МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных Технологий

Кафедра Информационных систем и технологий

Специальность 1-40 01 01 «Информационные системы и технологии»

Специализация Издательско-полиграфический комплекс

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА КУРСОВОГО ПРОЕКТА:**

по дисциплине «Объектно-ориентированное программирование»

Тема «Автоматизация ресторана»

Исполнитель

студент 2 курса 2 группы Гицарев В.А.

(Ф.И.О.)

Руководитель

Ассистент Чевжик Е.А.

(учен. степень, звание, должность, подпись, Ф.И.О.)

Курсовой проект защищен с оценкой

Руководитель Чевжик Е.А.

(подпись)

Минск 2019

Оглавление

[**Введение** 3](#_Toc10028081)

[**1. Обзор аналогов современных электронных диспетчеров таксопарка** 4](#_Toc10028082)

[**2. Проектирование и разработка систем** 7](#_Toc10028083)

[**2.1 Архитектура системы** 7](#_Toc10028084)

[**2.2 Проектирование логической структуры базы данных** 7](#_Toc10028085)

[**2.3 Разработка приложения** 8](#_Toc10028086)

[**2.3.1 Разграничение прав доступа** 8](#_Toc10028087)

[**2.3.2 Структура проекта** 9](#_Toc10028088)

[**2.3.3 Классы** 10](#_Toc10028089)

[**3. Руководство пользователя** 11](#_Toc10028090)

[**3.1 Менеджер** 11](#_Toc10028091)

[**3.2 Официант** 15](#_Toc10028092)

[**4. Тестирование программы** 17](#_Toc10028093)

[**Заключение** 19](#_Toc10028094)

[**Список литературы** 20](#_Toc10028095)

[**Приложение А** 21](#_Toc10028096)

[**Приложение B** 24](#_Toc10028097)

# **Введение**

Данная курсовая работа посвящена разработки WPF приложения «Автоматизация ресторана».

Цель приложения максимально упростить и автоматизировать процессы управления рестораном, такие как составление заказов, управление персоналом и др. Приложение позволяет более удобно хранить данные о заказах, сотрудников, быстрее их обрабатывать, собирать статистику.

В качестве интерфейса прикладного программирования был выбран обширный API-интерфейс — Windows Presentation Foundation (WPF).

Для работы с WPF использовался объектно-ориентированный язык программирования С#.

При разработке приложения будут использоваться:

* Microsoft SQL Server Management Studio 17;
* Visual Studio 2019 (установленный .Net framework 4.6.1);

Для нормального функционирования приложения требуется наличие на персональном компьютере ОС Windows 10.

# **Обзор аналогов современных электронных диспетчеров таксопарка**

Изучив рынок программных средств подобной тематики, я пришел к выводу, что существует достаточно большое количество аналогов, отличающихся функционалом и доступностью. Я воспользовался этим для изучения функционала аналогов.

1.1 Приложение «iiko» (рисунок 1.1).

Система характеризуется наличием большого числа различных модулей, которые подключаются к специальному серверу и работают через него. Есть функциональность фронт- и бэк-офиса.

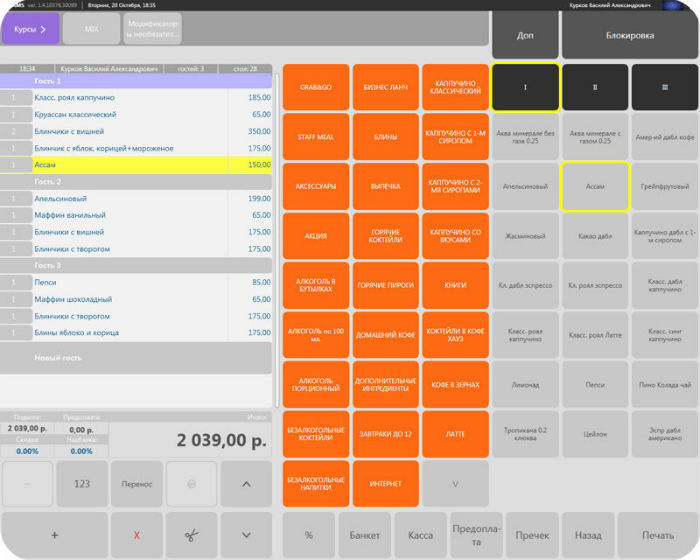


Рисунок 1.1

1.2 Приложение «Quick Resto» (рисунок 1.2).

Облачная система автоматизации ресторанного бизнеса, которая позволяет управлять меню, рассадкой гостей, принимать оплату и настраивать принтеры для печати чеков.

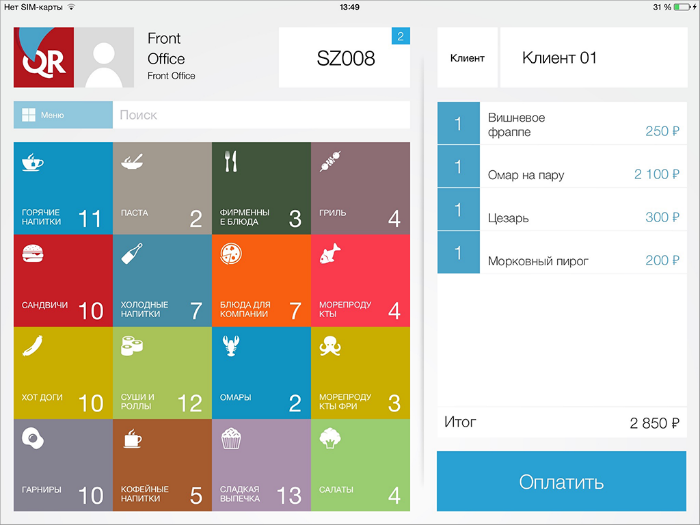


Рисунок 1.2

1.3 Приложение «GBS.Market» (рисунок 1.3).

