МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных Технологий

Кафедра Информационные системы и технологии

Специальность 1-98 01 03 «Программное обеспечение информационной

безопасности мобильных систем»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**КУРСОВОГО ПРОЕКТА:**

По дисциплине «Современные технологии программирования мобильных систем»

Тема Разработка программного средства «Food Shop»

Исполнитель

студент 2 курса группы 8 Першай Я. Б.

(Ф.И.О.)

Руководитель

ассистент Северинчик Н. А.

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

Курсовой проект защищен с оценкой

Руководитель Северинчик Н. А.

(подпись)

Минск 2019

**Содержание**

[**ВВЕДЕНИЕ** 3](#_Toc9847589)

[**1. Постановка задачи** 4](#_Toc9847590)

[**1.1. Основные задачи** 4](#_Toc9847591)

[**1.2. Обзор прототипов** 4](#_Toc9847592)

[**1.3. Информационные системы и технологии** 6](#_Toc9847593)

[**2. Разработка архитектуры проекта** 8](#_Toc9847594)

[**2.1. Обобщённая структура** 8](#_Toc9847595)

[**2.2. Диаграммы UML, взаимосвязь всех компонентов** 9](#_Toc9847596)

[**3. Разработка функциональной модели и модели данных** 10](#_Toc9847597)

[**3.1. Модель базы данных** 10](#_Toc9847598)

[**4. Руководство пользователя** 13](#_Toc9847599)

[**4.1. Руководство для входа и регистрации** 13](#_Toc9847600)

[**4.2. Руководство для пользования программой Администратором** 15](#_Toc9847601)

[**4.3. Руководство для пользования программой Пользователем** 18](#_Toc9847602)

[**5. Тестирование** 20](#_Toc9847603)

[**Заключение** 24](#_Toc9847604)

[**Список используемой литературы** 25](#_Toc9847605)

[**Приложение А** 26](#_Toc9847606)

[**Приложение Б** 27](#_Toc9847607)

[**Приложение В** 28](#_Toc9847608)

# **ВВЕДЕНИЕ**

Темой моего курсового проекта является разработка приложения для заказа еды и корпоративных обедов.

В наши дни мир живёт на больших скоростях. Время летит быстро, и не всегда успеваешь хорошо питаться. Более того, человек не всегда помнит, что нужно поужинать или пообедать. А ведь нормальный приём пищи просто необходим человеческому организму для нормального функционирования.

Но что делать, если нужно работать, а не всегда можно выбраться где-то нормально пообедать?! Для этого нужно прервать работу, одеться, выйти из офиса, найти подходящее заведение, изучить репертуар и т.д. Много времени и нервов уйдет на это, не правда ли?

Гораздо проще, когда есть доступ к приложению, в котором, не вставая с рабочего кресла можно заказать себе порцию супа с пюре и котлетой, который вам доставят в нужное время в нужное место. Сразу на месте можно посмотреть меню, узнать цену, заказать необходимое количество. Гораздо удобнее, чем ходить в кафе или столовую.

То же можно сказать и о корпоративных обедах. Особенно в ИТ-компаниях. Сотрудники должны быть продуктивны, много работать, и работать хорошо. И гораздо удобнее, когда можно встать, пойти на кухню в офисе, взять порцию еды и хорошо пообедать. Экономится время и силы. Эту проблему также может решить моё приложение: пользователь может заказать большое количество порций еды, которой хватит на всех сотрудников и которую подвезут прямо под офис.

Таким образом, наличие хорошего приложения для заказа еды является важным аспектом для людей, ведущих активный деловой образ жизни, особенно в больших городах, где время летит незаметно. Своевременный приём пищи сохранит здоровье и нервную систему на более долгий промежуток времени. Данные проблемы и помогает решить приложение, которое я разработал в своём курсовом проекте.

# **Постановка задачи**

# **Основные задачи**

Основной задачей программного средства «Food Shop» является предоставление функционала, позволяющего пользователю осуществить заказ еды, а администратору управлять складом, историей заказов и также осуществлять заказ.

Программное средство выполняет следующие задачи для Администратора:

* Просмотр склада товаров
* Пополнение склада товаров
* Просмотр истории заказов
* Осуществление заказов

Программное средство выполняет следующие задачи для Пользователя:

* Регистрация
* Просмотр каталога (меню)
* Осуществление заказа

# **Обзор прототипов**

В наше время современные технологии являются неотъемлемой частью жизни современного человека. Поэтому с каждым днем появляется всё больше программных средств, упрощающих и улучшающих жизнь людей. В том числе и в плане доставки еды.

Одним из самых популярных приложений данной тематики является Menu.by.

Данное приложение предназначено для заказа еды в больших городах, предпочтительно в областных центрах. Оно имеет как мобильную версию, так и версию веб-приложения т.е. заказ можно осуществлять на сайте. Функционал Меню.бай (рисунок 1.1) включает в себя проведение таких операций, как: регистрация пользователя (можно входить в приложение с помощью учётных записей VK, Google, Одноклассники, Твиттер, Facebook) (рисунок 1.2), определение местоположения при входе, выбор ресторанов, просмотр акций, получение уведомлений об акциях и скидках.



Рисунок 1.1 – Логотип приложения Меню.бай

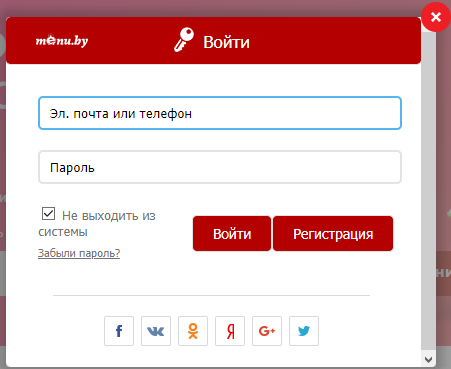


Рисунок 1.2 – Форма входа

На данной картинке мы видим, что приложение предоставляет вход с помощью сторонних учётных записей. Так же, в случае, если пароль был забыт, приложение предоставляет удобную функцию «забыли пароль». Главная страница (рисунок 1.3) предоставляет список популярных заведений для доставки их еды.

