МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных Технологий

Кафедра Программной инженерии

Специальность 1-40 01 01 Программное обеспечение информационных технологий

Направление специальности 1-40 01 01 10 Программное обеспечение информационных технологий (программирование интернет приложений)

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**КУРСОВОГО ПРОЕКТА:**

по дисциплине «Объектно-ориентированные технологии программирования и стандарты проектирования»

Тема Программное средство «Магазин автозапчастей»

Исполнитель

студент 2 курса группы 4 Шуст Юрий Олегович

(Ф.И.О.)

Руководитель работы ассистент Северинчик Н.А.

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

Курсовой проект защищен с оценкой

Председатель Северинчик Н.А.

(подпись)

Минск 2020

**Содержание**

[**ВВЕДЕНИЕ** 3](#_Toc73065782)

[**1 Аналитический обзор литературы и постановка задачи** 4](#_Toc73065783)

[1.1 Анализ прототипов 4](#_Toc73065784)

[1.2 Постановка задачи 6](#_Toc73065785)

[**2 Анализ требований к программному средству и разработка функциональных требований** 7](#_Toc73065786)

[2.1 Определение требований к программному средству 7](#_Toc73065787)

[2.2 Описание средств разработки 7](#_Toc73065788)

[2.3 Описание функциональности программного средства 8](#_Toc73065789)

[2.4 Спецификация функциональных требований 8](#_Toc73065790)

[**3 Проектирование и создание программного средства** 10](#_Toc73065791)

[3.1 Архитектура системы 10](#_Toc73065792)

[3.2 Диаграммы UML 10](#_Toc73065793)

[3.3 Проектирование логической структуры базы данных 11](#_Toc73065794)

[3.4 Структура проекта 12](#_Toc73065795)

[**4 Реализация программного средства** 14](#_Toc73065796)

[4.1 Реализация MVVM 14](#_Toc73065797)

[4.2 Реализация авторизации и регистрации пользователей 15](#_Toc73065798)

[4.3 Добавление и удаления товара из корзины 18](#_Toc73065799)

[4.4 Отображение оповещений внутри приложения 19](#_Toc73065800)

[4.5 Отправка сообщений на почту и генерация чека 20](#_Toc73065801)

[4.6 Формирование, подтверждение и отмена заказа 21](#_Toc73065802)

[**5 Тестирование** 24](#_Toc73065803)

[**6 Руководство по использованию** 28](#_Toc73065804)

[6.1 Регистрация и авторизация 28](#_Toc73065805)

[6.2 Использование приложения обычным пользователем 29](#_Toc73065806)

[6.3 Использование приложения администратором 29](#_Toc73065807)

[**ЗАКЛЮЧЕНИЕ** 31](#_Toc73065808)

[**СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ** 32](#_Toc73065809)

[Приложение А 33](#_Toc73065810)

[Приложение Б 34](#_Toc73065811)

[Приложение В 35](#_Toc73065812)

[Приложение Г 36](#_Toc73065813)

[Приложение Д 37](#_Toc73065814)

# **ВВЕДЕНИЕ**

Рынок автомобилей с каждым годом растет, количество автомобилей увеличивается, однако каждый автомобиль имеет высокий шанс получить повреждения. В настоящее время автовладельцы для экономии времени предпочитают использовать приложения и онлайн площадки для поиска и покупки автомобильных запчастей. Это так же экономит время, так как многие площадки имеют возможность доставить товар бесплатно.

У многих крупных и не очень организаций уже существует возможность просмотра и приобретения автомобильных запчастей через веб-сайты и приложения. Поэтому темой курсового проекта является разработка программного средства «Магазин автозапчастей». Данное программное средство должно позволить пользователям в лице клиентов заказывать автомобильные запчасти на единой площадке, отслеживать статус заказа. Пользователи в лице администраторов могут добавлять новые товары, просматривать заказы. Так же администраторы имеют возможность удалять товары и поставщиков.

Для успешной реализации курсового проекта необходимо:

* провести анализ соответствующей литературы;
* ознакомиться с прототипами программных средств выбранной мной темы;
* определить функциональные требования;
* продумать структуру базы данных;
* продумать структуру проекта;
* реализовать программное средство;
* протестировать программное средство;
* написать руководство пользователя.

Содержание данной пояснительной записки отражает этапы выполнения курсового проекта.

# **1 Аналитический обзор литературы и постановка задачи**

Для того чтобы окончательно определиться с постановкой задачи курсового проекта, необходимо проанализировать прототипы программных средств выбранной темы.