Система для автоматизации магазинов и кафе — довольно простая программа для небольших заведений, обладающая соответствующим скромным набором функций. Отчетность с терминалов можно отправлять на почту.

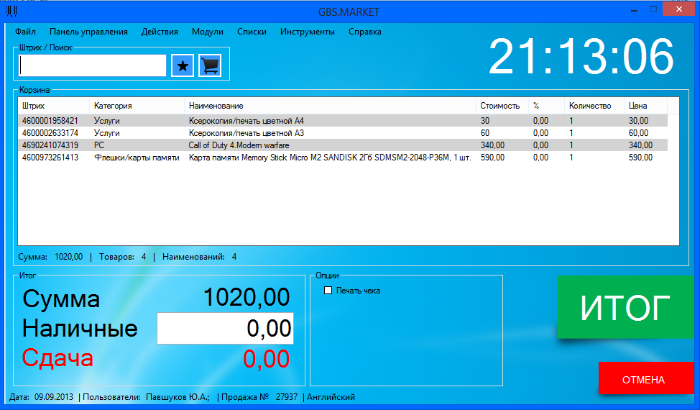


Рисунок 1.3

# **Проектирование и разработка систем**

# **Архитектура системы**

Для программного средства в курсовом проекте используется иерархическая система пользователей со строго ограниченным функционалом:

* Официант имеет ограниченный функционал (составление заказа и просмотр личной статистики) и занимает самую низшую ступень иерархии пользователей.
* Менеджер имеет полный контроль над программой, ему доступны функции добавления, удаления, редактирования. Занимает высшую ступень иерархии пользователей.

# **Проектирование логической структуры базы данных**

Для реализации поставленной задачи была создана база данных Restaurant. Для её создания использовалась система управления реляционными базами данных Microsoft SQL Server.

База данных нашей автоматизированной системы состоит из 3 таблиц, схематично базы данных изображена на рисунке 2.1.

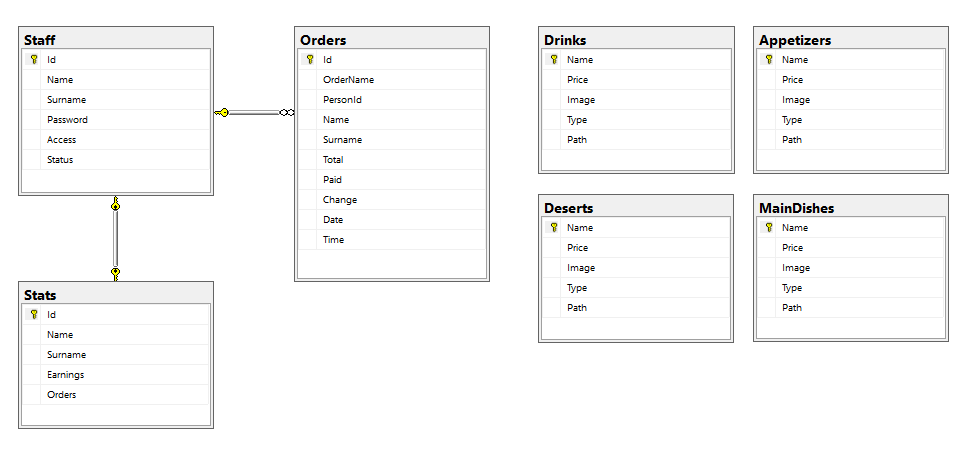


Рисунок 2.1. – Диаграмма базы данных.

На листингах Приложения А показаны классы, созданные для работы в приложении с данными, хранящимися в базе данных, а также скрипты создания таблиц базы данных.

# **Разработка приложения**

# **Разграничение прав доступа**

В программной подсистеме существует несколько разновидностей ролей пользователей. На диаграмме вариантов использования (рисунок 2.2.) представлены следующие роли пользователей:

* менеджер;
* официант;

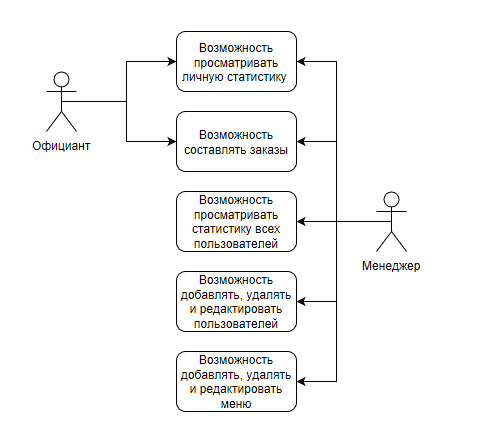


Рисунок 2.2.

Так же эта диаграмма отображает различные взаимодействия между ролями и элементами приложения. Система не позволяет без аутентификации пользоваться приложением.

Аутентификация – проверка идентификации пользователя, устройства или другого компонента в системе (обычно для принятия решения о разрешении доступа к ресурсам системы).

Авторизация – предоставление субъектам доступа к объектам системы.

Администратору предоставляется доступ к панели админа в которой он может добавлять и удалять пользователей и автомобили. Также администратору предоставляется полный доступ к приложению, возможность добавлять/редактировать/удалять информацию.