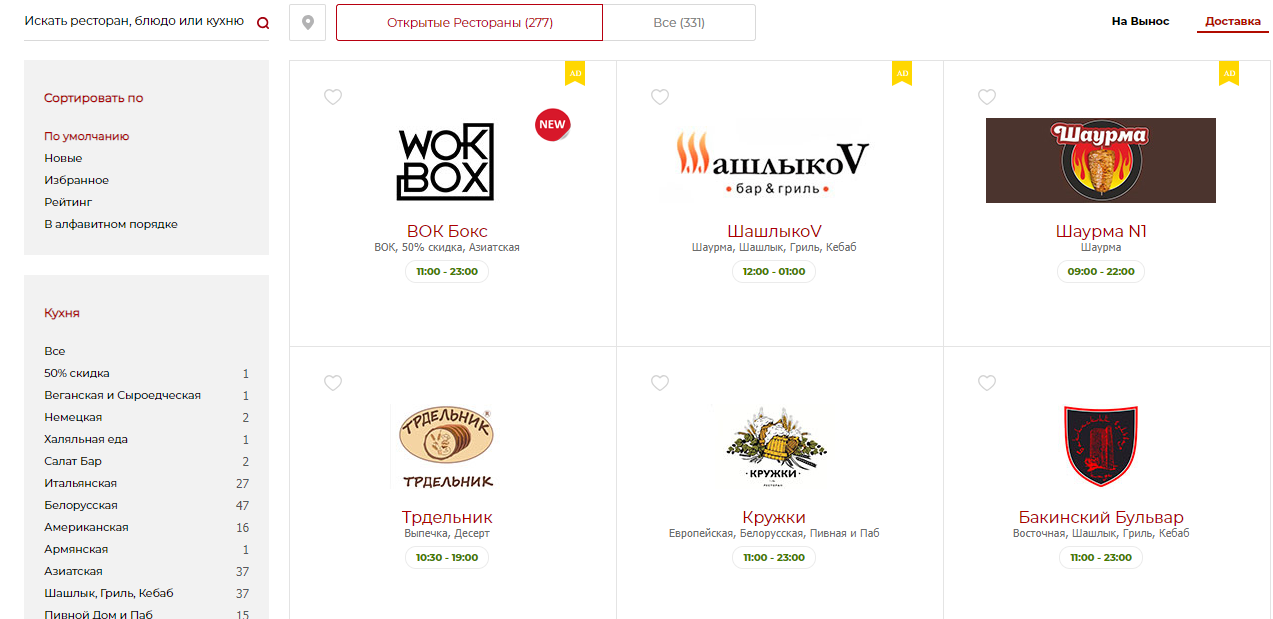


Рисунок 1.3 – главная страница приложения

Выбор и поиск ресторана, определенной кухни (рисунок 1.4), а так же сортировка для удобного восприятия.

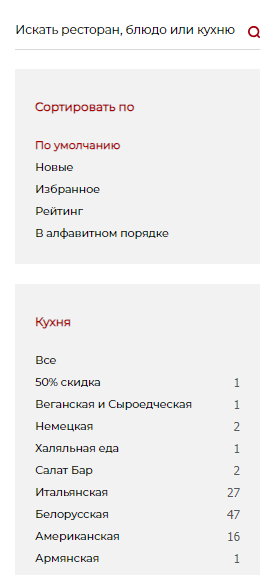


Рисунок 1.4 – поиск и сортировка

На мой взгляд, данное приложение отлично выполняет свои функции и даже более того. Предоставляет возможность осуществления доставки для заведений, которые не включают в себя самостоятельную функцию доставки т.е. происходит интеграция с заведениями по приёму пищи. Также, для экономных очень удобно предоставляется раздел «Скидки и акции». Основная суть данного приложения – интеграция с пищевыми заведениями города.

# **Информационные системы и технологии**

При разработке приложения были использованы:

* интегрированная среда разработки Microsoft Visual Studio Professional 2017;
* Microsoft SQL Server
* язык программирования C#;
* язык запросов к БД SQL
* технология WPF;
* ADO.NET;

На сегодняшний момент язык программирования C# один из самых мощных, быстро развивающихся и востребованных языков в ИТ-отрасли. В настоящий момент на нем пишутся самые различные приложения: от небольших десктопных программок до крупных веб-порталов и веб-сервисов, обслуживающих ежедневно миллионы пользователей. Когда говорят C#, нередко имеют в виду технологии платформы .NET (WPF, ASP.NET). И, наоборот, когда говорят .NET, нередко имеют в виду C#. Язык C# был создан специально для работы с фреймворком .NET.

Windows Presentation Foundation — система для построения клиентских приложений Windows с визуально привлекательными возможностями взаимодействия с пользователем, графическая (презентационная) подсистема в составе .NET Framework (начиная с версии 3.0), использующая язык XAML.

SQL Server является одной из наиболее популярных систем управления базами данных (СУБД) в мире. Данная СУБД подходит для самых различных проектов: от небольших приложений до больших высоконагруженных проектов.

SQL Server характеризуется такими особенностями как:

* Производительность. SQL Server работает очень быстро.
* Надежность и безопасность. SQL Server предоставляет шифрование данных.
* Простота. С данной СУБД относительно легко работать и вести администрирование.

SQL (Structured Query Language) – язык структурированных запросов, с помощью него пишутся специальные запросы (так называемые SQL инструкции) к базе данных с целью получения данных из базы данных или для манипулирования этими данными. Также обязательно стоит отметить и то, что база данных, и в частности реляционная модель, основана на теории множеств, которая подразумевает объединение разных объектов в одно целое, под одним целым в базе данных как раз и имеется в виду таблица. Это важно, так как язык SQL работает именно со множеством, с набором данных, т.е. с таблицами.

ADO.NET предоставляет собой технологию работы с данными, которая основана на платформе .NET Framework. Эта технология представляет нам набор классов, через которые мы можем отправлять запросы к базам данных, устанавливать подключения, получать ответ от базы данных и производить ряд других операций.

# **Разработка архитектуры проекта**

# **Обобщённая структура**

Решение FoodShop представлено проектом FoodShop (рисунок 2.1). Он включает в себя классы, предназначенные для роли Администратора и для роли Пользователя.

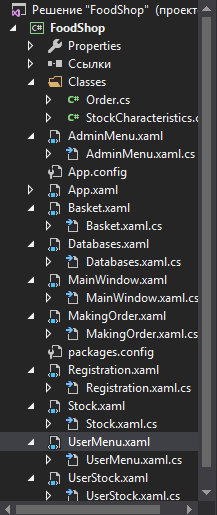


Рисунок 2.1 – Структура проекта

Таблица 2.1 – Описание структуры проекта

|  |  |
| --- | --- |
| Properties | Свойства проекта, содержит информацию о сборке, используемых ресурсах и настройках |
| Ссылки | Перечень сборок, используемых в проекте |
| Classes | Папка со свойствами, предназначенных для хранения характеристик элементов |
| AdminMenu | Окно меню для роли Администратора |
| App.config | Окно для центра трансфузиологии |
| Продолжение таблицы 2.1. | |
|  |  |
| App.xaml | Класс Application |
| Basket | Окно для конечных данных заказа и завершения работы | |
| Databases | Окно для просмотра истории заказов |
| MainWindow | Окно отображения Главной страницы |
| MakingOrder | Окно оформления заказа |
| Package.config | Файл со списком пакетов, на которые ссылается проект |
| Registration | Окно регистрации пользователя |
| Stock | Окно отображения склада и добавления товаров |
| UserMenu | Окно меню для роли Пользователя |
| UserStock | Окно списка товаров в наличии для роли Пользователя |

# **Диаграммы UML, взаимосвязь всех компонентов**

UML-диаграмма – это специализированный язык графического описания, предназначенный для объектного моделирования в сфере разработки различного программного обеспечения. Данный язык имеет широкий профиль и представляет собой открытый стандарт, в котором используются различные графические обозначения, чтобы создать абстрактную модель системы. UML создавался для того, чтобы обеспечить определение, визуализацию, документирование, а также проектирование всевозможных программных систем.

