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 «Объектно-ориентированное программирование»

Тема Программное средство «Логистическая компания»

Исполнитель

студент 2 курса группы 8 Рыжук Е.В.

(Ф.И.О.)

Руководитель работы ассистент Радиванович Д.А

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

Курсовой проект защищен с оценкой

Председатель Пацей Н.В.

(подпись)

Минск 2020

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования   
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий   
Кафедра программной инженерии

Утверждаю

Заведующий кафедрой

\_\_\_\_\_\_\_\_ \_\_ Н.В. Пацей

подпись инициалы и фамилия

“\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2020г.

**ЗАДАНИЕ**

**к курсовому проектированию**

**по дисциплине** "Объектно-ориентированное программирование"

|  |  |
| --- | --- |
| Специальность: 1-40 05 01 Информационные системы и технологии  Студент: Рыжук Е.В. | Группа: \_2\_\_ |
| **Тема: Программное средство «Логистическая компания»** | |

**1. Срок сдачи студентом законченной работы**: "20 мая 2020 г."

**2. Исходные данные к проекту:**

**2.1**. Функционально ПС поддерживает:

* Выполнять функции администратора производства:
  + Поддерживать работу c базой данных;
  + Оповещать клиента о выполнении заказа;
  + Выполнять поисковые запросы;
* Выполнять функции клиента для пользователя:
  + Выполнять регистрацию и авторизацию;
  + Заполнять форму заказа по заданным параметрам;
  + Совершать заказы по определённым параметрам.
  + Просматривать информацию о статусе заказа.

**2.2.** При выполнении курсового проекта необходимо использовать принципы и приемы ООП. Приложение разрабатывается под ОС Windows и представляет собой настольное приложение (desktop). Отображение, бизнес логика должны быть максимально независимы друг от друга для возможности расширения. Диаграммы вариантов использования, классов реализации задачи, взаимодействия разработать на основе UML. Язык разработки проекта – C#. Управление программой должно быть интуитивно понятным и удобным. При разработке использовать несколько наиболее подходящих шаблонов проектирования ПО.

**3. Содержание расчетно-пояснительной записки**

(перечень вопросов, подлежащих разработке)

* Введение
* Постановка задачи и обзор литературы (алгоритмы решения, обзор прототипов, актуальность задачи)
* Проектирование архитектуры проекта (структура модулей, классов).
* Разработка функциональной модели и модели данных ПС (выполняемые функции)
* Тестирование
* Заключение
* Список используемых источников
* Приложения

**4. Форма представления выполненной курсового проекта:**

* + Теоретическая часть курсового проекта должны быть представлены в формате docx. Оформление записки должно быть согласно выданным правилам.
  + Листинги программы представляются частично в приложении.
  + Пояснительную записку, листинги, проект (инсталляцию проекта) необходимо загрузить диск, указанный преподавателем.

#### Календарный план

|  |  |  |  |
| --- | --- | --- | --- |
| №  п/п | Наименование этапов курсового проекта | Срок выполнения этапов проекта | Примечание |
| 1 | Введение | 19.02.2020 |  |
| 2 | Аналитический обзор литературы по теме проекта. Изучение требований, определение вариантов использования | 12.03.2020 |  |
| 3 | Анализ и проектирование архитектуры приложения (построение диаграмм, проектирование бизнес-слоя, представления и данных) | 26.03.2020 |  |
| 4 | Проектирование структуры базы данных. Разработка дизайна пользовательского интерфейса | 2.04.2020 |  |
| 5 | Кодирование программного средства | 23.04.2020 |  |
| 6 | Тестирования и отладка программного средства | 30.04.2020 |  |
| 7 | Оформление пояснительной записки | 7.05.2020 |  |
| 9 | Сдача проекта | 20.05.2020 |  |

**5. Дата выдачи задания \_\_\_\_**12.02.2020**\_\_\_\_**

Руководитель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *Д.А. Радиванович*

(подпись)

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата и подпись студента)

# **ВВЕДЕНИЕ**

Темой моего курсового проекта является разработка приложения для логистической компании.

В современном месте каждый день происходит очень много различных перемещений. Каждая компания пользуется услугами логистического назначения. Изо дня в день грузовики перевозят миллионы тонн грузов, корабли доставляют их по воде, грузовые самолеты по воздуху, поезда по железной дороге. Это одна из самых важных отраслей в мире во все времена. Исходя из этого, существует огромная конкуренция между компаниями, представляющими данную сферу. В лидеры входят компании с наилучшей репутацией, которые могут гарантировать доставку груза в целостности и в сроки, которые лучше относятся к своим клиентом, однако важной частью является и способ взаимодействия с клиентом, а так же скорость обработки заказов. Таким образом, если компания старается составить конкуренцию, ей нужно приложение, удобное как для пользователей, так и для администраторов. Такое приложения я создавал в своем курсовом проекте.

Оглавление

[**ВВЕДЕНИЕ** 4](#_Toc41659926)

[**1.** **Постановка задачи** 6](#_Toc41659927)

[**1.1** **Основные задачи** 6](#_Toc41659928)

[**1.2** **Обзор прототипов** 6](#_Toc41659929)

[**1.3** **Информационные системы и технологии** 8](#_Toc41659930)

[**2.** **Разработка архитектуры проекта** 10](#_Toc41659931)

[**2.1** **Обобщённая структура** 10](#_Toc41659932)

[10](#_Toc41659933)

[**2.2 Диаграммы UML, взаимосвязь всех компонентов** 11](#_Toc41659934)

[**3** **Разработка функциональной модели и модели данных** 13](#_Toc41659935)

[**3.1** **Модель базы данных** 13](#_Toc41659936)

[**3.2** **Диаграмма последовательности** 15](#_Toc41659937)

[**3.3** **Диаграмма деятельности** 17](#_Toc41659938)

[**4** **Создание (реализация) программного средства** 18](#_Toc41659939)

[**4.1** **Реализация библиотеки классов** 18](#_Toc41659940)

[**4.2** **Подключение базы данных к проекту** 19](#_Toc41659941)

[**4.3** **Реализация авторизации и регистрации пользователей** 19](#_Toc41659942)

[**4.4** **Реализация работы с данными** 21](#_Toc41659943)

[**5. Тестирование, проверка работоспособности и анализ полученных результатов** 22](#_Toc41659944)

[**6**  **Руководство по установке и использованию** 24](#_Toc41659945)

[**6.1 Использование приложения обычным пользователем** 24](#_Toc41659946)

[**6.2 Использование приложения администратором** 27](#_Toc41659947)

[**Список использованных источников** 29](#_Toc41659948)

[**Приложение А** 30](#_Toc41659949)

# **Постановка задачи**

# **Основные задачи**

Основной задачей моего программного средства является предоставление функционала, позволяющего пользователю оформить заказ на перевозку, а администратору управлять заказами.

Программное средство выполняет следующие задачи для администратора:

* Просмотр всех заказов
* Просмотр принятых и выполненных заказов
* Возможность удаления заказов
* Поиск и сортировка заказов

Программное средство выполняет следующие задачи для пользователя:

* Регистрация
* Просмотр всех собственных заказов
* Просмотр выполненных и принятых заказов
* Оформление заказа
* Изменение личных данных

# **Обзор прототипов**

В обзоре прототипов я решил рассмотреть приложение логистической компании Aastra.

Данная компании занимается перевозкой грузов по всему миру.

В данном программном средстве для клиентов отсутствует регистрация, они оставляют заявку, после чего с ними связывается менеджер и обсуждает детали сделки. В приложении можно просмотреть список услуг, информация о компании, ее особенности.