Официанту предоставляется доступ ко вкладкам:

* заказы;
* статистика (доступ только к личной статистике)

Менеджеру предоставляется доступ ко вкладкам:

* заказы;
* статистика (просмотр статистики всех пользователей);
* редактор меню (добавление, удаление, редактирование);
* персонал (добавление, удаление, редактирование)

# **Структура проекта**

Структура проекта изображена на рисунке 2.3.

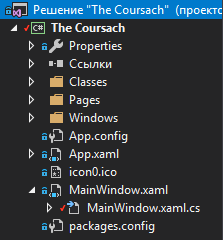


Рисунок 2.3. – Структура проекта

***Папка Classes***

Папка Classes содержит классы, используемые приложением.

***Папка Pages***

Папка Pages содержит страницы, используемые приложением.

***Папка Windows***

Папка Windows содержит вспомогательные окна приложения.

***App.Config***

Файл хранения данных конфигурации.

***App.xaml***

Файл хранения ресурсов.

***MainWindow.xaml***

Файл Главного окна приложения.

***Icon0.ico***

Файл иконки приложения.

# **Классы**

Структура проекта представлена на рисунке 2.4.

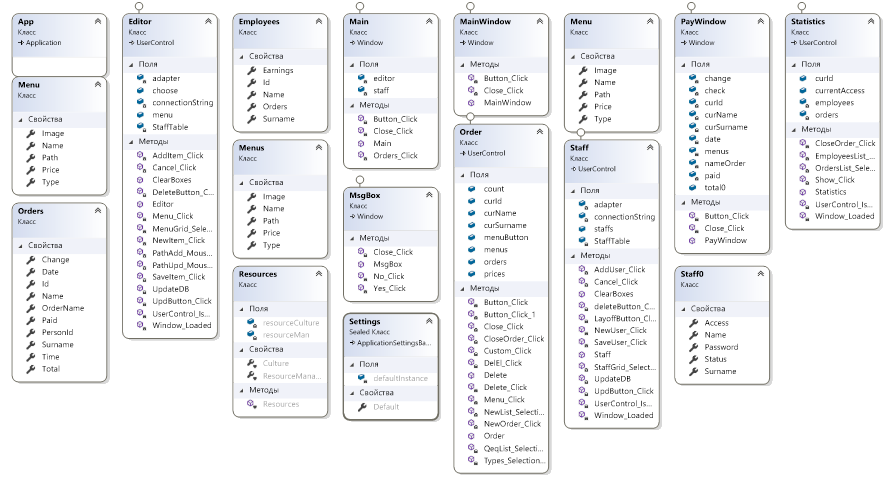


Рисунок 2.4. – Диаграмма классов

На листингах Приложения B показана реализация ключевых функций, которые выполняет программное средство: управление базой данных, обработка вводимых пользователями данных и исключительных ситуаций.

# **Руководство пользователя**

# **Менеджер**

При запуске приложения пользователь попадает на окно авторизации. Окно авторизации включает в себя форму, где необходимо указать свой логин и пароль (рисунок 3.1.1)

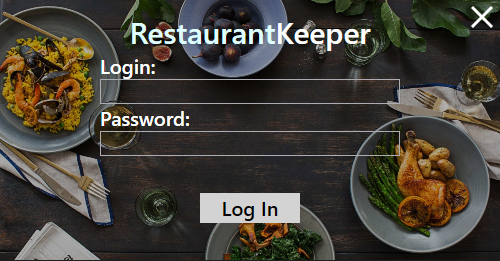


Рисунок 3.1.1. – Окно авторизации.

После входа пользователь видит краткое руководство пользователя и меню.

Также в главном окне отображается роль, имя и фамилия пользователя (рисунок 3.1.2).



Рисунок 3.1.2. – Стартовая страница.

Далее пользователь с ролью «Менеджер» может перейти во вкладку «Заказы» для начала работы и формирования заказов (рисунок 3.1.3). Здесь он может добавлять в заказ доступные элементы меню, удалять их, переключаться между активными заказами, а затем перейти к оплате заказа (рисунок 3.1.4).