Диаграммы вариантов использования описывают взаимоотношения и зависимости между группами вариантов использования и действующих лиц, участвующими в процессе. Каждый вариант использования определяет последовательность действий, которые должны быть выполнены проектируемой системой при взаимодействии ее с соответствующим актером.

Таким образом, использование моего приложения подразумевает участие двух актеров: Администратора и Пользователя. В зависимости от статуса, каждый актер имеет свой набор функционала. Очевидно, что у Администратора функционал будет шире.

На UML диаграмме классов я должен отобразить план действий для каждой роли.

На рисунке 2.2 представлена диаграмма последовательности, которая отражает поток событий, происходящих в рамках варианта использования.

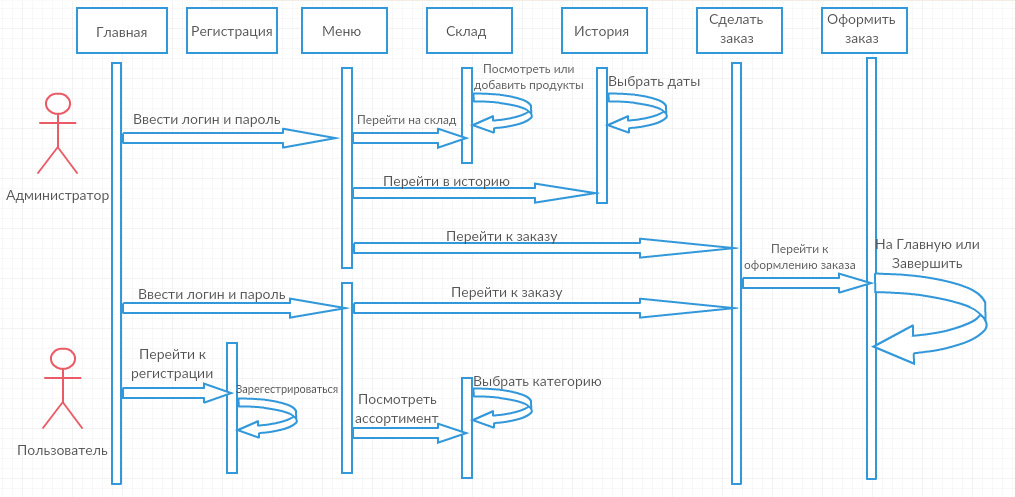


Рисунок 2.2 – Диаграмма последовательности Приложения для заказа еды

# **Разработка функциональной модели и модели данных**

# **Модель базы данных**

Для хранения данных о заказах и пользователях была создана База Данных (далее БД) FoodShop. Для её создания использовалась СУБД SQL Server 2012.

Текущая БД состоит из 4 таблиц (рисунок 3.1). Скрипты для работы с базой данных представлены в приложении А.

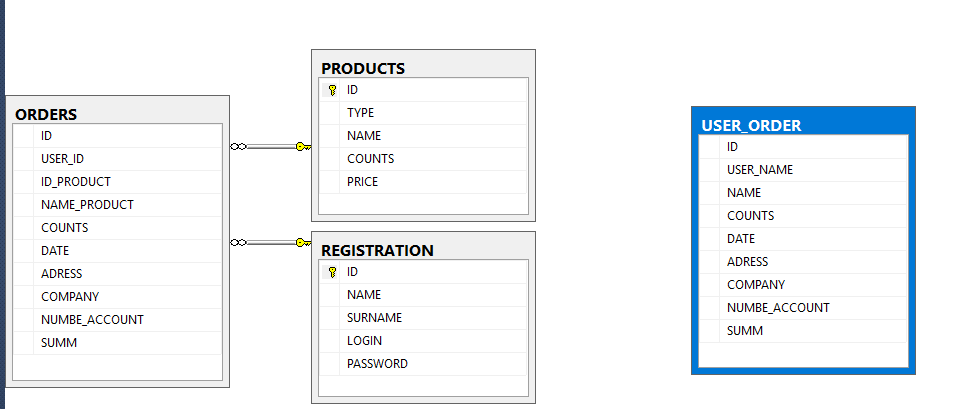


Рисунок 3.1 – БД FoodShop

На рисунке 3.2 представлена структура таблицы REGISTRATION, которая содержит информацию о зарегистрированном пользователе, которая также потом используется при входе.

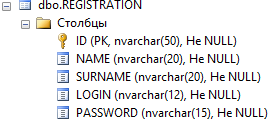


Рисунок 3.2 – Структура таблицы REGISTRATION

Таблица содержит поля:

* ID – уникальный идентификатор пользователя (создаётся с помощью класса Guid в C#, который генерирует уникальную комбинацию из цифр и букв) ;
* NAME – имя пользователя;
* SURNAME – фамилия пользователя;
* LOGIN – логин, выбранный пользователем;
* PASSWORD – пароль, выбранный пользователем.

На рисунке 3.3 представлена структура таблицы PRODUCTS, которая содержит в себе информацию о продуктах и блюдах в наличии.

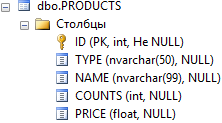


Рисунок 3.3 – Структура таблицы PRODUCTS

Таблица содержит поля:

* ID – идентификатор продукта;
* TYPE – тип блюдах (первое, второе, фастфуд, напиток);
* NAME – название блюда;
* COUNTS – количество на складе;
* PRICE – цена блюда.

На рисунке 3.4 представлена структура таблицы ORDERS, которая содержит информацию о заказах.

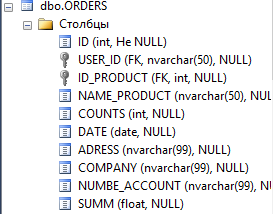


Рисунок 3.4 – Структура таблицы ORDERS

Таблица содержит поля:

* ID – идентификатор операции;
* USER\_ID – уникальный идентификатор пользователя, осуществившего заказ;
* ID\_PRODUCT – идентификатор продукта(блюда);
* NAME\_PRODUCT – название блюда;
* COUNTS – количество заказанных единиц блюда;
* DATE – дата доставки;
* ADRESS – адрес доставки;
* COMPANY – компания, осуществившая заказ;
* NUMBE\_ACCOUNT – номер счёта;
* SUMM – сумма заказа

На рисунке 3.5 представлена структура таблицы USER\_ORDER, которая является временным хранилищем данных о заказе для отображения информации пользователю.