Рисунок 1.1 – Логотип приложения Aastra

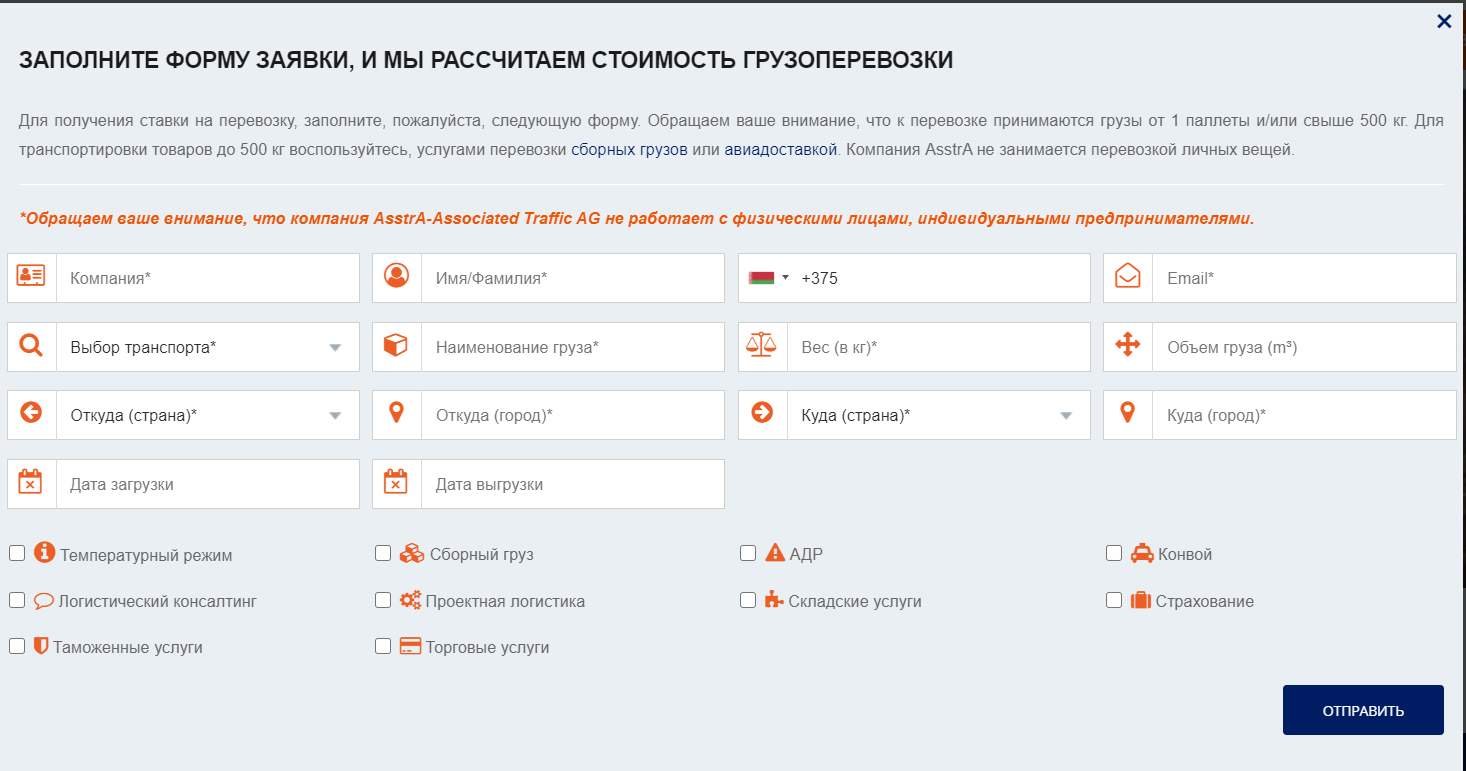


Рисунок 1.2 – Форма оформления заказа

Компания предоставляет учитывает все особенности груза, что определенно является плюсом (Рис 1.2). Требуется ввести довольно много информации, что является обычным в данной сфере.

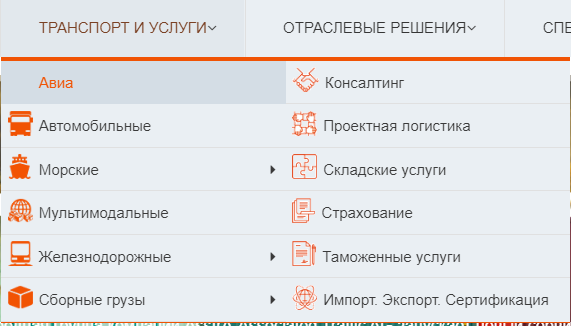


Рисунок 1.3 Спектр услуг и транспорта

Помимо обычных грузоперевозок, компания предоставляет и другие услуги, перечень которых можно легко посмотреть.

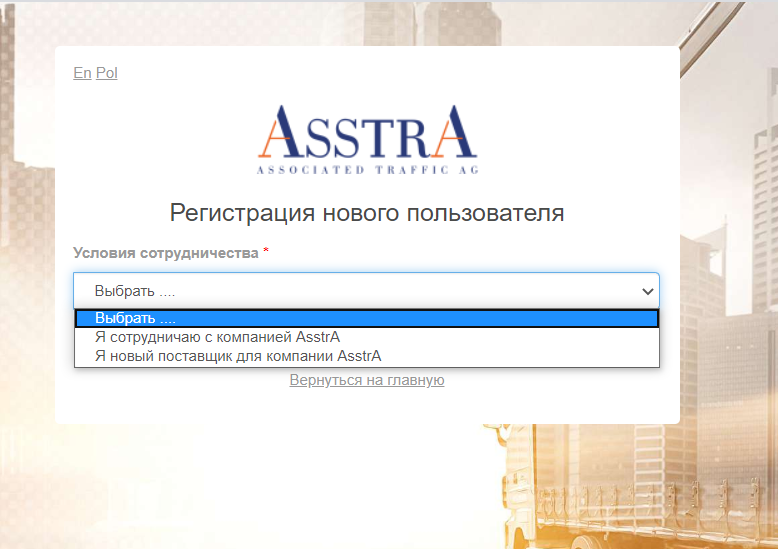


Рисунок 1.4 Регистрация

Для клиентов регистрация отсутствует, однако для поставщиков, а так же сотрудников своей компании, данная функция присутствует. Зарегистрированный пользователь будет иметь большие возможности.

Я считаю, что данное приложение довольно неплохое, однако было бы удобнее, если бы мы имели возможность зарегистрироваться, после чего могли просматривать историю своих заказов и следить за их статусом.

# **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**

Структура проекта представлена на следующем скриншоте(Рис.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

|  |  |
| --- | --- |
| Ссылки | Перечень сборок, используемых в проекте |
| Classes | Папка со свойствами, предназначенных для хранения характеристик элементов |
| Продолжение таблицы 2.1. | |
|  |  |
| App.xaml | Класс Application |
| Account | Страница с данными о пользователе | |
| AddOrder | Страница для добавления заказа |
| MainWindow | Окно отображения Главной страницы |
| AdminDataPage | Страница, содержащая элемент для отображения заказов администратору |
| UserDataPage | Страница, содержащая элемент для отображения заказов пользователю |
| AdminMainPage | Главная страница администратора |
| UserMainPage | Главная страница пользоавтеля |
| AuthorizationPage | Страница авторизации |
| RegistrationPage | Страница регистрации |