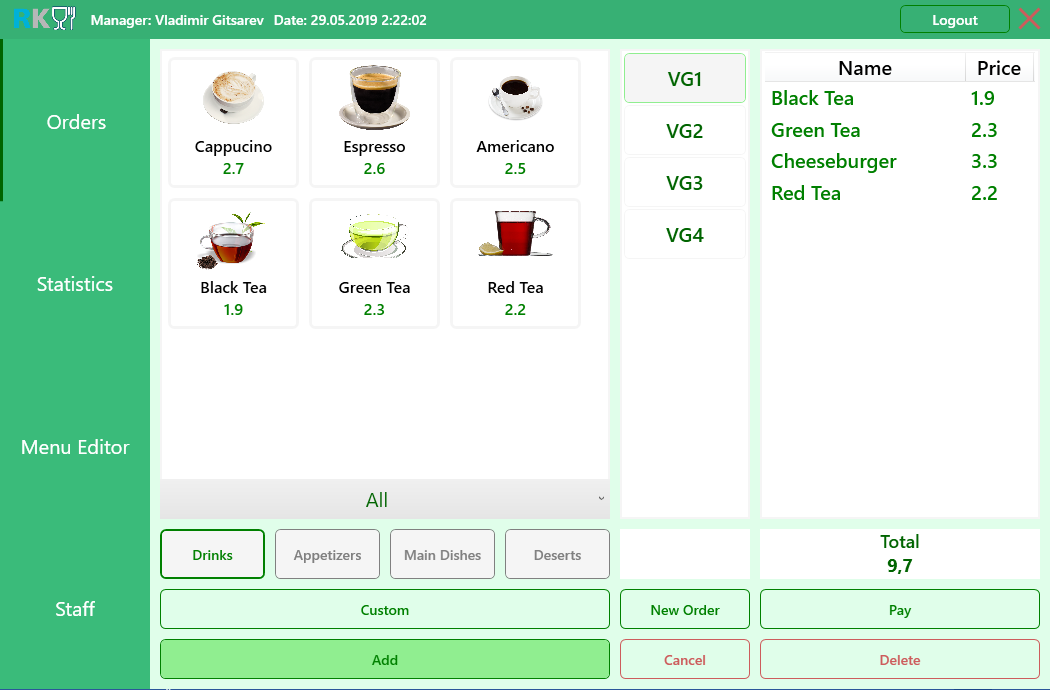


Рисунок 3.1.3. – Составление заказов.



Рисунок 3.1.4. – Оплата заказа

Пользователю «Менеджер» доступна вкладка «Статистика», в том числе статистика всех зарегистрированных пользователей, он может просматривать статистику по пользователям, а также отфильтровать результаты сбора статистики по дате (рисунок 3.1.5).

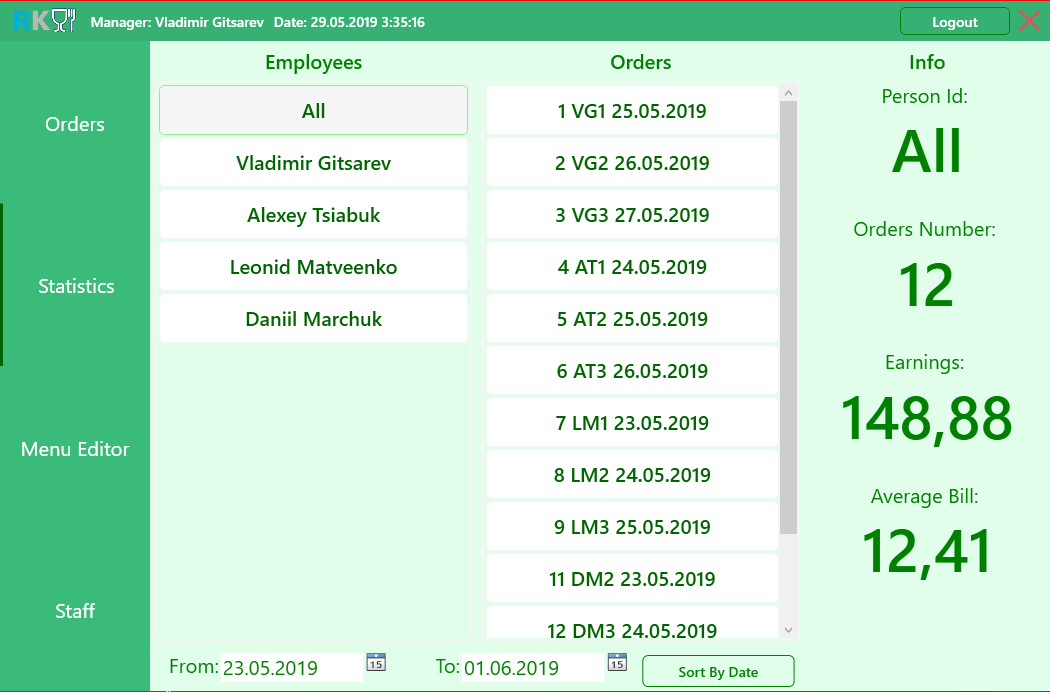


Рисунок 3.1.5 – Просмотр статистики

Вкладка «Редактор меню» доступна только пользователю «Менеджер».

Она позволяет редактировать меню: добавлять, удалять и изменять элементы меню.

Для добавления следует выбрать необходимый пункт и нажать на соответствующую кнопку. Далее появится форма, где следует внести информацию о добавляемом элементе. Для выбора изображения необходимо произвести двойной щелчок мыши по полю «Путь», и в открывшемся диалоговом окне выбрать нужное изображение. Аналогичным образом происходит редактирование элементов меню, предварительно выбирая элемент из списка. Для удаления следует выделить удаляемый элемент и нажать соответствующую кнопку (рисунок 3.1.6).