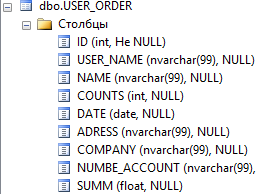


Рисунок 3.5 – Структура таблицы USER\_ORDER

Таблица содержит поля:

* ID – идентификатор операции;
* USER\_NAME – уникальный идентификатор пользователя,осуществившего заказ;
* NAME – идентификатор продукта(блюда);
* COUNTS – количество заказанных единиц блюда;
* DATE – дата доставки;
* ADRESS – адрес доставки;
* COMPANY – компания, осуществившая заказ;
* NUMBE\_ACCOUNT – номер счёта;
* SUMM – сумма заказа

# **Руководство пользователя**

# **Руководство для входа и регистрации**

Для запуска программы необходимо активировать .exe – файл с названием «FoodShop», после чего запустится программа.

Красный цвет приложения я выбрал в связи с тематикой программного продукта т.к. красный цвет вызывает у людей аппетит, что выгодно для приложения заказа еды.

Данная программа предназначена для пользователей двух типов: Администратора и Пользователя. Можно посмотреть информацию «О нас», нажав соответствующую кнопку в левом нижнем углу. Если нужно выйти из приложения, всегда есть кнопка со значком «выключение» в правом верхнем углу на каждой странице.

Вход в приложение осуществляется в окне MainWindow.xaml (рисунок 4.1).



Рисунок 4.1 – Главная страница

Вход без регистрации может осуществить Админ или уже зарегистрированный пользователь. Если у пользователя еще нет аккаунта, ему необходимо перейти в раздел «Регистрация» – окно Registration.xaml.



Рисунок 4.2 – Окно регистрации

На данном окне находятся поля для ввода имени, фамилии, логина, пароля и повтора пароля для проверки на правильность. Вводя логин и пароль на главной странице, пользователь авторизуется и получает доступ к основному функционалу приложения.

Для Администратора не предусмотрена регистрация во избежание авторизации пользователя, данные которого могут случайно совпасть с данными администратора. Также для дополнительной защиты, пользователям запрещено выбирать логин, содержащий не только цифры, что не позволит им авторизоваться под логином Админа т.к. его логин включает в себя символы алфавитов.

При регистрации пользователя идёт проверка на наличие повторяемого логина, которая защищает от регистрации двух пользователей с идентичными логинами. Код проверки на существование логина представлен в Приложении Б.

После регистрации пользователь перебрасывается на главную страницу.

# **Руководство для пользования программой Администратором**

При входе загружается главная страница Меню. Здесь представлен выбор для пользования основным функционалом: пункт меню «Склад», кнопка «История», кнопка «Сделать заказ». Также находятся кнопки для выхода из приложения, для выхода на Главную. Страница представлена на (рисунке 4.3).



Рисунок 4.3 – Меню

При нажатии на пункт «Склад» переходим на страницу Склада (рисунок 4.4). На данной странице можно посмотреть список блюд по каждой категории (Первое, Второе, Фастфуд, Напитки). Также можно добавить блюдо в базу данных т.е. осуществить «привоз». При добавлении продукта, который уже имеется, просто обновляется количество единиц на складе: суммируется старое значение и новое.

Чтобы добавить блюдо нужно выбрать тип блюда, ввести данные в соответствующие TextBox и нажать кнопку «Добавить». Чтобы посмотреть имеющиеся данные, нужно также выбрать соответствующий тип блюда и нажать кнопку «Показать» либо выбрать пункт «Все» и посмотреть все имеющиеся блюда.

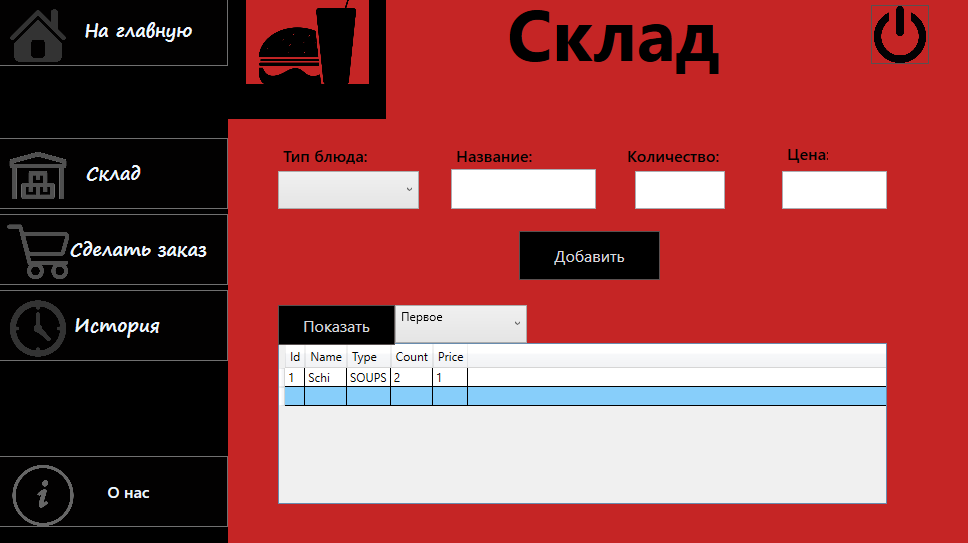


Рисунок 4.4 – Склад

При нажатии кнопки «История» переходим к просмотру истории заказов. Имеется два элемента DatePicker, позволяющие фильтровать данные по дате доставки. Если выбран первый элемент даты, то показываются заказы, доставка которых должна произойти после указанной даты. Если выбран второй элемент даты, то отображаются заказы, доставка которых должна произойти до указанной даты. Если даты выбраны на обоих элементах, то показываются заказы, доставка которых должна произойти в данном диапазоне дат (рисунок 4.5).