# **2.2 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.

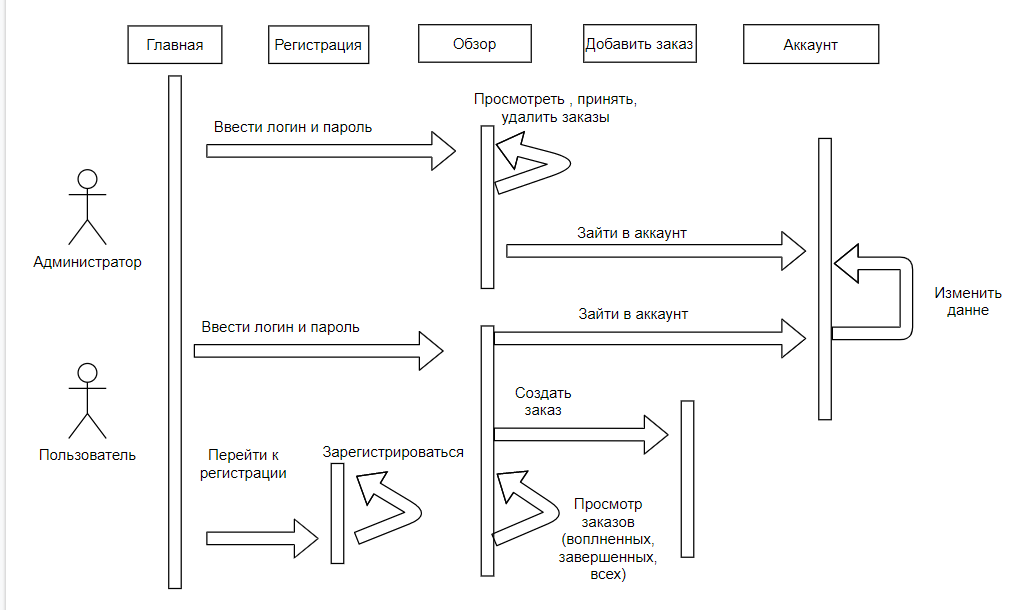


Рисунок 2.2 Диаграмма последовательности

# **Разработка функциональной модели и модели данных**

# **Модель базы данных**

Все данные будут постоянно храниться в базе данных. Мне понадобилось 3 таблицы, которые были созданы при помощи скрипта в СУБД SQL Server 2012 (Рисунок 3.1).

В базе данных LogisticApp представлены следующие таблицы:

* Orders (Заказы)
* Goods (Товары)
* Users (Пользователи)

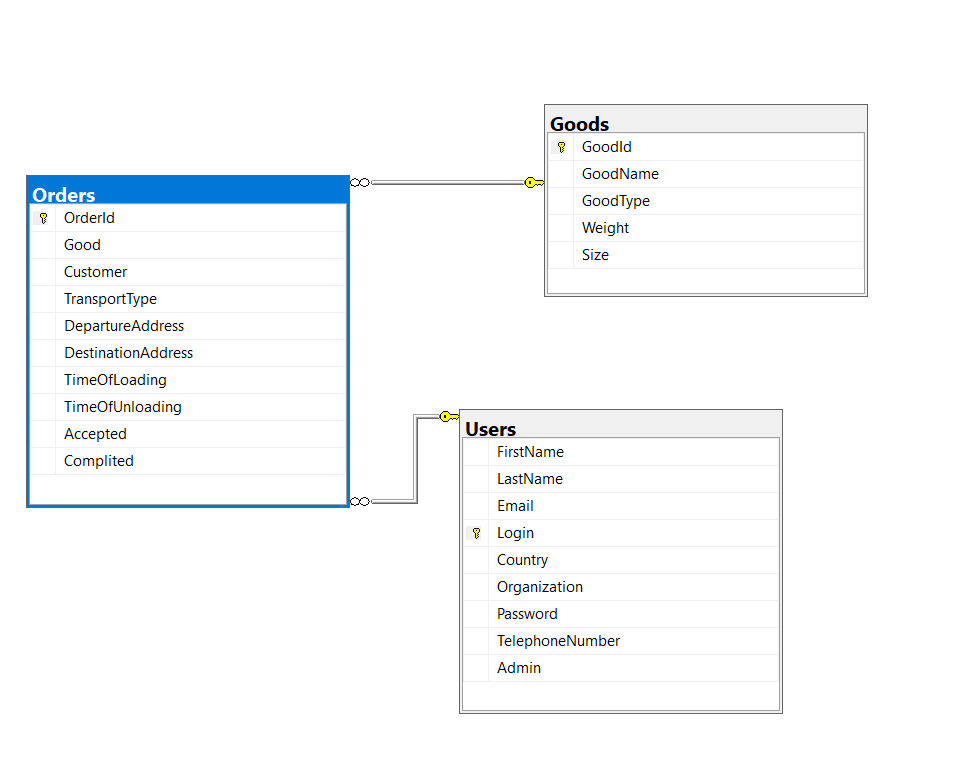


Рисунок 3.1 Диаграмма базы данных

Таблица Users используется для авторизации. При регистрации пользователи добавляются в данную базу данных. Orders хранит всю подробную информацию о заказах, а Goods содержит информацию о товарах.

Далее мы рассмотрим структуру каждой из таблиц.

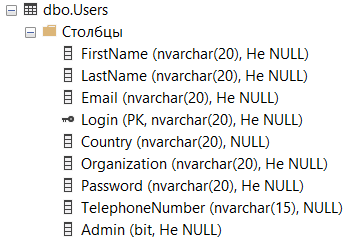


Рисунок 3.2 Таблица Users.

Таблица Users (Рисунок 3.2) содержит в себе следующие поля:

* FirstName (Имя)
* LastName (Фамилия)
* Email
* Login
* Country (страна)
* Organization (организация)
* Password (пароль)
* TelephoneNumber (Номер телефона)
* Admin (Является ли пользователь администратором)

Первичным ключем является Login.

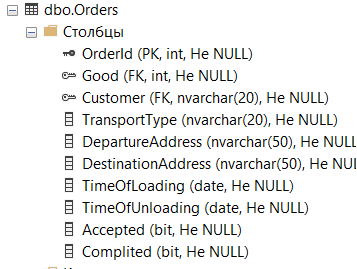


Рисунок 3.3 Таблица Orders

* Таблица Orders:
* OrderId(Номер заказа)
* Good(Номер товара)
* Customer(Заказчик)
* TransportType(Тип Транспорта)
* DepartureAddress (Адрес отправки)
* DestinitionAddress (Адрес получателя)
* TimeOfLoafing (Нижняя граница даты, в интервале которой заказ должен быть выполнен)
* TimeOfUnLoading(Верхняя граница даты, в интервале которой заказ должен быть выполнен)
* Accepted(Принят ли заказ)
* Complited(Выполнен ли заказ)

Первичным ключом является номер заказа. Номер товара и заказчик связана с таблицами Goods и Users соответственно и являются внешними ключами.

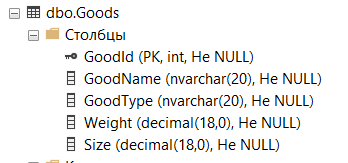


Рисунок 3.4 таблица Goods

Таблица Goods связана с таблицей Orders по первичному ключу GoodId:

* GoodId (Номер товара)
* GoodName(Название Товара)
* GoodType(Тип товара)
* Weight(Масса товара)
* Size(Объем товара)

# **Диаграмма последовательности**

Для более детального показа логики, мы создаем диаграмму последовательности. Она уточняет диаграмму прецедентов, тем самым показывая нам более точную картину. Данная диаграмма расположена в приложении А.

На данной диаграмме показана последовательность действий, происходящая при оформление нового заказа. Чтобы оформить заказ, пользователь должен перейти на страницу добавления заказа, затем заполнить форму корректными данными, отправить, после чего в он добавится в базу данных как непринятый и невыполненный, и после этого дождаться, когда администратор примет заказ. Когда это случится, пользователю придет уведомление.

## **Диаграмма деятельности**

Дальше надо было разработать диаграмму деятельности. Эта диаграмма предоставляет для рассмотрения блок-схему, на которой наглядно отображается, как поток управлений переходит от одной деятельности к другой.

В приложении Д представлена диаграмма деятельности приложения «Аренда строительной техники»

Диаграмма деятельности позволяет любому, кто выполняет данный процесс, выбирать порядок действий. Другими словами, диаграмма только устанавливает правила обязательной последовательности действий, которым пользователь должен следовать. Это важно для моделирования бизнес-процессов, поскольку эти процессы часто выполняются параллельно.