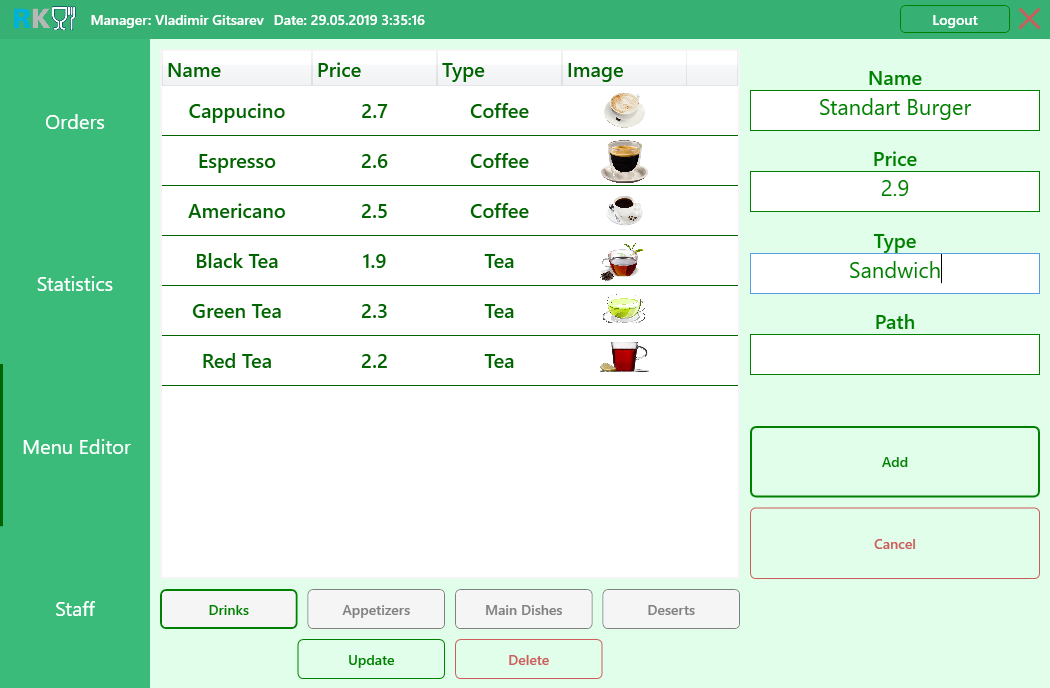


Рисунок 3.1.6 – Редактор меню

Вкладка «Персонал» позволяет управлять данными о пользователях: добавлять, удалять, изменять. Для добавления следует выбрать необходимый пункт и нажать на соответствующую кнопку. Далее появится форма, где следует внести информацию о добавляемом сотруднике. Аналогичным образом происходит редактирование информации о сотрудниках, предварительно выбирая нужного сотрудника из списка. Для удаления следует выделить удаляемый элемент и нажать соответствующую кнопку, но в данном случае произойдет удаление всей имеющейся информации о данном пользователе. Поэтому, если нужно сохранить данные о пользователе, но удалить его из списка сотрудников, следует использовать кнопку «Уволить», в таком случае статус сотрудника изменится, и он не сможет больше авторизоваться в приложении, но информация о нем будет сохранена (рисунок 3.1.7)

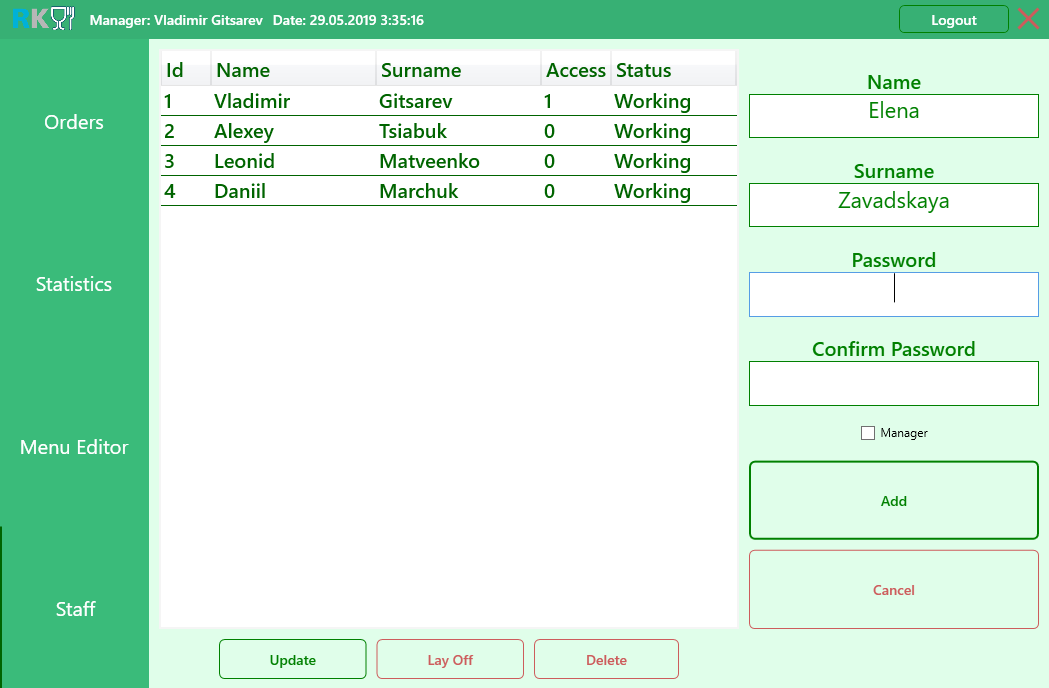


Рисунок 3.1.7 – Управление персоналом

# **Официант**

Официант будет иметь доступ только к двум вкладкам: «Заказы» и «Статистика». Во вкладке «Заказы» пользователь официант будет иметь такой же функционал, что и пользователь «Менеджер». Однако во вкладке «Статистика» ему будет доступна только личная статистика, в отличие от пользователя «Менеджер», которому доступна статистика всех пользователей (рисунок 3.2.1).