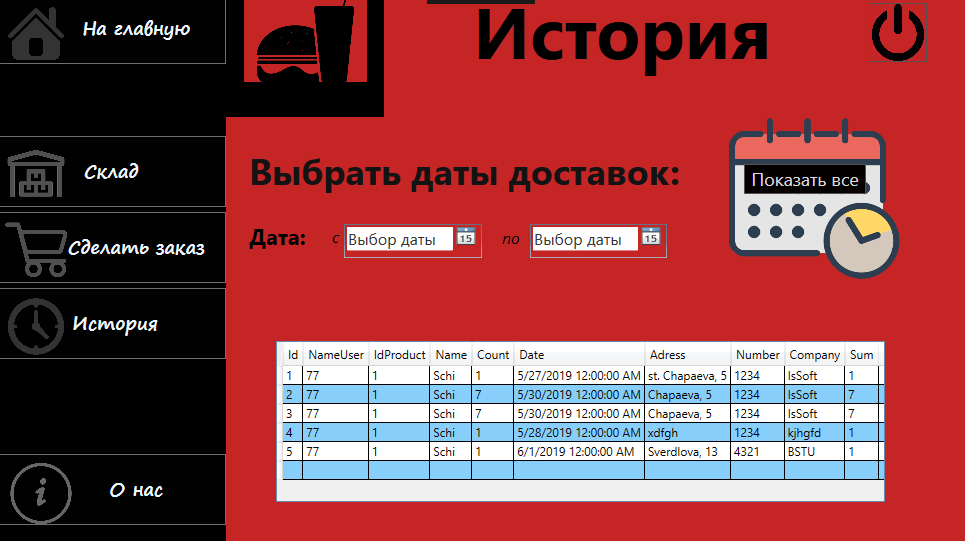


Рисунок 4.5 – История заказов

При нажатии на кнопку «Сделать заказ» открывается окно оформления заказа. Чтобы сделать заказ, необходимо заполнить все пункты необходимыми данными, правильность которых также проверяется, что будет показано в пункте «Тестирование». При успешном вводе данных можно посмотреть свой заказ и переходить к его оформлению (рисунок 4.6). При внезапном выходе из приложения данные о заказе обнуляются.

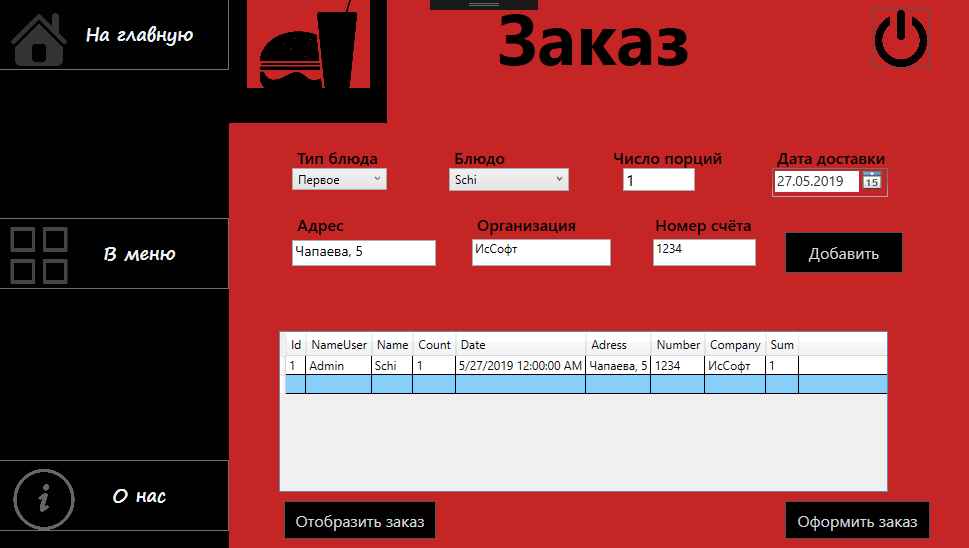


Рисунок 4.6 – Заказ

При нажатии на «Оформление заказа» пользователь переходит на конечную страницу, отображающую информацию о сумме заказа и дате доставки (рисунок 4.7).

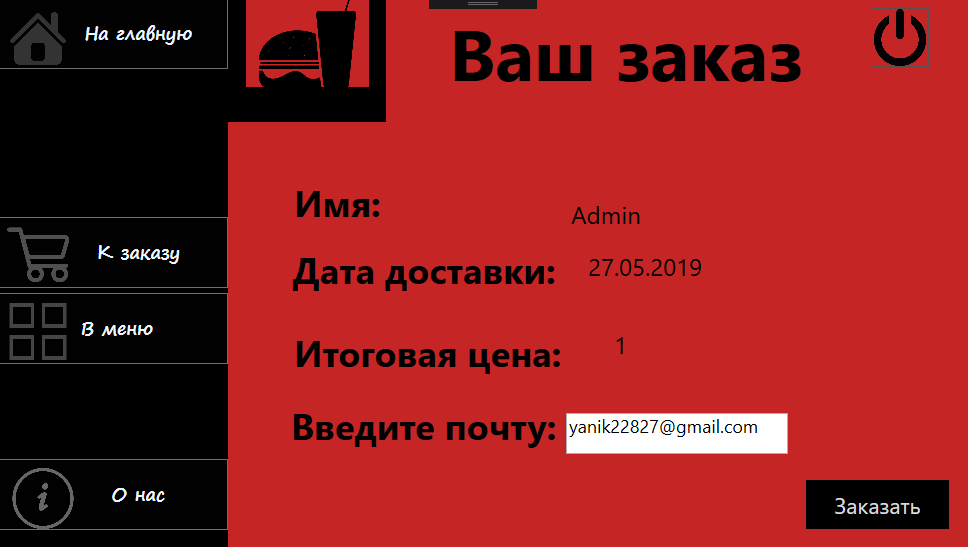


Рисунок 4.7 – Оформление заказа

Также у пользователя есть возможность получить письмо на почту с информацией о заказе. Для этого он должен ввести данные почты в TextBox на странице и ему придет письмо о заказе (рисунок 4.8).

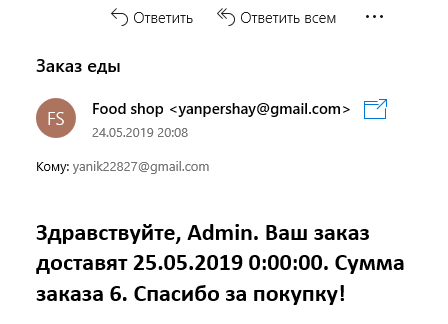


Рисунок 4.8 – Письмо на почте

# **Руководство для пользования программой Пользователем**

При входе отображается страница меню, которая дает возможность перехода в репертуар блюд или непосредственно к заказу (рисунок 4.9).



Рисунок 4.9 – Меню пользователя

При выборе пункта меню «Ассортимент», пользователь может посмотреть, какие блюда есть в наличии по категориям (рисунок 4.10).