# **Создание (реализация) программного средства**

### **Реализация библиотеки классов**

В работе использовался паттерн MVVM, таким образом,были структура делилась на model, viewmodel и view. Диаграмма классов показывает отношения между классами(Рис.4.1).

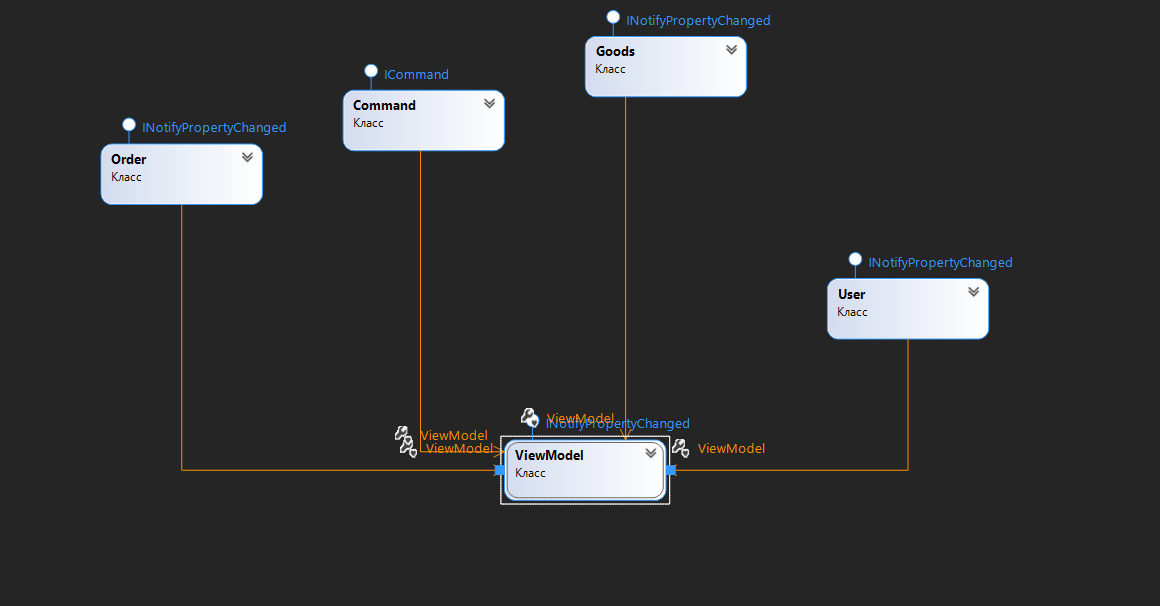


Рисунок 4.1 Диаграмма классов

В model были определены классы, с которыми я работал, а так же реализован интерфейс INotifyPropertyChange. Было реализовано 3 модели данных, что соответствует количеству таблиц в базе данных. Во viewmodel реализовывалась вся логика приложения. Был создан класс Command, который использовался для связи между view и viewmodel через команды.

# **Подключение базы данных к проекту**

Данный проект подразумевает использование базы данных. База данных, которую я использовал очень распространенная: Microsoft SQL server 2012. В качестве способа подключения я выбрал ADO.NET. Эта технология позволяет довольно просто получить доступ к базе данных. С помощью ADO мы можем выполнять стандартные запросы используемой базы данных. На начальном этапе был написан скрипт для создания базы данных, ее таблиц, а также занесены некоторые изначальные данные. Потом было реализовано подключение. Чтобы подключиться, мы должны в AppConfig добавить строку подключения (Рис 4.2).

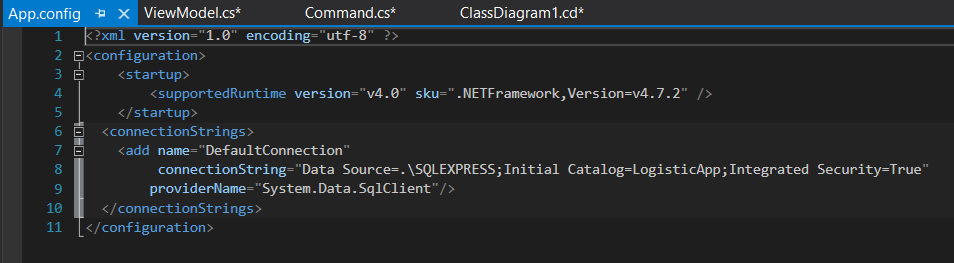


Рисунок 4.2 Строка подключения

# **Реализация авторизации и регистрации пользователей**

Для регистрации пользователей использовалась команда, в которой применяется валидация через регулярные выражения. Если валидация прошла, вызывается метод regist(), который и выполняет регистрацию (Рис.4.3).

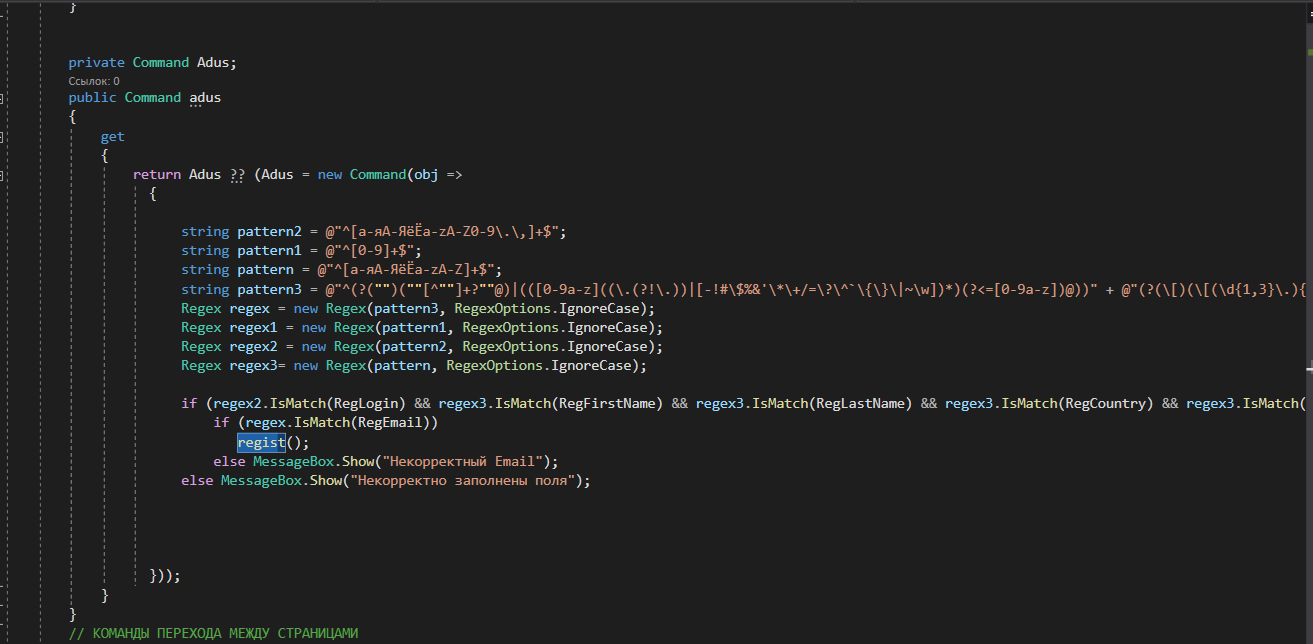


Рисунок 4.3 Команда регистрации

В команде авторизации вызывается метод CheckDB, который принимает введенные нами данные и проверяет, есть ли в базе данных пользователь с таким логином и паролем, если есть, открывается доступ, в противном случае — сообщение о том, что данные введены неверно(Рис.4.4).