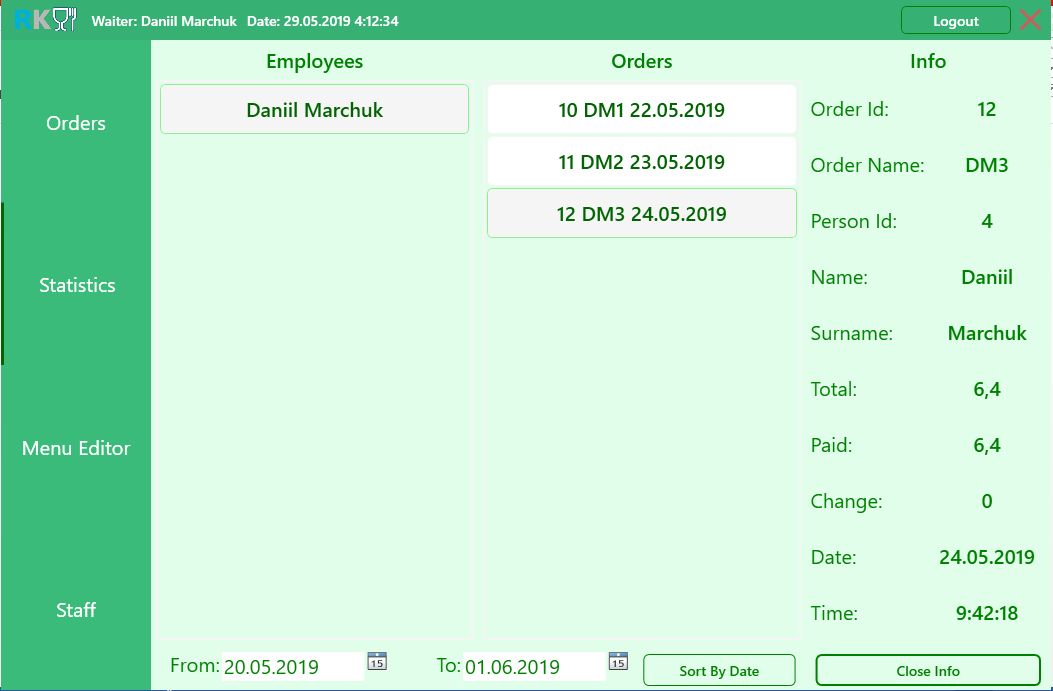


Рисунок 3.2.1 – Личная статистика

# **Тестирование программы**

Для корректной работы программы необходимо обеспечить защиту работы пользователя от ошибок и сбоев. Для этого используются конструкции типа «try-catch» и «if-else». «Try-Catch» служит для «отлавливания» исключений с посредствующей их обработкой, а «If-Else» для проверки того или иного. Это необходимо, чтобы при вводе некорректной информации, удаление несуществующих данных, изменение данных на непредполагаемые типы данных, программа не получала сбоев.

Предусмотрена обработка следующих исключительных ситуаций:

* ввод неверного логина или пароля (рисунок 4.1):

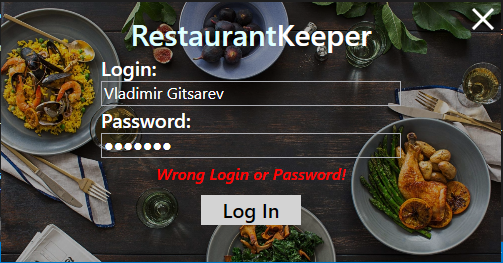


Рисунок 4.1. – Ошибка при вводе неверного логина или пароля.

* при неправильном вводе данных в окне «добавления пользователя» или «добавление элемента меню» (рисунок 4.2 и 4.3):

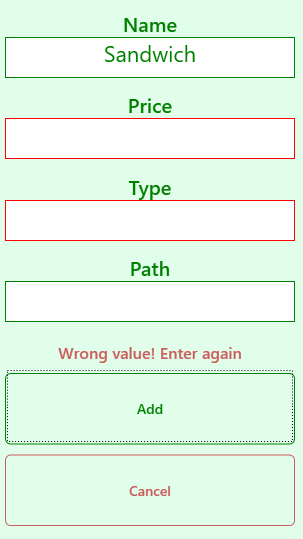


Рисунок 4.2 – Добавление элемента меню

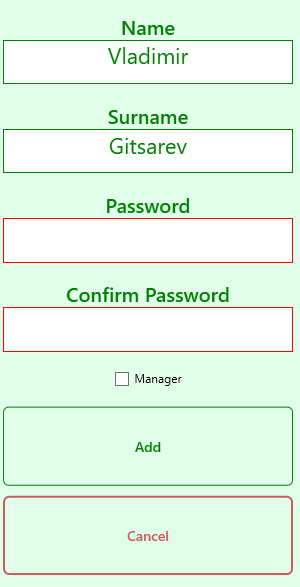


Рисунок 4.3. – Добавление пользователя.

* При вводе неверной суммы в окне оплаты (рисунок 4.4)

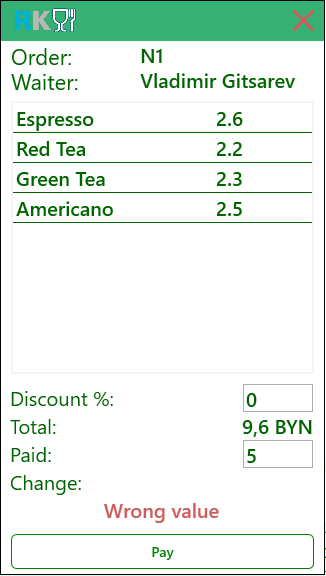


Рисунок 4.4 – Оплата заказа

# **Заключение**

Данный проект был разработан для ресторана с максимально простой системой управления. Главной задачей было сделать простое и удобное приложение. Для этого был использован API-интерфейс WPF и база данных, основанная на принципах ADO.NET. Чтобы сделать программу легко изменяемой и доступной, были использованы максимально удобные функции наведения пользователя. Также был проработан ряд исключительных ситуаций с выводом сообщений, в случае некорректной работы пользователя.

Разработка данного приложения – мой первый опыт создания достаточно масштабного проекта, который включает в себя работу с базой данных, а также создание функционала, который позволяет назвать данное приложение конкурентоспособным на рынке программного обеспечения.