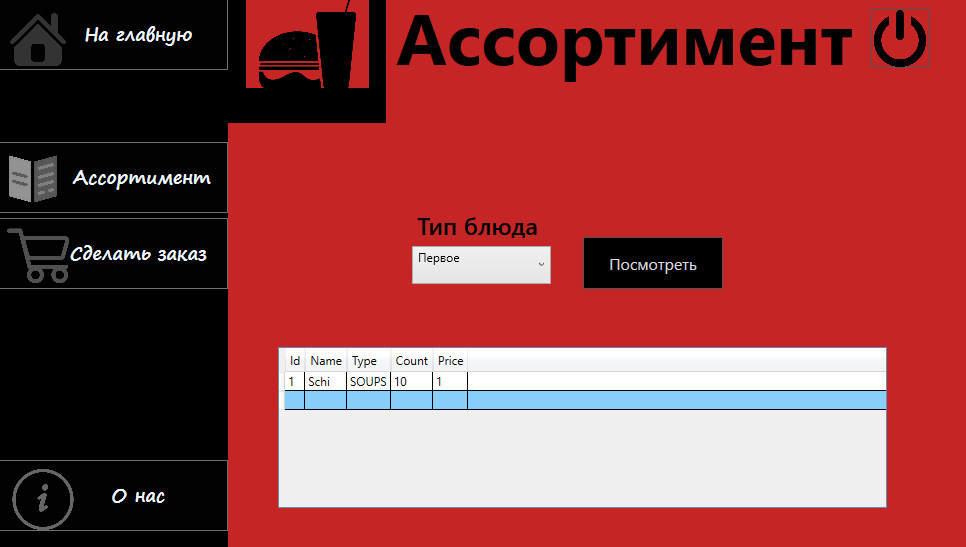


Рисунок 4.10 – Ассортимент

Осуществление и оформление заказа происходит аналогично данному процессу у Админа.

# **Тестирование**

Данное программное средство тестировалось вручную, были выполнены все технические моменты, предусмотренные реализацией, а также были предприняты попытки нарушить работу приложения.

* Изначально, при регистрации, при попытке ввода в окно логина символов, не соответствующих цифрам, выскакивало сообщение об ограничении данного поля (рисунок 5.1)

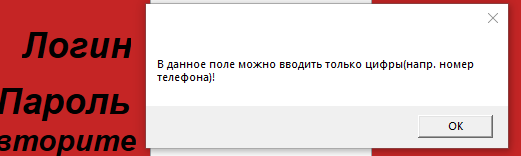


Рисунок 5.1— Коррекция ввода логина

* При попытке оставить поля пустыми, также выскакивало сообщение с предупреждением (рисунок 5.2).

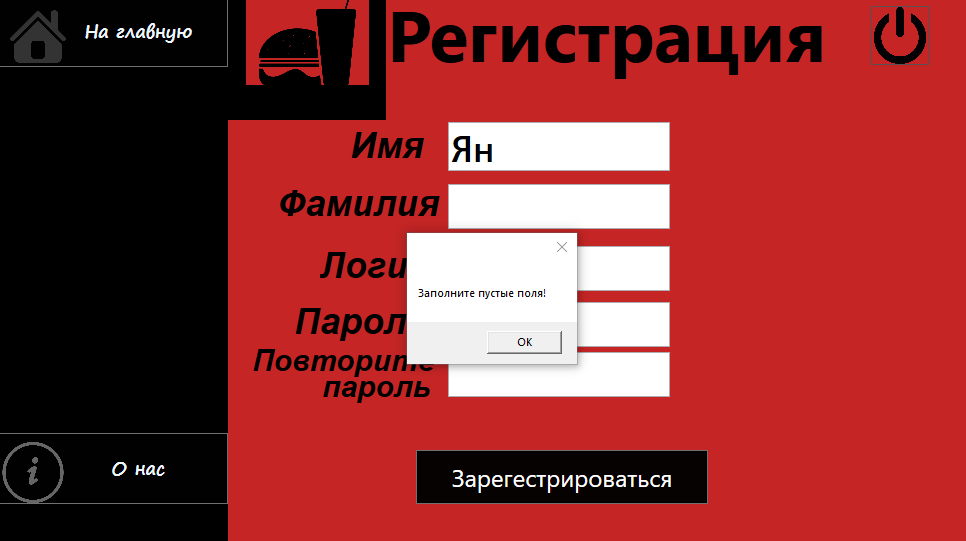


Рисунок 5.2 — Пустые поля

* При неправильном повторении пароля выскакивает сообщение (рисунок 5.3).



Рисунок 5.3— Обязательное заполнение полей

* Проверка на длину логина, аналогично и пароля (рисунок 5.4).

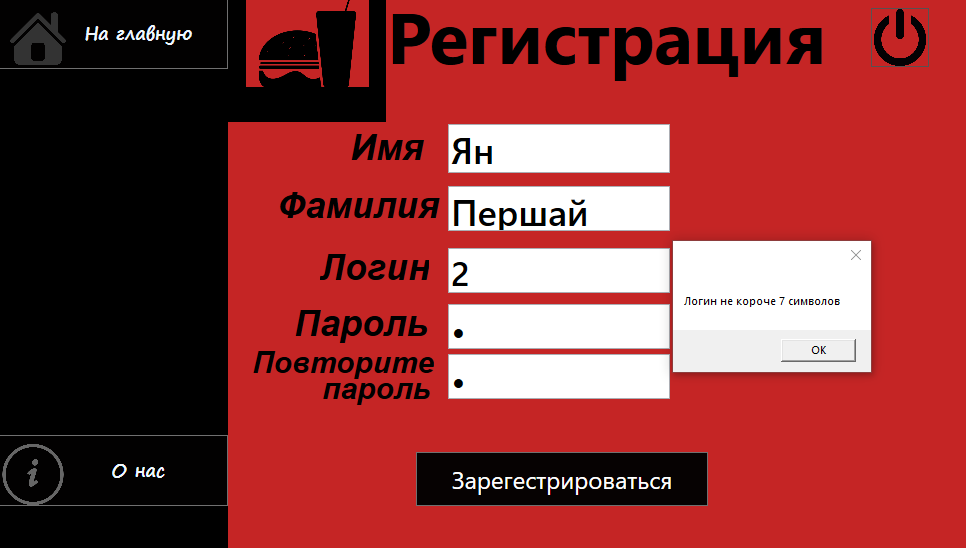


Рисунок 5.4 — Проверка на длину логина

* Если пользователь с данным логином существует, то приложение отказывается его регистрировать (рисунок 5.5).