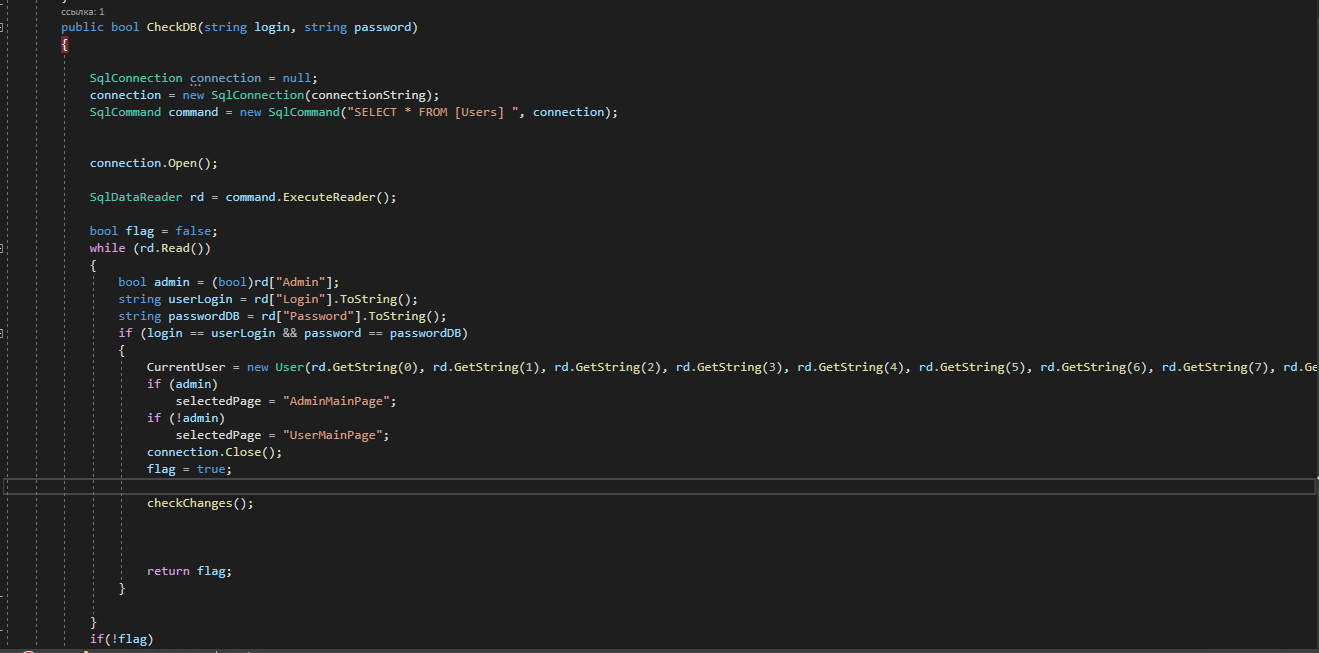


Рисунок 4.4 Метод для авторизации

# **Реализация работы с данными**

На данном этапе были разработаны функции удаления, добавления заказов, поиск, выборка элементов, (Приложение Б) а так же отправление сообщения пользователю в случае приема либо выполнения его заказа (Рис.4.5).

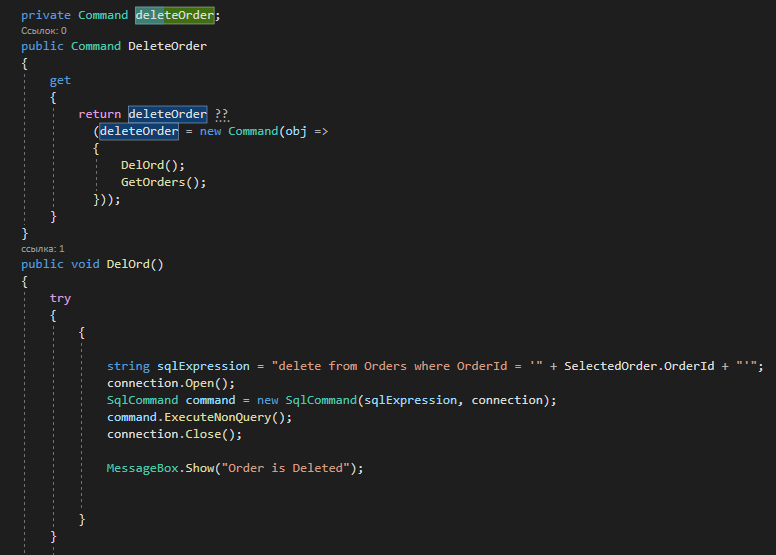


Рисунок 4.5 Пример функции

Вся работа с базой данный реализовывалась через ADO.NET. Кроме того, были провалидированы все поля для ввода(Рис.4.6).



Рисунок 4.6 Пример Функции с валидацией

# **5. Тестирование, проверка работоспособности и анализ полученных результатов**

Тестирование проводилось вручную, в результате него все данные вводятся исключительно корректно, валидируется абсолютно все. При попытке некорректного ввода данных пользователь получает соответствующее сообщение (Рис. 5.1).

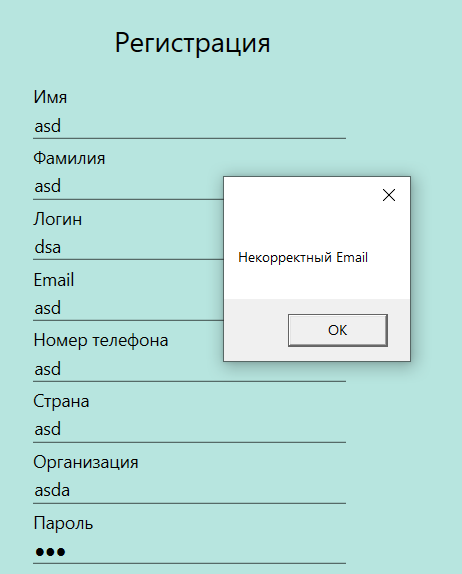


Рисунок 5.1 Попытка ввода некорректных данных

Так же, если пользователь при регистрации введет верные данные, однако, пользователь с заданным логином или Email уже будет зарегистрирован, то пользователя сообщат о данном факте (Рис 5.2).

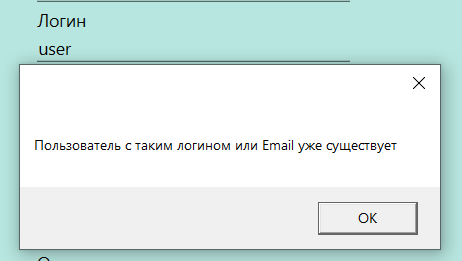


Рисунок 5.2 Проверка логина

Важным моментом данного проекта было оповещение пользователя о приеме его заказа, либо о его завершении. Данный функционал был реализован и работает исправно (рис.5.3).

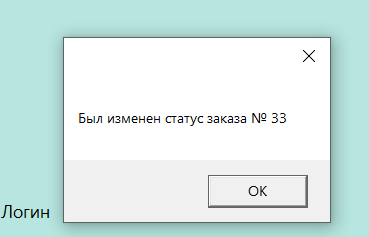


Рисунок 5.3 Оповещение пользователя о приеме либо выполнении заказа

Так же, при добавлении пользователе нового заказа, все данные провалидируются, после чего пользователю будет выдано сообщение о том, что заказ был создан (Рис. 5.4).

