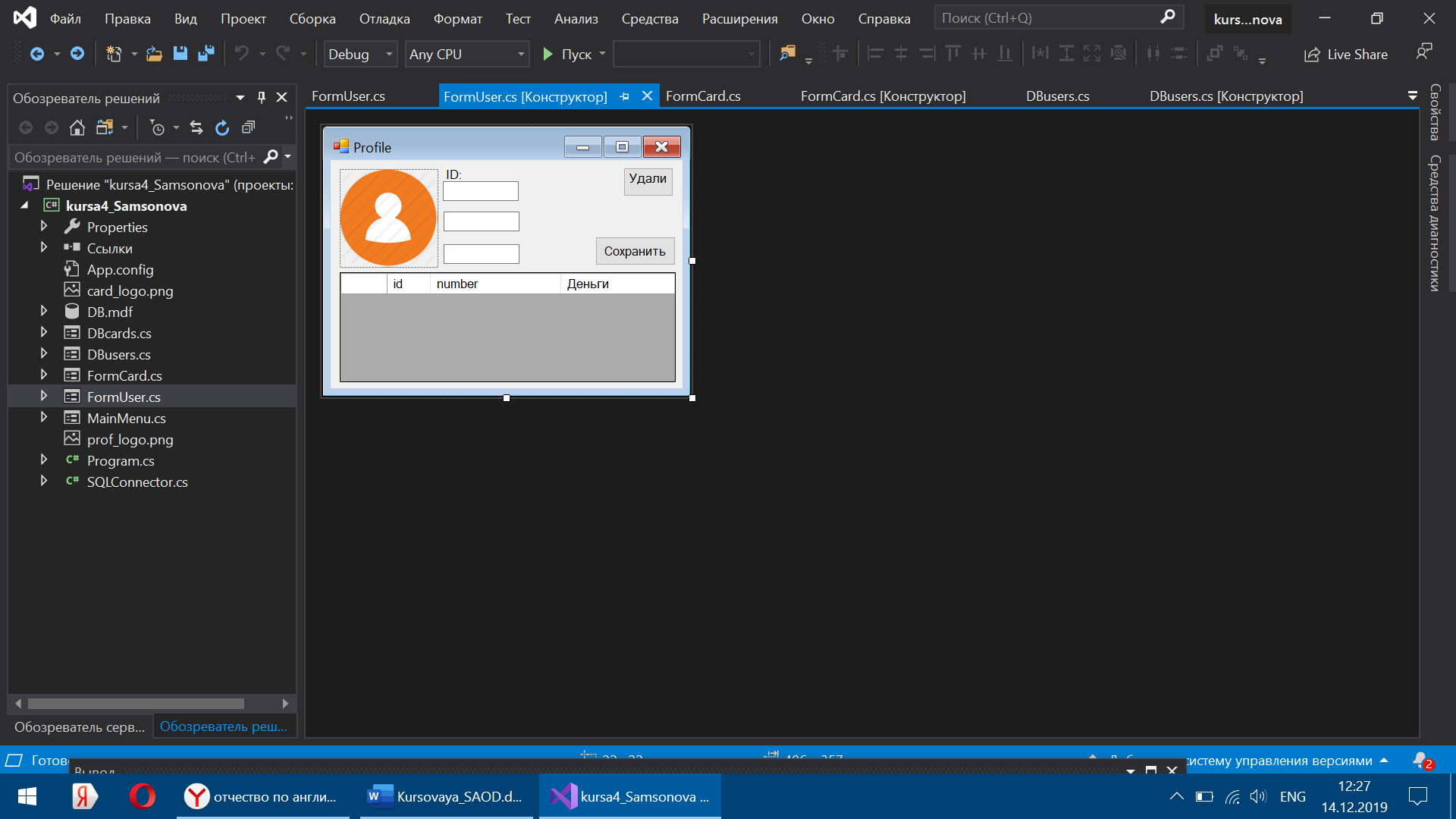


Рис.13. FormUser.cs.

**MainMenu.cs**

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;

namespace kursa4\_Dotsenko

{

public partial class MainMenu : Form

{

public MainMenu()

{

InitializeComponent();

but\_usr.Click += But\_usr\_Click;

but\_cards.Click += But\_cards\_Click;

}

private void But\_cards\_Click(object sender, EventArgs e)

{

DBcards newForm = new DBcards();

// this.Hide();

newForm.Show();

// throw new NotImplementedException();

}

private void But\_usr\_Click(object sender, EventArgs e)

{

DBusers newForm = new DBusers();

// this.Hide();

newForm.Show();

// throw new NotImplementedException();

}

}

}

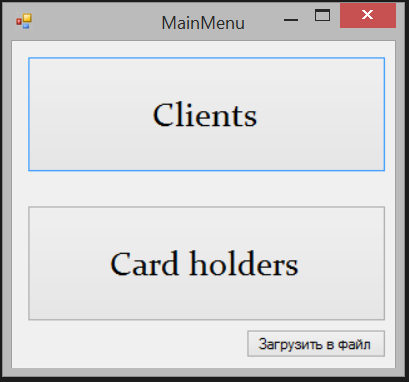


Рис.14. MainMenu.cs.

**C#**

**Program.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace kursa4\_Dotsenko

{

static class Program

{

/// <summary>

/// Главная точка входа для приложения.

/// </summary>

[STAThread]

static void Main()

{

new SQLConnector();

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new MainMenu());

}

}

}

**C#**

**SQLConnector.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.SqlClient;

using System.Data;

using System.Windows.Forms;

namespace kursa4\_Dotsenko

{

class SQLConnector

{

// строка подключения к БД

static string connectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\Алексей\Desktop\Доценко курсовая\kursa4\_Dotsenko\kursa4\_Dotsenko\DB.mdf;Integrated Security=True;Connect Timeout=30";

private SqlConnection con;

public string ConnectionString { get => connectionString; set => connectionString = value; }

public SQLConnector()

{

con = new SqlConnection(connectionString);

Console.WriteLine("Try");

con.Open();

SqlCommand cmd = new SqlCommand();

Console.WriteLine("Connect");

con.Close();

printALL("table\_card");

}

public void printALL(string table)

{

}

public async Task<SqlDataReader> ExecuteQuery(string query)

{

Console.WriteLine("Execute query: " + query);

SqlConnection sqlConnection = new SqlConnection(connectionString);

await sqlConnection.OpenAsync();

SqlDataReader sqlReader = null;

SqlCommand command = new SqlCommand(query, sqlConnection);

try

{

if (query.IndexOf("SELECT") == 0)

{

sqlReader = await command.ExecuteReaderAsync();

return sqlReader;

}

await command.ExecuteNonQueryAsync();

}

catch (Exception ex)

{

Console.WriteLine(ex.Message.ToString());

MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(), MessageBoxButtons.OK, MessageBoxIcon.Error);

}

return null;

}

}

}