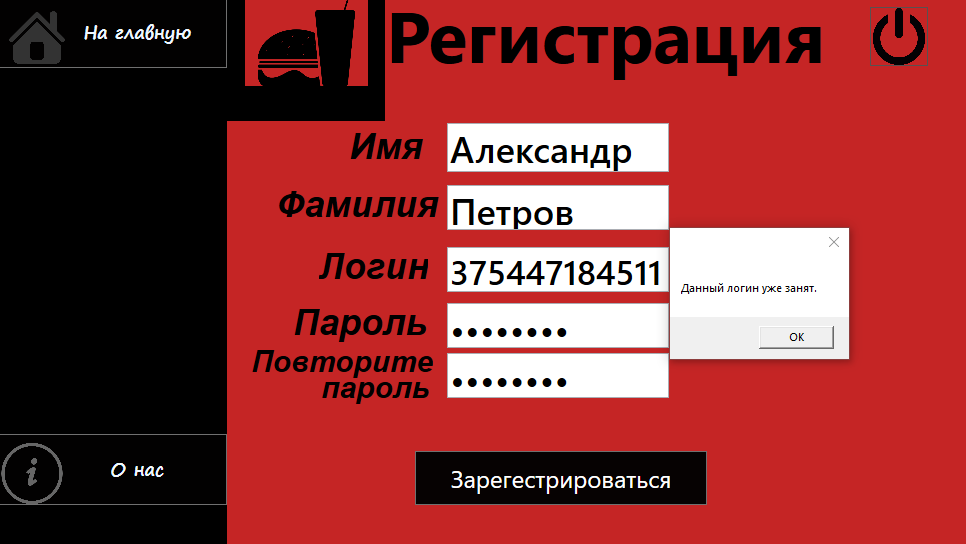


Рисунок 5.5— Проверка на существование логина

* Валидация правильного диапазона дат при просмотре истории заказов(рисунок 5.6).

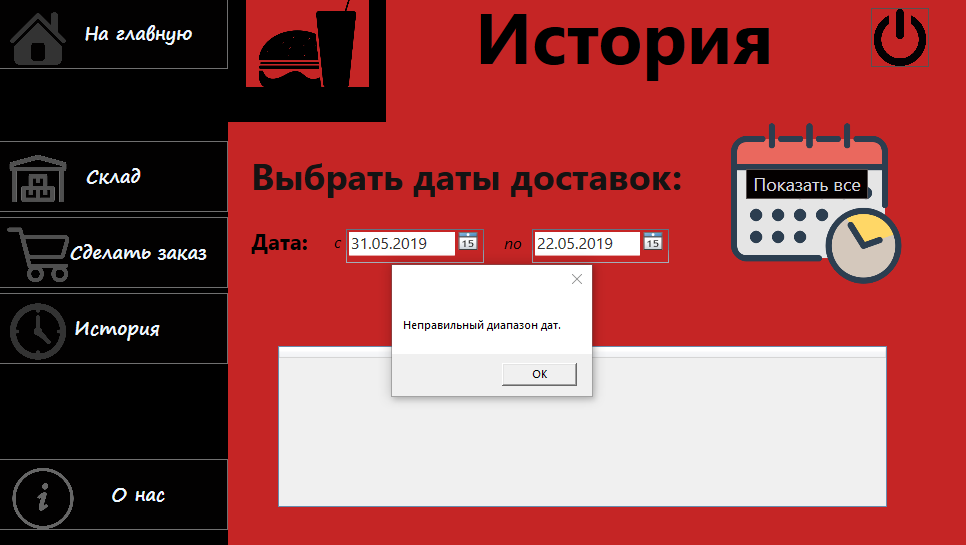


Рисунок 5.6— Проверка на диапазон дат

* Проверка на заполнение полей также есть при добавлении блюда на склад, при просмотре ассортимента и при покупке товара. Также, нельзя выбрать дату доставки раньше сегодняшней (рисунок 5.7).

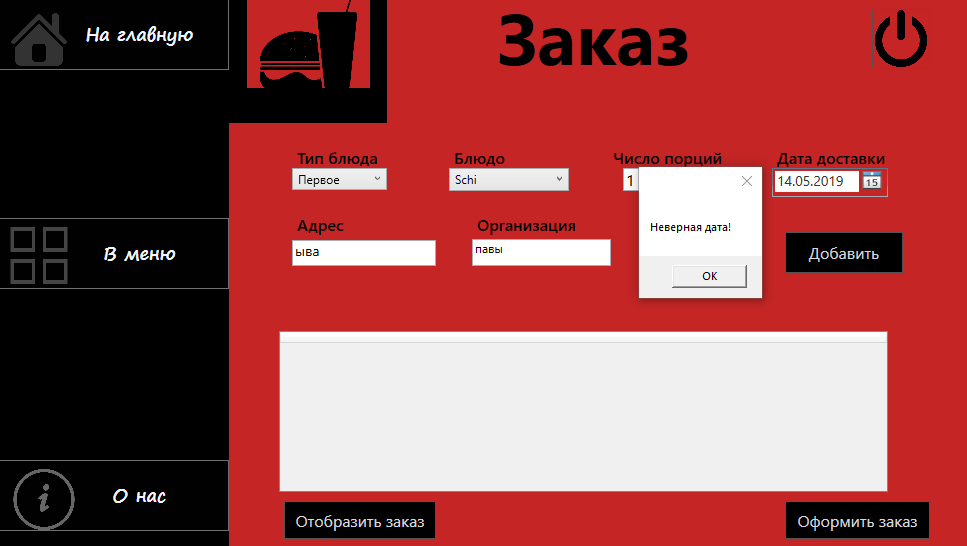


Рисунок 5.7— Проверка на верную дату

* Нельзя заказать порций больше, чем их есть на складе (рисунок 5.8).

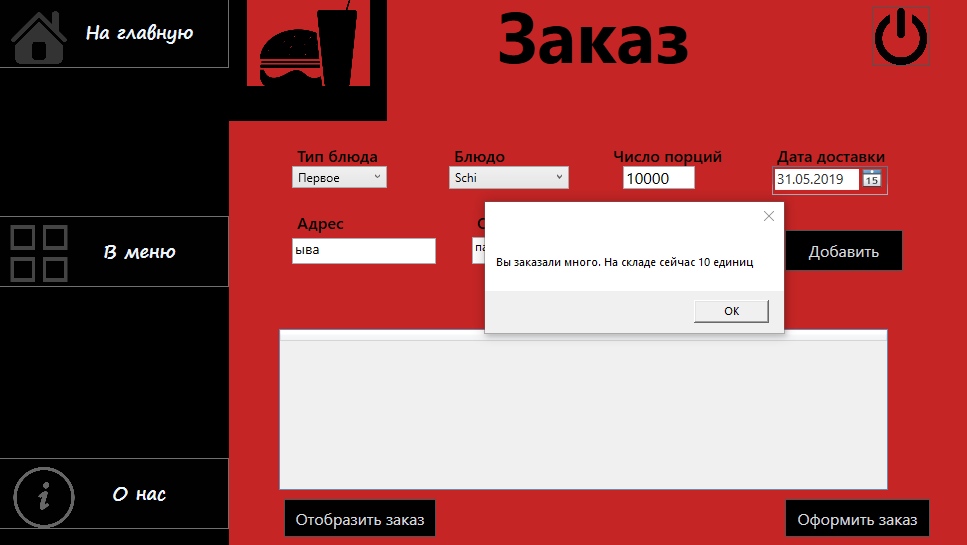


Рисунок 5.8— Проверка на количество порций

Таким образом, организация проверки вводимых данных позволит защитить приложение от некорректного ввода, ошибок в БД и исключений.