## 1.1 Анализ прототипов

Exist.by ­– платформа, на которой предоставлен широкий выбор автомобильных запчастей и аксессуаров для автомобиля (рисунок 1.1).

Достоинства:

* большой выбор товаров
* доставка курьерской службой

Недостатки:

* слишком нагруженный интерфейс
* отсутствие собственных пунктов выдачи товара
* для Windows существует только веб-версия

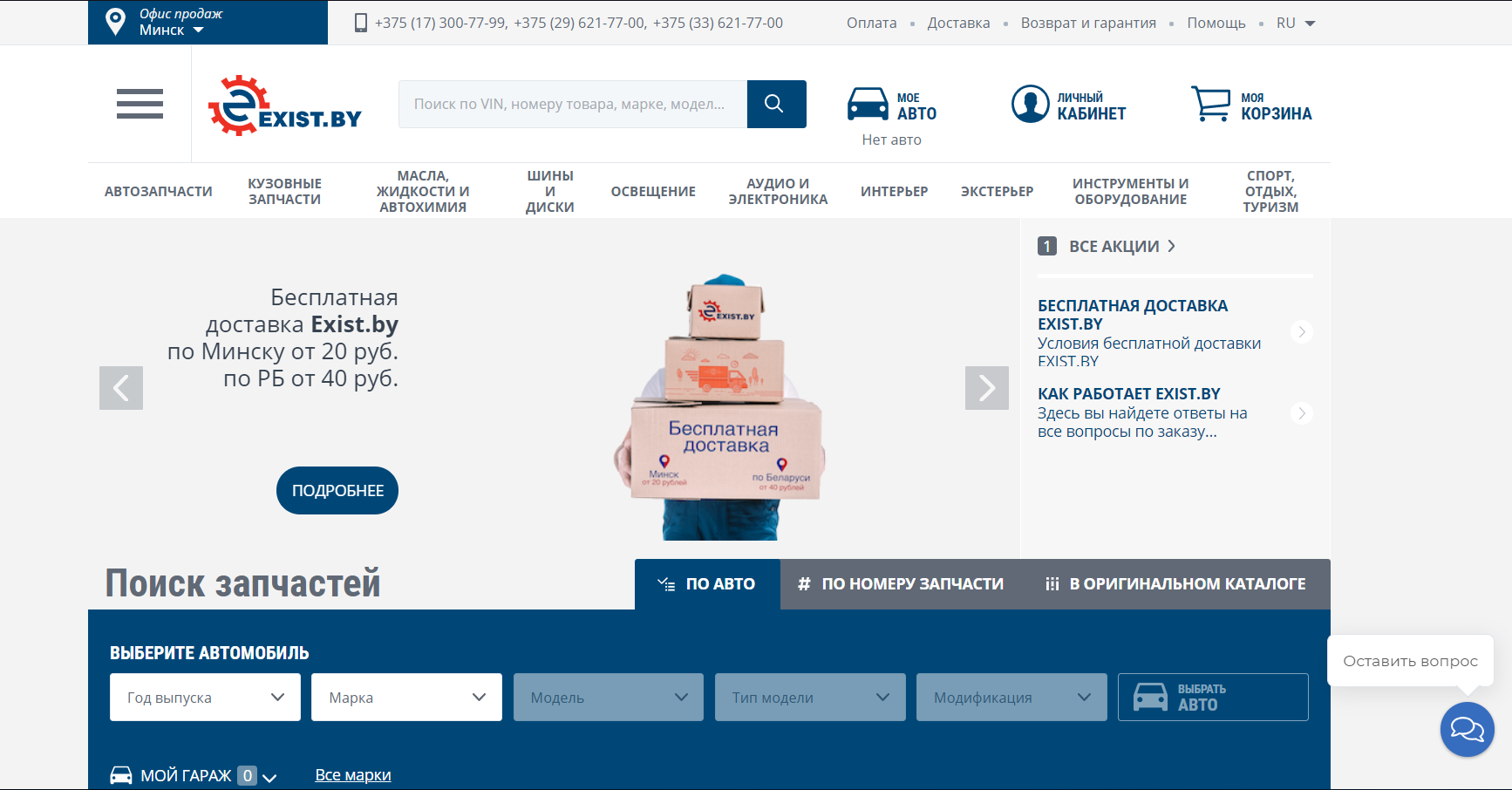


Рисунок 1.1 – Интерфейс веб-приложения «Exist.by»

Zap.by – платформа для заказа автозапчастей (рисунок 1.2).