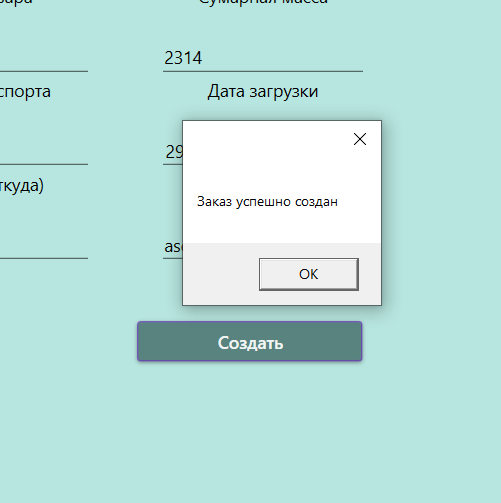


Рисунок 5.4 Создание заказа

# **6** **Руководство по установке и использованию**

# **6.1 Использование приложения обычным пользователем**

После запуска приложения пользователя встречает страница авторизации, где пользователю предлагают люби авторизоваться, либо зарегистрироваться.

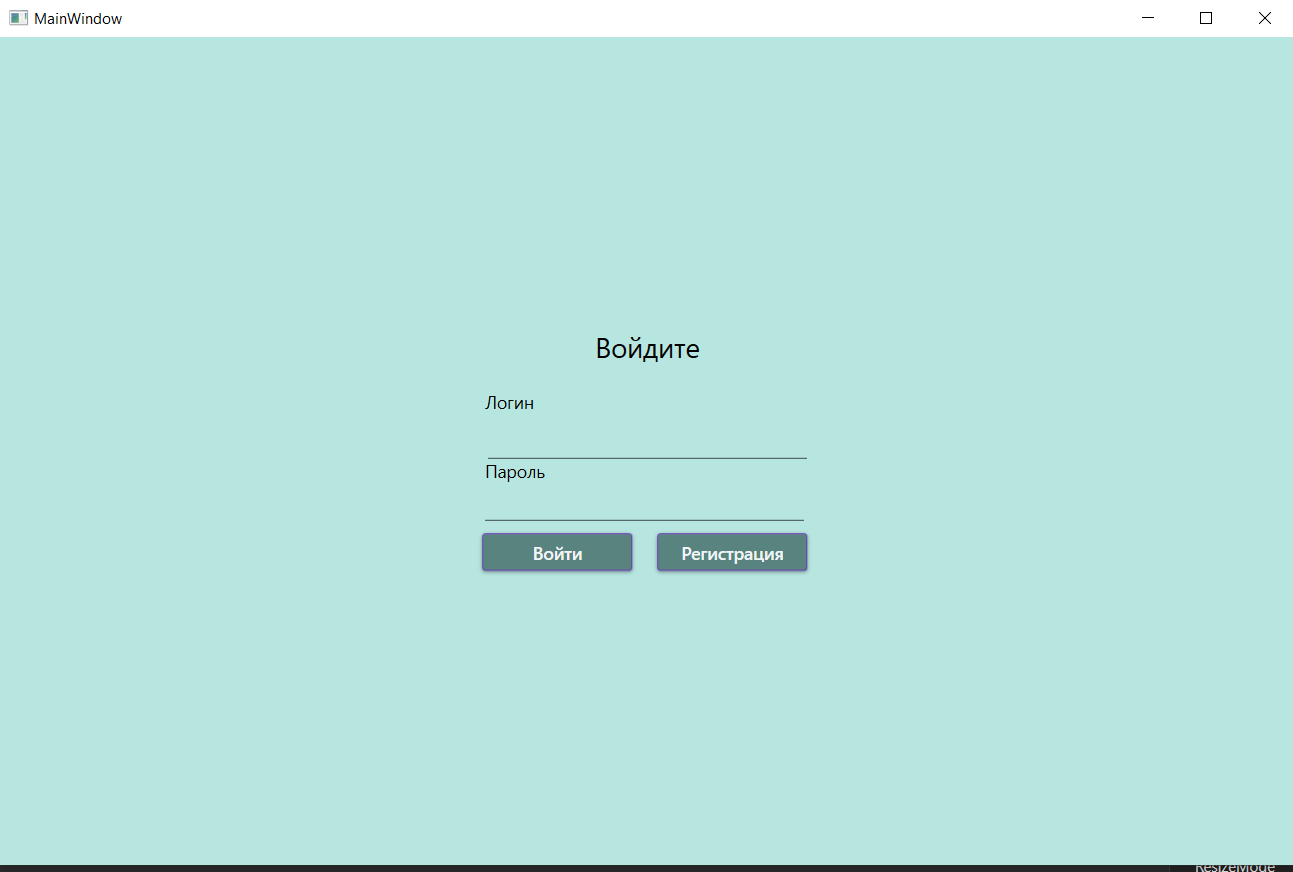


Рисунок 6.1 — Страница авторизации

В случае, если пользователь не зарегистрирован, он жмет кнопку регистрации и ему открывается страница, где нужно заполнить форму данными (Рис.6.2).

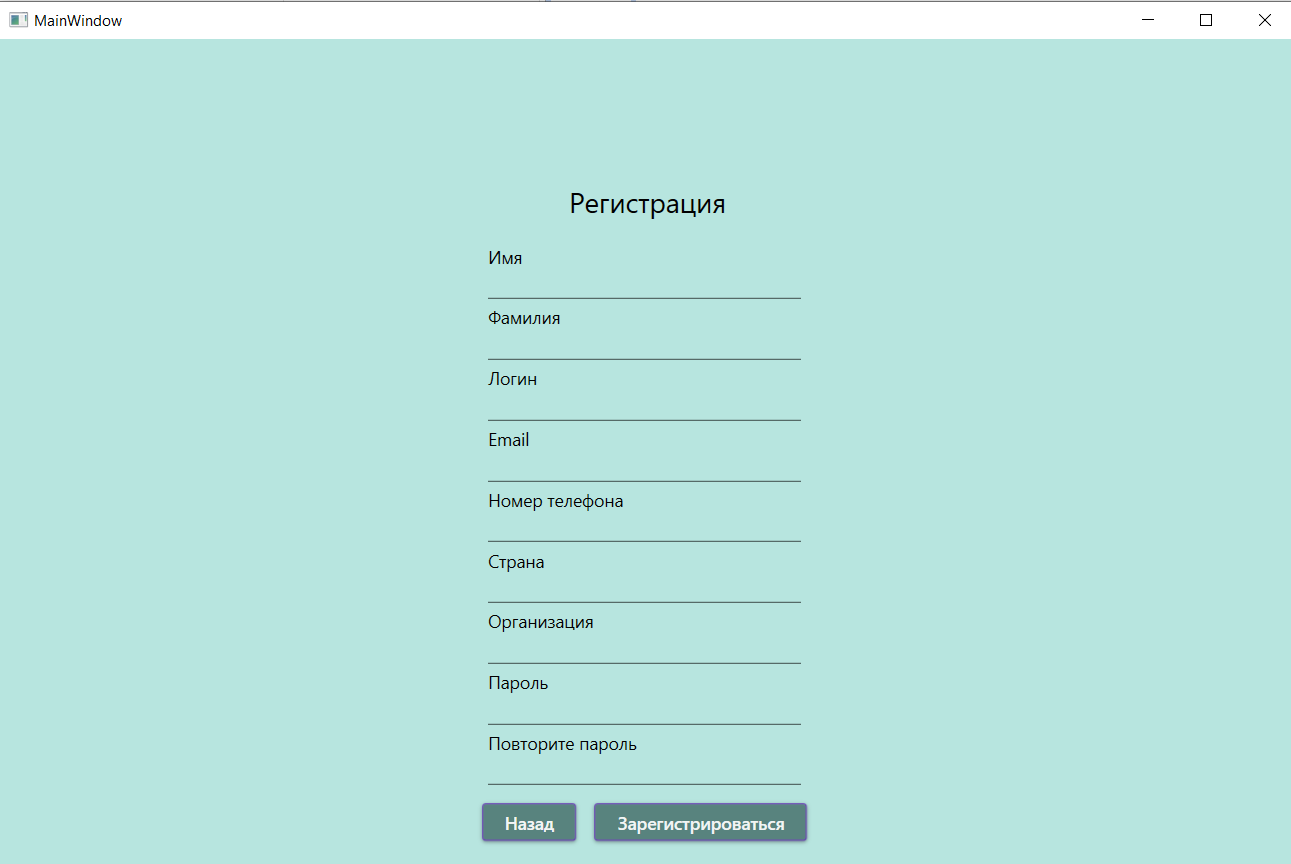


Рисунок 6.2 Страница регистрации

После того, как пользователь зарегистрировался, он возвращается на страницу авторизации, авторизуется и попадает на главную страницу (Рис6.3), где можно увидеть список оформленных данным пользователем товаров.