**Заключение**

Таким образом, в курсовом проекте реализовано программное средство с использованием технологий C# .NET Framework, SQL, ADO.NET, WPF.

В ходе реализации приложения был реализован функционал для заказа еды, имеющий две основных роли: Администратор и Пользователь и позволяющий:

* Регистрироваться
* Пополнять и смотреть содержимое склада
* Смотреть историю заказов
* Осуществлять заказ еды с отправкой письма на почту

Данное приложение имеет простой интерфейс, красные тона, что вызывает аппетит при восприятии у человека, основной функционал, соответствующий стандартному приложению для заказа еды.

# **Список используемой литературы**

1. Полное руководство по языку программирования С# 7.0 и платформе .NET 4.7. Режим доступа:  <https://metanit.com/sharp/tutorial/>
2. Пацей, Н.В. Курс лекций по языку программирования C# / Н.В. Пацей. – Минск: БГТУ, 2016.
3. Руководство по WPF [Электронный ресурс] – <https://metanit.com/sharp/wpf/>.
4. Мак-Дональд М. «WPF Windows Presentation Foundation в .NET 4.0 с примерами на C# 2010 для профессионалов (2011)»
5. Дэвид Сеппа «Программирование на Microsoft ADO.NET 2»
6. Руководство по ADO.NET [Электронный ресурс] – <https://metanit.com/sharp/wpf/>.

# **Приложение А**

**Скрипты для создания таблиц Базы Данных «FoodShop»**

use FoodShop

go

CREATE TABLE REGISTRATION(

ID nvarchar(50) primary key,

NAME NVARCHAR(20) NOT NULL,

SURNAME NVARCHAR(20) NOT NULL,

LOGIN NVARCHAR(12) NOT NULL,

PASSWORD NVARCHAR(15) NOT NULL

);

INSERT INTO REGISTRATION VALUES ('77', 'Admin', 'Admin', '-', '-')

CREATE TABLE PRODUCTS(

ID INT IDENTITY(1,1) primary key,

TYPE NVARCHAR(50),

NAME NVARCHAR(99),

COUNTS INT,

PRICE FLOAT

);

CREATE TABLE ORDERS(

ID INT IDENTITY(1,1),

USER\_ID NVARCHAR(50) foreign key references Registration(ID),

ID\_PRODUCT int foreign key references Products(ID),

NAME\_PRODUCT NVARCHAR(50),

COUNTS INT,

DATE DATE,

ADRESS NVARCHAR(99),

COMPANY NVARCHAR(99),

NUMBE\_ACCOUNT NVARCHAR(99),

SUMM float

);

CREATE TABLE USER\_ORDER(

ID INT IDENTITY(1,1),

USER\_NAME NVARCHAR(99),

NAME NVARCHAR(99),

COUNTS INT,

DATE DATE,

ADRESS NVARCHAR(99),

COMPANY NVARCHAR(99),

NUMBE\_ACCOUNT NVARCHAR(99),

SUMM float

);

# **Приложение Б**

**Добавление нового пользователя и проверка на уникальность логина**

Guid guid = Guid.NewGuid();

string sqlExpression = $"SELECT \* FROM REGISTRATION WHERE LOGIN = '{LoginBox.Text}'";

string sqlExpression2 = $"INSERT INTO REGISTRATION (ID, NAME, SURNAME, LOGIN, PASSWORD) VALUES('{guid}','{NameBox.Text}','{SurnameBox.Text}','{LoginBox.Text}','{PassBox1.Password}')";

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand sqlCommand = new SqlCommand(sqlExpression, connection);

SqlDataReader sqlDataReader = sqlCommand.ExecuteReader();

if (sqlDataReader.HasRows)

{

MessageBox.Show("Данный логин уже занят.");

}

else

{

sqlDataReader.Close();

SqlCommand sqlCommand2 = new SqlCommand(sqlExpression2, connection);

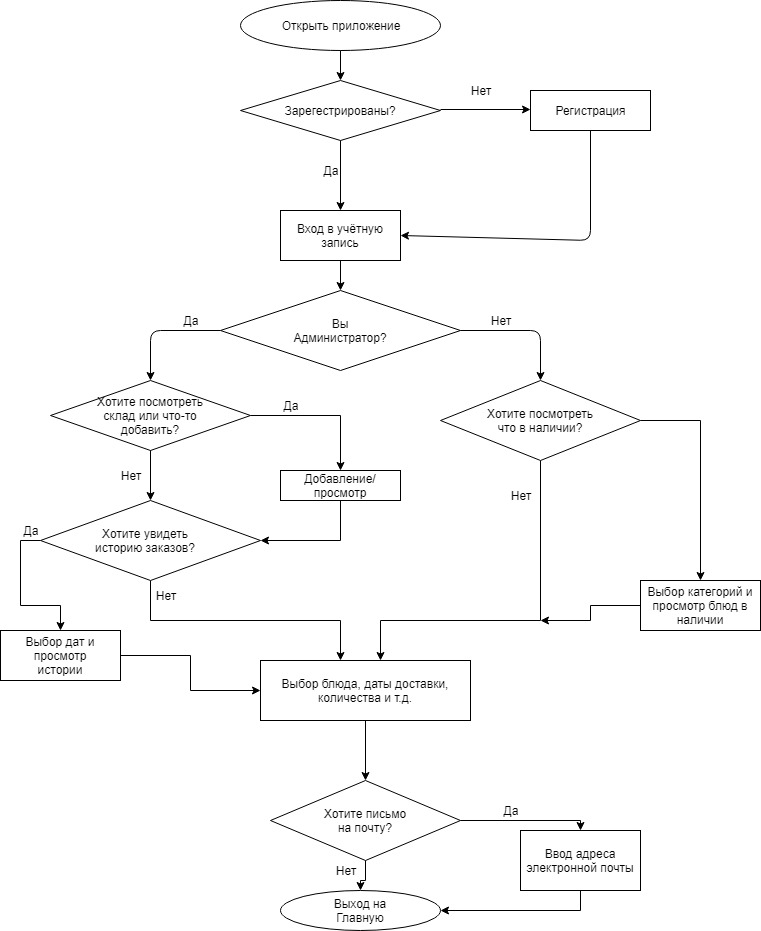
int number = sqlCommand2.ExecuteNonQuery();

connection.Close();

}

}

# **Приложение В**

**Блок-схемы логики приложения**