Достоинства:

* широкий выбор товаров
* бесплатная доставка

Недостатки:

* устаревший интерфейс
* для Windows существует только веб-приложение

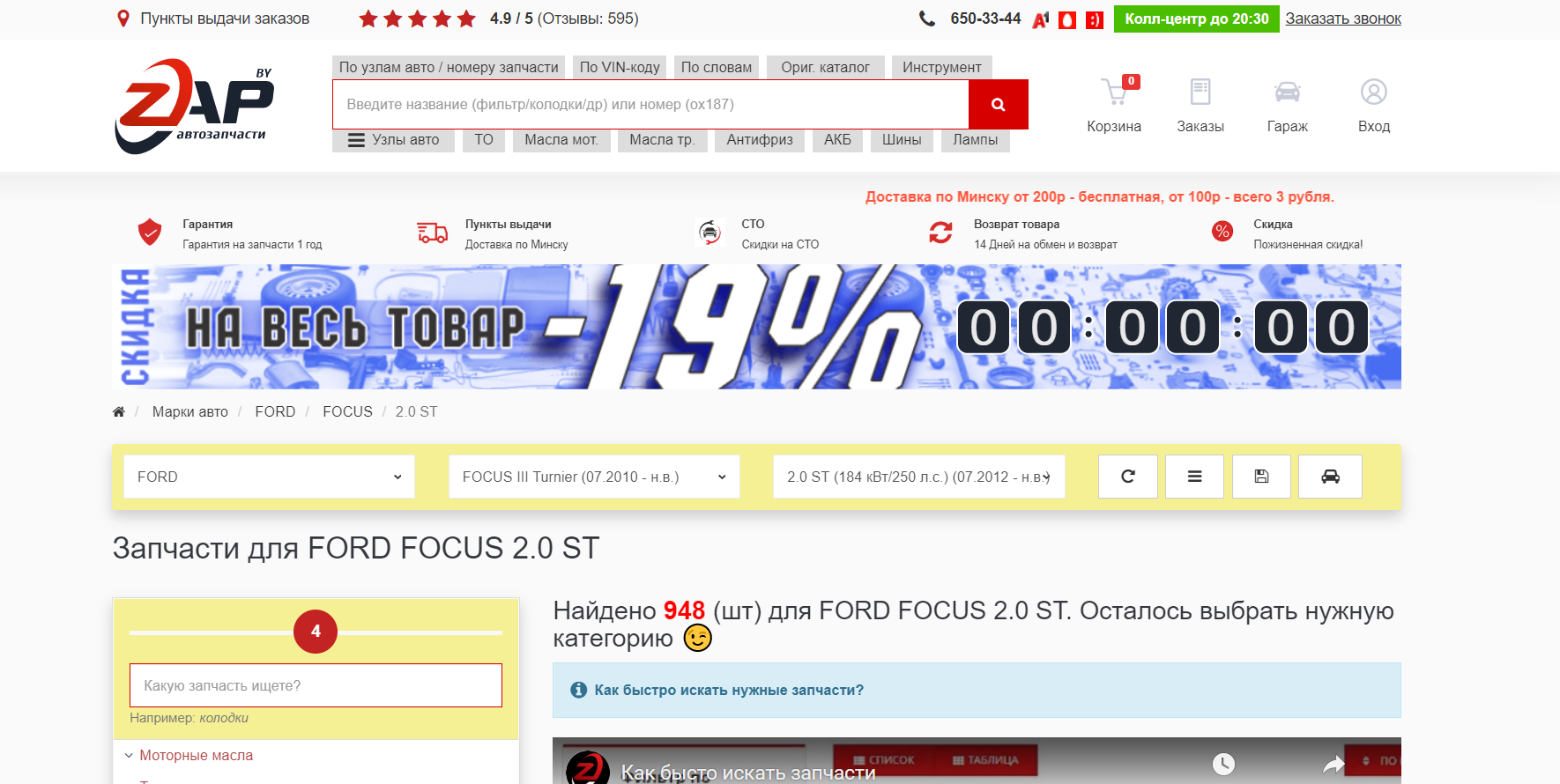


Рисунок 1.2 – Интерфейс веб-приложения «Zap.by»

Auto1.by – сервис предоставляющий широкий выбор автозапчастей и услуги по обслуживанию автомобилей (рисунок 1.3).