Опыт, полученный в ходе создания данного курсового проекта, пригодятся мне и в дальнейшей разработке проектов, созданных на WPF с использование ЯП C# и работой с .NET Framework и базой данных SQL Server.

# **Список литературы**

1. Интернет-портал [Электронный ресурс] / [https://metanit.com](https://vk.com/away.php?to=https%3A%2F%2Fmetanit.com&cc_key=) – WPF и С#. Полное руководство: [https://metanit.com/sharp/wpf/](https://vk.com/away.php?to=https%3A%2F%2Fmetanit.com%2Fsharp%2Fwpf%2F&cc_key=) – Дата доступа: 10.05.2018   
2. Интернет-портал [Электронный ресурс] / [https://docs.microsoft.com](https://vk.com/away.php?to=https%3A%2F%2Fdocs.microsoft.com&cc_key=) – SQL Server Documentation: [https://docs.microsoft.com/en-us/sql/sql-server/sql-s..](https://vk.com/away.php?to=https%3A%2F%2Fdocs.microsoft.com%2Fen-us%2Fsql%2Fsql-server%2Fsql-server-technical-documentation%3Fview%3Dsql-server-2017&cc_key=) – Дата доступа: 11.05.2018   
3. Пацей, Н.В. Курс лекций по языку программирования C# / Н.В. Пацей. – Минск: БГТУ, 2016. – 175 с.

# **Приложение А**

namespace The\_Coursach.Classes

{

public class Employees

{

public int Id { get; set; }

public string Name { get; set; }

public string Surname { get; set; }

public double Earnings { get; set; }

public int Orders { get; set; }

}

}

Создание класса Employees.

namespace The\_Coursach.Classes

{

public class Menu

{

public string Path { get; set; }

public string Type { get; set; }

public double Price { get; set; }

public string Name { get; set; }

public byte[] Image { get; set; }

}

}

Создание класса Menu.

namespace The\_Coursach.Classes

{

public class Orders

{

public int Id { get; set; }

public string OrderName { get; set; }

public int PersonId { get; set; }

public string Name { get; set; }

public string Surname { get; set; }

public double Total { get; set; }

public double Paid { get; set; }

public double Change { get; set; }

public string Date { get; set; }

public string Time { get; set; }

}

}

Создание класса Orders.

namespace The\_Coursach.Classes

{

public class Staff0

{

public string Name { get; set; }

public string Surname { get; set; }

public string Password { get; set; }

public int Access { get; set; }

public string Status { get; set; }

}

}

Создание класса Staff.

create table Deserts

(

Name nvarchar(50) primary key,

Price float,

Image varbinary(Max),

Type nvarchar(50),

Path nvarchar(200)

)

create table Drinks

(

Name nvarchar(50) primary key,

Price float,

Image varbinary(Max),

Type nvarchar(50),

Path nvarchar(200)

)

create table Appetizers

(

Name nvarchar(50) primary key,

Price float,

Image varbinary(Max),

Type nvarchar(50),

Path nvarchar(200)

)

create table MainDishes

(

Name nvarchar(50) primary key,

Price float,

Image varbinary(Max),

Type nvarchar(50),

Path nvarchar(200)

)

Создание таблиц с элементами меню.

create table Staff

(

Id int primary key identity(1,1) not null,

Name nvarchar(50),

Surname nvarchar(50),

Password nvarchar(50),

Access int not null,

Status nvarchar(50)

)

create table Stats

(

Id int primary key not null identity(1,1) foreign key references Staff (Id),

Name nvarchar(50),

Surname nvarchar(50),

Earnings float default 0,

Orders int default 0

)

create table Orders

(

Id int primary key identity(1,1) not null,

OrderName nvarchar(50),

PersonId int not null foreign key references Staff (Id),

Name nvarchar(50),

Surname nvarchar(50),

Total float default 0,

Paid float,

Change float,

Date date,

Time nvarchar(50)

)

Создание таблиц Персонал, Заказы, Статистика.

# **Приложение B**

Чтение данных о пользователях из БД и авторизация пользователей:

bool check = false;

SqlConnection sqlConnection = new SqlConnection(ConfigurationManager.ConnectionStrings["DefaultConnection"].ConnectionString);

sqlConnection.Open();

SqlCommand sqlCommand = sqlConnection.CreateCommand();

sqlCommand.CommandText = "Select Name, Surname, Password, Access, Id, Status from Staff order by Id desc";

SqlDataReader thisReader = sqlCommand.ExecuteReader();

while (thisReader.Read())

{

if ((thisReader.GetValue(0).ToString() + " " + thisReader.GetValue(1).ToString()) == lgn.Text && thisReader.GetValue(2).ToString() == psw.Password.GetHashCode().ToString() && thisReader.GetValue(5).ToString() != "Fired")

{

check = true;

int access = (int)thisReader.GetValue(3);

Main main = new Main((int)thisReader.GetValue(4), thisReader.GetValue(0).ToString(), thisReader.GetValue(1).ToString(), access);

main.Show();

this.Close();

}

}

Чтение данных из БД, создание объектов классов, заполнение элементов управления прочитанными данными:

string sql = "SELECT Id, Name, Surname, Access, Status FROM Staff order by Access desc";

StaffTable = new DataTable();

SqlConnection connection = null;

try

{

connection = new SqlConnection(connectionString);