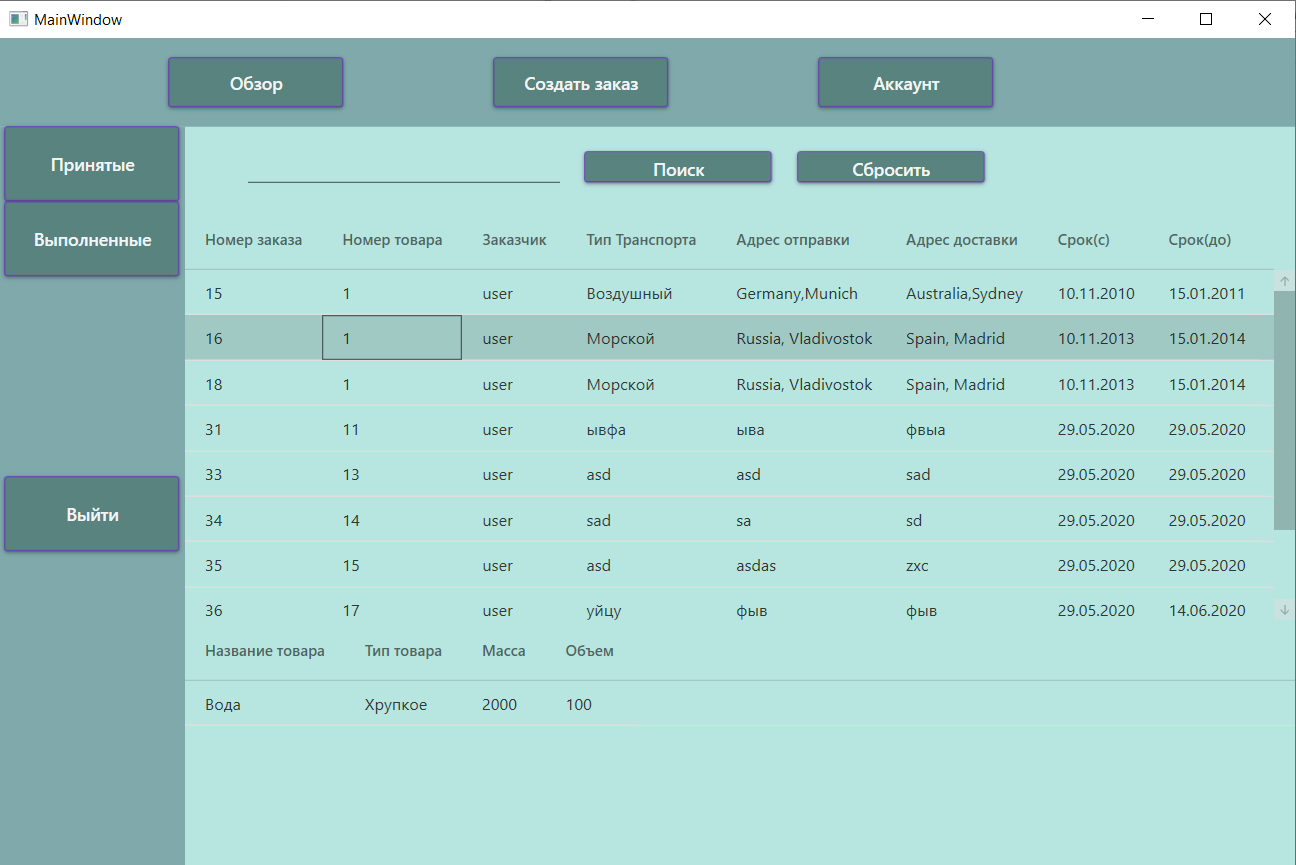


Рисунок 6.3 Главная страница пользователя

Затем он может просмотреть принятые либо выполненные заказы, перейти на страницу добавления нового заказа, а так же перейти на страницу с личными данными, где их можно изменить (Рис. 6.4).