Достоинства:

* приятный дизайн
* широкий выбор товаров
* большое количество точек продаж
* наличие рекомендаций по подбору запчастей

Недостатки: не выявлены

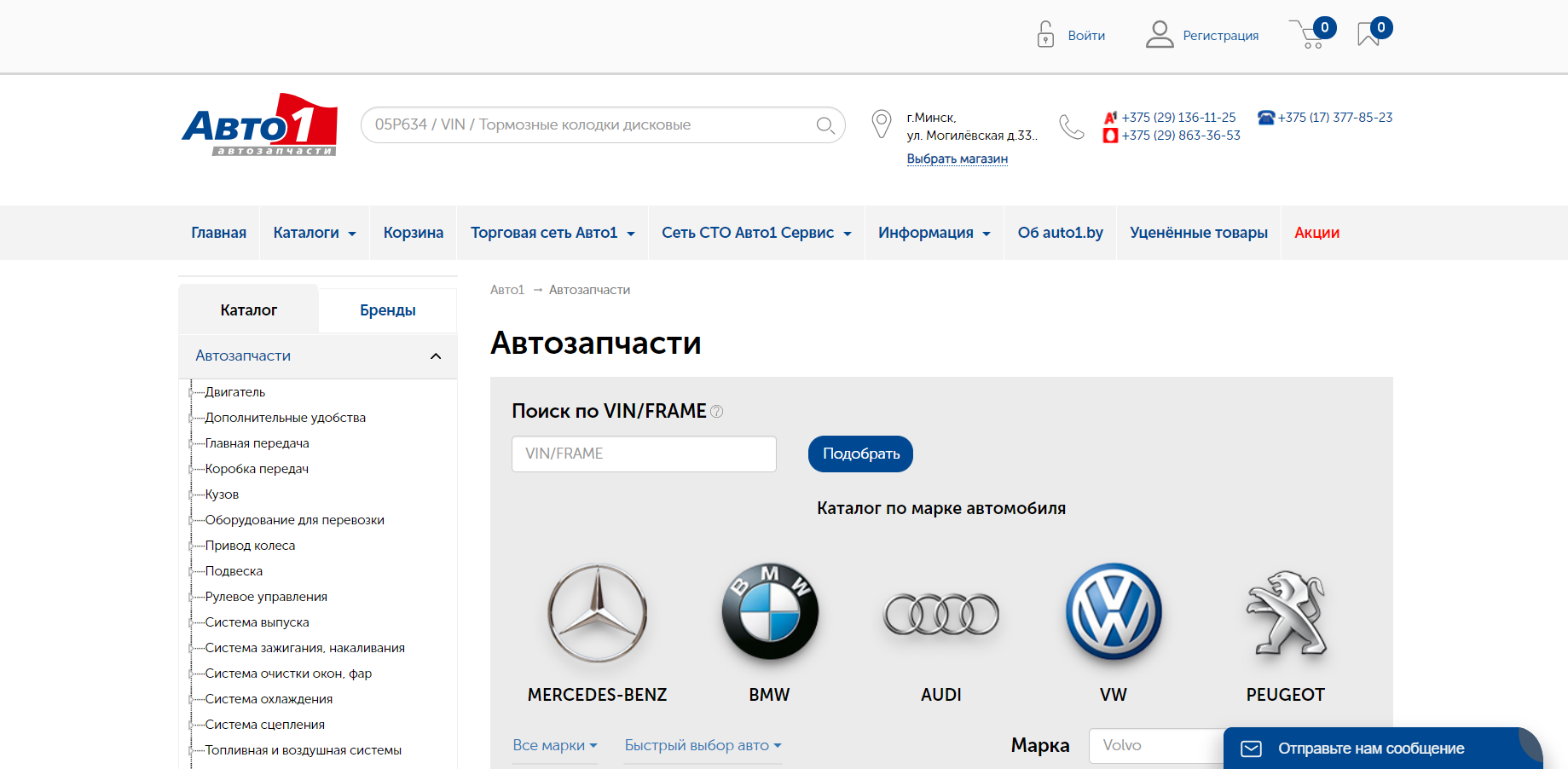


Рисунок 1.3 – Интерфейс веб-приложения «auto1.by»

## 1.2 Постановка задачи

Проанализировав прототипы, были выделены основная задача и функциональные требования проекта.

Основной задачей курсового проекта является разработка десктопного приложения, позволяющего выполнять авторизацию и регистрацию пользователей, пользователю в лице администратора добавлять продукты с информацией о них с возможностью удаления. Пользователю в лице клиента просматривать все продукты, добавлять их в корзину, удалять и оформлять заказ с последующей возможностью отмены.

Функциональные требования описаны в главе 2.

# **2 Анализ требований к программному средству и разработка функциональных требований**

Для того, что бы приступить к этапу разработки проекта, необходимо четко сформулировать функциональные требования к программному средству.