SqlCommand command = new SqlCommand(sql, connection);

adapter = new SqlDataAdapter(command);

adapter.InsertCommand = new SqlCommand("sp\_InsertStaff", connection);

adapter.InsertCommand.CommandType = CommandType.StoredProcedure;

adapter.InsertCommand.Parameters.Add(new SqlParameter("@id", SqlDbType.Int, 0, "Id"));

adapter.InsertCommand.Parameters.Add(new SqlParameter("@name", SqlDbType.NVarChar, 50, "Name"));

adapter.InsertCommand.Parameters.Add(new SqlParameter("@surname", SqlDbType.NVarChar, 50, "Surname"));

adapter.InsertCommand.Parameters.Add(new SqlParameter("@access", SqlDbType.Int, 0, "Access"));

adapter.InsertCommand.Parameters.Add(new SqlParameter("@status", SqlDbType.NVarChar, 50, "Status"));

connection.Open();

adapter.Fill(StaffTable);

StaffGrid.ItemsSource = StaffTable.DefaultView;

SqlCommand command0 = new SqlCommand("Select \* from Staff order by Access desc", connection);

SqlDataReader reader = command0.ExecuteReader();

while (reader.Read())

staffs.Add(new Staff0 { Name = reader.GetValue(1).ToString(), Surname = reader.GetValue(2).ToString(), Password = reader.GetValue(3).ToString(), Access = (int)reader.GetValue(4), Status = reader.GetValue(5).ToString() });

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

if (connection != null)

connection.Close();

}

Добавление записей в БД и обработка исключительных ситуаций:

try

{

if (correct == true)

{

SqlCommand command = new SqlCommand();

command.Connection = connection;

command.CommandText = @"INSERT INTO " + choose + " VALUES (@Name, @Price, @Image, @Type, @Path)";

command.Parameters.Add("@Name", SqlDbType.NVarChar, 50);

command.Parameters.Add("@Price", SqlDbType.Float, 50);

command.Parameters.Add("@Type", SqlDbType.NVarChar, 50);

command.Parameters.Add("@Image", SqlDbType.Image, 1000000);

command.Parameters.Add("@Path", SqlDbType.NVarChar, 100);

string filename = null;

if (pathAdd.Text == "")

filename = @"D:\All\Универ\2 курс\ООП\The Coursach\Images\image.png";

else

filename = pathAdd.Text;

string name = nameAdd.Text;

string price = priceAdd.Text.Replace(".", ",");

string type = typeAdd.Text;

byte[] imageData;

using (System.IO.FileStream fs = new System.IO.FileStream(filename, FileMode.Open))

{

imageData = new byte[fs.Length];

fs.Read(imageData, 0, imageData.Length);

}

command.Parameters["@Name"].Value = name;

command.Parameters["@Price"].Value = price;

command.Parameters["@Type"].Value = type;

command.Parameters["@Image"].Value = imageData;

command.Parameters["@Path"].Value = filename;

command.ExecuteNonQuery();

ClearBoxes();

UpdateDB();

Window\_Loaded();

}

else

errorAdd.Visibility = Visibility.Visible;

}

catch

{

if (priceAdd.Text.Length > 0 && choose != null)

priceAdd.BorderBrush = Brushes.Red;

else

errorAdd.Text = "Chose table to add!";

errorAdd.Visibility = Visibility.Visible;

}

Проверка вводимых пользователем данных и изменение записей базы данных:

SqlConnection sqlConnection = new SqlConnection(ConfigurationManager.ConnectionStrings["DefaultConnection"].ConnectionString);

bool correct = false;

if (nameUpd.Text.Length == 0)

nameUpd.BorderBrush = Brushes.Red;

if (surnameUpd.Text.Length == 0)

surnameUpd.BorderBrush = Brushes.Red;

if (passwordUpd.Password.Length == 0)

{

passwordUpd.BorderBrush = Brushes.Red;

passwordConfirmUpd.BorderBrush = Brushes.Red;

}

if (passwordUpd.Password != passwordConfirmUpd.Password)

passwordConfirmUpd.BorderBrush = Brushes.Red;

if (nameUpd.Text.Length != 0 && surnameUpd.Text.Length != 0

&& passwordUpd.Password.Length != 0 && passwordUpd.Password == passwordConfirmUpd.Password)

correct = true;

if (correct == true)

{

nameUpd.BorderBrush = Brushes.Green;

surnameUpd.BorderBrush = Brushes.Green;

passwordUpd.BorderBrush = Brushes.Green;

passwordConfirmUpd.BorderBrush = Brushes.Green;

int manager = 0;

if (IsManagerUpd.IsChecked == true)

manager = 1;

String str = "Update Staff Set Name = '" + nameUpd.Text + "', Surname = '" + surnameUpd.Text + "', Password = '" + passwordUpd.Password.GetHashCode() + "', Access = " + manager + " Where Surname = '" + staffs[StaffGrid.SelectedIndex].Surname + "'";

String str0 = "Update Stats Set Name = '" + nameUpd.Text + "', Surname = '" + surnameUpd.Text + "' Where Surname = '" + staffs[StaffGrid.SelectedIndex].Surname + "'";

String str1 = "Update Orders Set Name = '" + nameUpd.Text + "', Surname = '" + surnameUpd.Text + "' Where Surname = '" + staffs[StaffGrid.SelectedIndex].Surname + "'";

sqlConnection.Open();

SqlCommand cmd = new SqlCommand(str, sqlConnection);

SqlCommand cmd0 = new SqlCommand(str0, sqlConnection);

SqlCommand cmd1 = new SqlCommand(str1, sqlConnection);

cmd.ExecuteNonQuery();

cmd0.ExecuteNonQuery();

cmd1.ExecuteNonQuery();

}