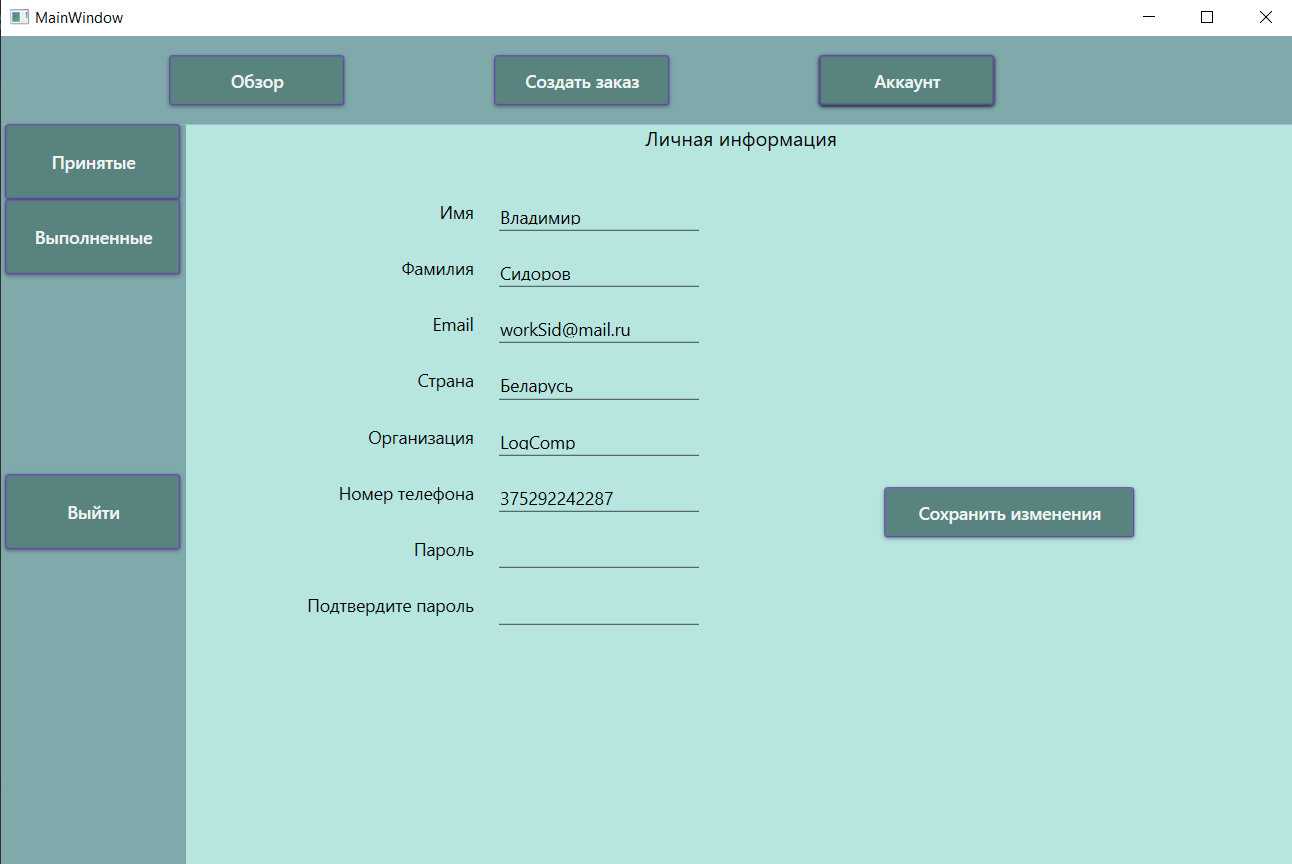
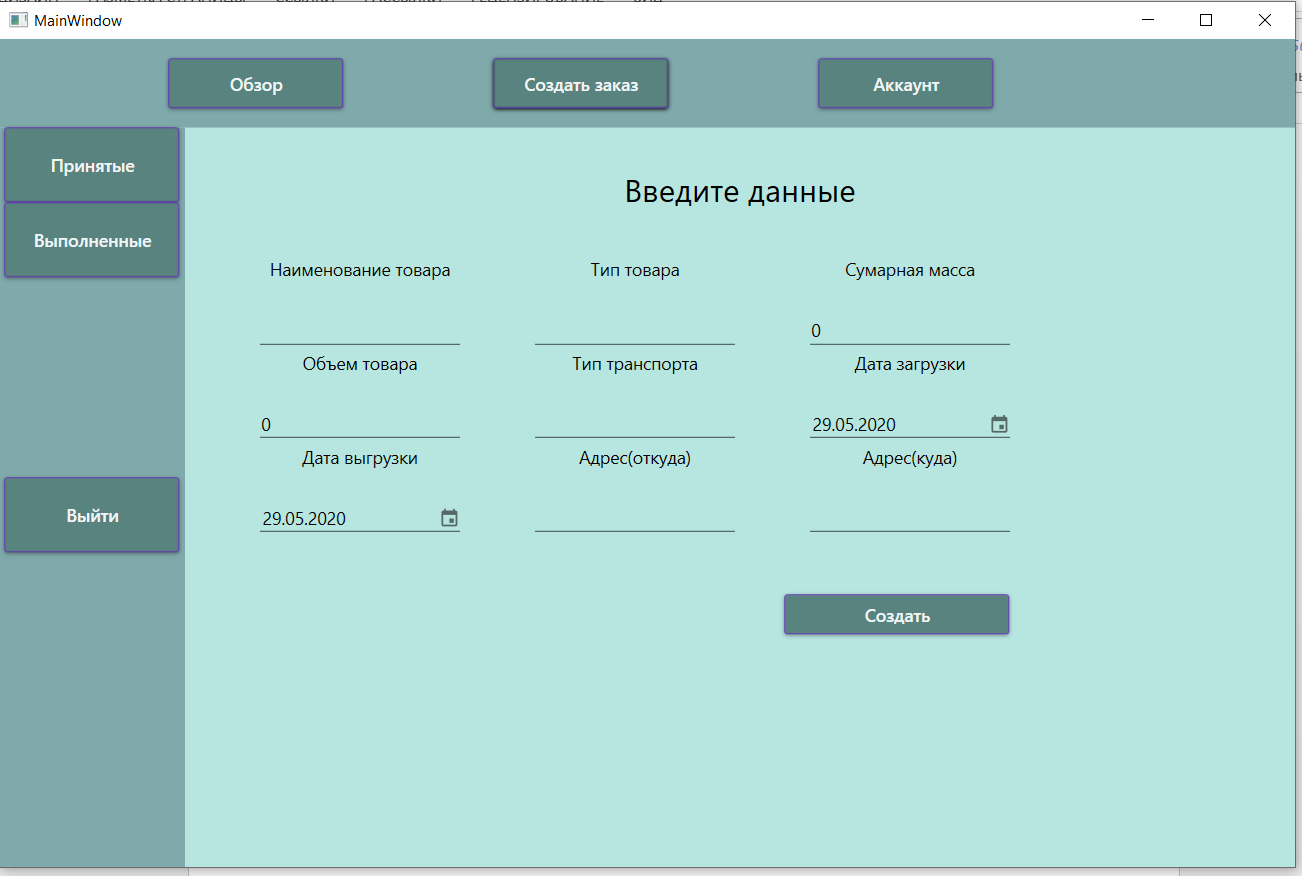


Рисунок 6.4 Страницы «Аккаунт» и «Создать заказ»

# **6.2 Использование приложения администратором**

Администратор авторизовывается по тому же принципу, что и пользователи (Рис. 6.1), однако он попадает на другую страницу(Рис.6.5).



Рисунок 6.5 Главная страница администратора

В отличие от пользователя, администратор просматривает все заказы. К тому же, он может проводить с ними разные манипуляции. Заказ может быть принят, выполнен, либо удален.

**Заключение**

Таким образом, в курсовом проекте реализовано программное средство «Логистическа компания»

В проекте использовались такие средства, как C# .NET Framework, SQL, ADO.NET, WPF, MVVM, Command.

В процессе выполнения работы было создано средство для работы логистической компании, которые выполняет следующие функции:

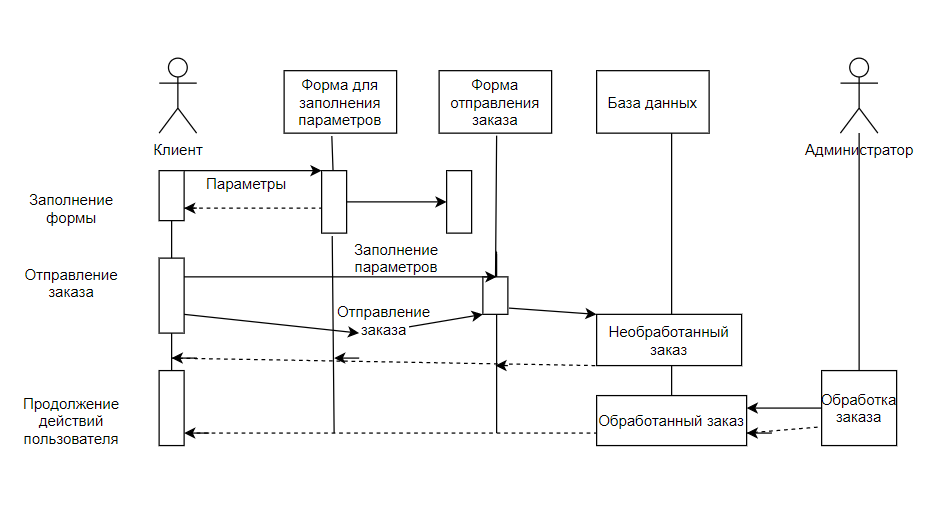
* Авторизация, регистрация
* Просмотр истории заказов, фильтрация, сортировка, а так же поиск заказов
* Возможность создания заказа
* Возможность редактировать личные данные
* Обработка заказов с отправлением уведомлений

В результате была выполнена задача создания программного средтсва «Логистическая компания».

# **Список использованных источников**

1. Пацей, Н.В. Курс лекций по языку программирования С# / Н.В. Пацей. – Минск: БГТУ, 2016. – 175 с.
2. Пацей, Н.В. Технология разработки программного обеспечения / Н.В. Пацей. – Минск: БГТУ, 2016. – 129 с.
3. MSDN сеть разработчиков в Microsoft [Электронный ресурс] / Режим доступа: <http://msdn.microsoft.com/library/rus/>. Дата доступа: 20.04.2018
4. METANIT.COM Сайт о программировании [Электронный ресурс] / Режим доступа: <https://metanit.com>. Дата доступа: 21.05.2018
5. ProfessorWeb .NET & Web Programming [Электронный ресурс] / Режим доступа: <https://professorweb.ru> Дата доступа: 13.04.2018
6. Рихтер Дж. CLR via C#. Программирование на платформе Microsoft .NET Framework 4.5 на языке C#. 4-е изд. — СПб.: Питер, 2013. — 896 с.: ил. — (Серия «Мастер-класс»).

# **Приложение А**

**Диаграмма последовательности** 

**Диаграмма деятельности**