## 2.1 Определение требований к программному средству

Функционально ПС должно выполнять следующие задачи:

* регистрация и авторизация пользователей;
* сохранение рабочей информации в централизованной базе данных;
* для администратора добавление товаров с возможностью их удаления;
* для клиента добавление товаров в корзину с возможностью их удаления;
* для клиента заказ товара из корзины и его подтверждение;
* отправка кода подтверждения заказа посредством отправки электронного письма;
* для администратора просмотр всех товаров и пользователей с возможностью удаления.

## 2.2 Описание средств разработки

При разработке приложения были использованы:

* интегрированная среда разработки Microsoft Visual Studio 2019;
* программная платформа .NET Framework 4.7.2;
* язык программирования C#;
* расширяемый язык разметки XAML;
* технология WPF;
* технология Entity Framework 6.4.4;
* Microsoft SQL Server 2019;

В качестве интерфейса прикладного программирования был выбран обширный API-интерфейс – Windows Presentation Foundation (WPF), предназначенный для создания настольных программ с графически насыщенным пользовательским интерфейсом. В основе графической технологии WPF лежит мощная инфраструктура, основанная на DirectX. Это является одним из основных отличий WPF от более ранней технологии создания пользовательских интерфейсов – Windows Forms.

Для работы с WPF использовался объектно-ориентированный язык программирования с С-подобным синтаксисом – С#, разработанный для создания приложений на платформе Microsoft .NET Framework.

Чтобы осуществлять связь между базой данных и приложением на C# необходим посредник. И именно таким посредником является технология Entity Framework. Она предоставляет собой объектно-ориентированную технологию доступа к данным, является object-relational mapping (ORM) решением для платформы .NET Framework. Entity Framework предоставляет возможность взаимодействия с объектами посредством LINQ to Entities. Центральной концепцией Entity Framework является понятие сущности или entity. Сущность представляет набор данных, ассоциированных с определенным объектом. Поэтому данная технология предполагает работу не с таблицами, а с объектами и их наборами.

## 2.3 Описание функциональности программного средства

Описание функциональности программного средства представлено с помощью UML-диаграммы вариантов использования (Приложение А).

Спецификация функциональности программного средства представлена в следующем пункте.

## 2.4 Спецификация функциональных требований

Для функциональности ПС необходимо создание базы данных для хранения информации приложения. Подробно о базе данных описано в главе 3.

В программном средстве при запуске необходимо реализовать регистрацию и авторизацию пользователей для дальнейшего использования приложения. Для авторизации входными параметрами являются логин и пароль пользователя, которые содержатся в базе данных. Для того, что бы зарегистрироваться необходимо ввести логин, пароль, адрес электронной почты, имя и фамилию. Введенные данные успешно прошедшие валидацию, заносятся в базу данных.

При авторизации пользователя он может на странице «*Главная*» просматривать товары, а так же осуществлять поиск путем ввода имени товара или выбора категории товара. На странице «*Поиск*» пользователь может осуществлять поиск по названию, а так же осуществлять подробный поиск в заданном диапазоне цены и выбранной категории товара. На странице «*Корзина*» пользователь может просматривать выбранные им товары для заказа, а так же при желании удалять их из корзины. На странице «*Настройки*» пользователь моет выбирать язык интерфейса. На странице «О нас» пользователь может найти ответы на часто задаваемые вопросы и контакты для обратной связи. На странице «*Профиль*» пользователь может просмотреть информацию о своем аккаунте, просмотреть историю заказов, привязать карту или пополнить баланс счета.

При авторизации администратора на странице «*Пользователи*» он может просматривать список всех пользователей, а так же изменять их роль или удалять. На странице «*Заказы*» администратор может просматривать список всех заказов. На странице «*Поставщики*» администратор может просматривать список всех поставщиков, добавлять и удалять поставщиков. На странице «*Товары*» администратор может просматривать все товары, добавлять и удалять товары. На странице «*Категории*» администратор может просматривать все категории, добавлять и удалять категории.

Кнопка для выхода их приложения должна возвращать пользователя к окну авторизации.

Все данные о товарах, пользователях и заказах должны заноситься в базу данных в соответствующие таблицы.

# **3 Проектирование и создание программного средства**

Проектирование программного средства — процесс создания проекта программного обеспечения. Целью проектирования является определение внутренних свойств системы и детализации её внешних свойств на основе исходных условий задачи. Исходные условия задачи уже были сформулированы во втором разделе данной пояснительной записки. Этап проектирования подразумевает их анализ.

## 3.1 Архитектура системы

В данном приложении используется архитектурный паттерн MVVM (MODEL-VIEW-VIEWMODEL). Он позволяет отделить логику приложения от визуальной части. MVVM состоит из трех частей:

* Модель описывает используемые в приложении данные.
* Представление определяет визуальный интерфейс, через который пользователь взаимодействует с приложением.
* Модель представления связывает модель и представление через механизм привязки данных. ViewModel также содержит логику по получению данных из модели, которые потом передаются в представление.

На рисунке 3.1 представлена диаграмма, которая показывает общую структуру приложения в рамках шаблона MVVM.



Рисунок 3.1 – Структура шаблона MVVM

## 3.2 Диаграммы UML

UML — унифицированный язык моделирования (Unified Modeling Language) — это система обозначений, которую можно применять для объектно-ориентированного анализа и проектирования.

Его можно использовать для визуализации, спецификации, конструирова­ния и документирования программных систем.

Диаграмма — это графическое представление набора элементов, чаще всего изображенного в виде связного графа вершин (сущностей) и путей (связей).

Диаграммы последовательностей используются для уточнения диаграмм прецедентов, более детального описания логики сценариев использования. Это отличное средство документирования проекта с точки зрения сценариев использования.

Диаграмма последовательности взаимодействия с окном авторизации приведена в приложении Б.

Диаграмма деятельности, как и диаграмма состояний, отражает динамические аспекты поведения системы. По существу, эта диаграмма представляет собой блок-схему, которая наглядно показывает, как поток управления переходит от одной деятельности к другой.

Диаграммы деятельности используются при моделировании бизнес-процессов, технологических процессов, последовательных и параллельных вычислений.

Часть диаграммы деятельности данного приложения приведена в приложении В, данная диаграмма приведена для части программы связанной со входом пользователя в приложение. Она разбита на 2 части: авторизация и регистрация, для каждой из этих частей приведен свой алгоритм, осуществляющий все проверки и этапы обработки информации необходимые для корректной работы данной части приложения.

Диаграммы использования показывает актеров и возможные прецеденты, при работе с приложением. Диаграмма использования приведена в приложении А.

Диаграмма классов служит для представления статической структуры мо­дели системы в терминологии классов объектно-ориентированного программирования.

Диаграмма классов ViewModel приведена в приложении Г

## 3.3 Проектирование логической структуры базы данных

Для реализации поставленной задачи была создана база данных. Для её создания использовались система управления реляционными базами данных Microsoft SQL Server 2019.

База данных – совокупность данных, организованных по определенным правилам, предусматривающим общие принципы описания, хранения и манипулирования данными, независимо от прикладных программ.

Система управления базами данных – совокупность программ и языковых средств, предназначенных для управления данными в базе данных, ведения базы данных и обеспечения взаимодействия ее с прикладными программами. База данных нашей программной подсистемы состоит из 9 таблиц, ее схема изображена в приложении Д.

Таблица Users используется для хранения данных о пользователях. Включает 7 столбцов: Id– id пользователя, Login – логин пользователя, Password – пароль пользователя, при записи в базу данных хэшируется, Is\_admin – 0 – пользователь, 1 - администратор, FirstName –имя пользователя, LastName – фамилия пользователя, Email – адрес электронной почты.

Таблица Providers используется для хранения данных о поставщиках товара. Включает 3 столбца: ProviderId – id поставщика, Name – имя поставщика, Email – адрес электронной почты поставщика.

Таблица Parts используется для хранения продуктов. Включает 10 столбцов: PartId – id продукта, Name – название продукта, Quantity – количество товара, доступного для покупки, ProviderId – id поставщика продукт, Price – цена покупки товара, Description – краткое описание товара, FullDescription – полное описание товара, ImageLink – ссылка на изображение товара, CategoryId – id категории товара, MarkId – id марки автомобиля, к которому принадлежит товар.

Таблица Orders используется для хранения информации о заказах. Включает 5 столбцов: OrderId – id заказа, OrderDate – дата открытия заказа, UserId – id пользователя, который сделал заказ, DeliveryId – id службы доставки, OrderState – статус заказа.

Таблица OrderedParts используется для связи между таблицами Orders и Parts. Используется для определения того, какие товары и какое их количество входит в заказ. Состоит из 3 столбцов: PartId – id товара, OrderId – id заказа, Amount – количество единиц определенного товара в заказе.

Таблица Marks используется для хранения данных о марках автомобилей. Включает 2 столбца: MarkId – id марки автомобиля, MarkName – наименование марки автомобиля.

Таблица Deliveries используется для хранения информации о службах доставки. Состоит из 4 столбцов: DeliveryId – id службы доставки, Description – описание службы доставки, Price – цена услуги доставки, Name – наименование службы доставки.

Таблица Categories хранит информацию о категориях товаров. Состоит из 3 столбцов: CategoryId – id категории, Name – имя категории, Description – описание категории.

Таблица Cards используется для хранения данных о привязанной карте для оплаты. Состоит из 5 столбцов: CardId – id карты, CardNumber – номер карты, UserId – id пользователя, CvvCode – код cvv карты, Balance – баланс карты.

## 3.4 Структура проекта

Структура проекта представлена на рисунке 3.2.

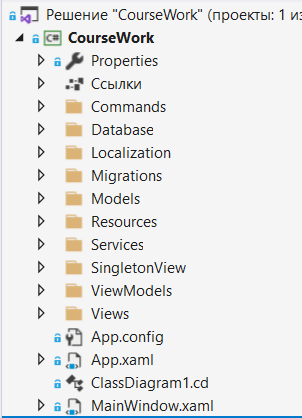


Рисунок 3.2 – структрура проекта

Папка «Database» содержит классы для работы с базой данных.

Папка «Models» содержит классы моделей.

Папка «Localization» содержит класс для работы с локализацией приложения.

Папка «Resources» содержит стили и шаблоны элементов управления, а так же каталог с «Images» в котором содержатся все необходимые графические элементы.

Папка «Services» содержит классы для поддержки некоторых функций приложения.

Папка «ViewModels» содержит классы моделей представлений.

Папка «Views» содержит все представления. Приложение включает в себя 5 окон и 19 пользовательских элементов управления, выступающих в качестве страниц. Связи между ними представлены на рисунке 3.3.

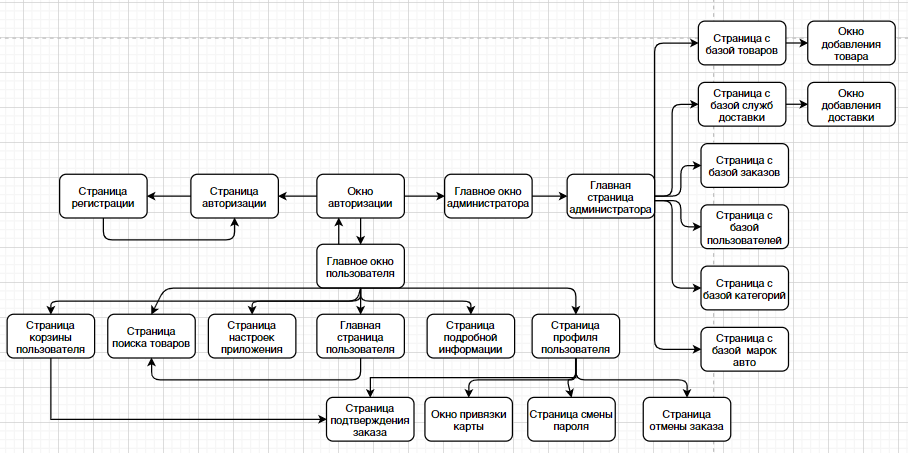


Рисунок 3.3 – связи между окнами

# **4 Реализация программного средства**

## 4.1 Реализация MVVM

Паттерн MVVM реализуется через базовый класс ViewModelBase, который реализует интерфейс INotifyPropertyChanged и от которого наследуются все страницы. Для реализации паттерна файлы программы были распределены по соответствующим пространствам имен (см. 3.2) и реализованы следующие функции. На рисунке 4.1 видим, что класс MainViewModel наследует класс ViewModelBase (рис. 4.2) .

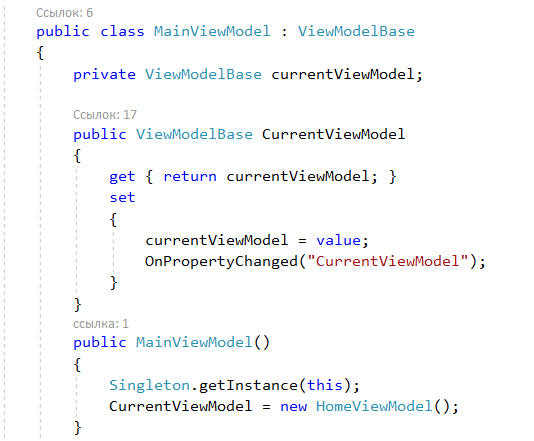


Рисунок 4.1 – пример использования ViewModelBase

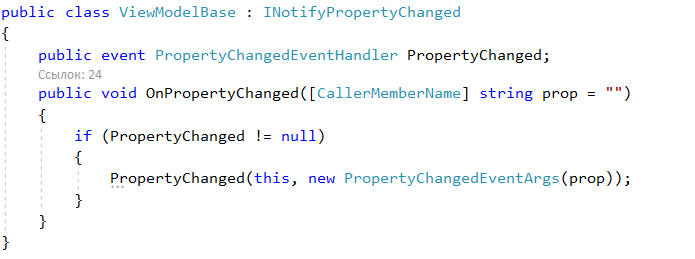


Рисунок 4.2 – структура класса ViewModelBase

В приложении используется паттерн Command который позволяет инкапсулировать запрос на выполнение определенного действия в виде отдельного объекта. В WPF команды представлены интерфейсом ICommand. В приложении он представлен в виде собственной команды Command код которой приведен в приложении Е. Класс реализует два метода:

* CanExecute: определяет, может ли команда выполняться
* Execute: собственно, выполняет логику команды

## 4.2 Реализация авторизации и регистрации пользователей

Для того чтобы пользоваться приложением каждый пользователь должен войти в систему, предварительно зарегистрировавшись.

На рисунке 4.3 находится код, реализующий регистрацию.



Рисунок 4.3 – Команда, реализующая регистрацию

Для безопас­ности хранения информации используется хэширование, в данном случае применяется встроенный класс MD5. Код метода хэширования приведен на рисунке 4.4.

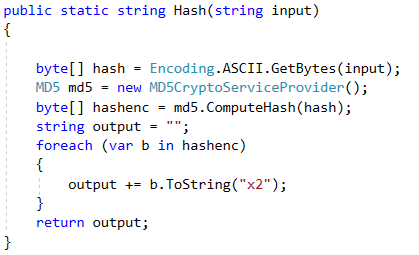


Рисунок 4.4 – Пример метода обеспечивающего хэширование пароля

При регистрации пользователю задается тип аккаунта user. Если все данные были введены верно и логин пользователя, почта еще не заняты, то осуществляется регистрация и переход на страницу авторизации.

Данные при регистрации проходят валидацию. Валидация реализована с использованием регулярных выражений в аннотациях к полям модели пользователя. Пример кода реализующего валидацию приведен на рисунке 4.5.

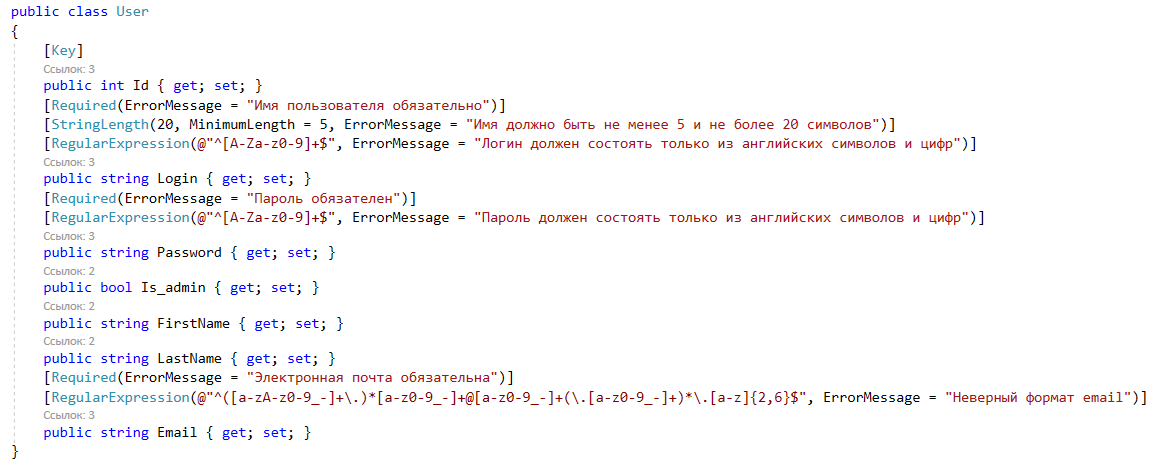


Рисунок 4.5 – Пример валидации с использованием аннотаций

При авторизации проверяется существует ли пользователь с такими данными, а затем проверяется значения хэш-функций паролей. В случае неудовлетворения требований появляется надпись с ошибкой. В ином случае создается окно обычного пользователя или администратора в зависимости от значения в поле is\_admin в базе данных. Пример команды, реализующей авторизацию приведен на рисунке 4.6.



Рисунок 4.6 – Команда, реализующая авторизацию

## 4.3 Добавление и удаления товара из корзины

Добавление товара в корзину пользователя производится с помощью команды привязанной на странице полной информации о товаре FullInfoView. Пример команды, реализующей добавление товара в корзину пользователя приведен на рисунке 4.7.

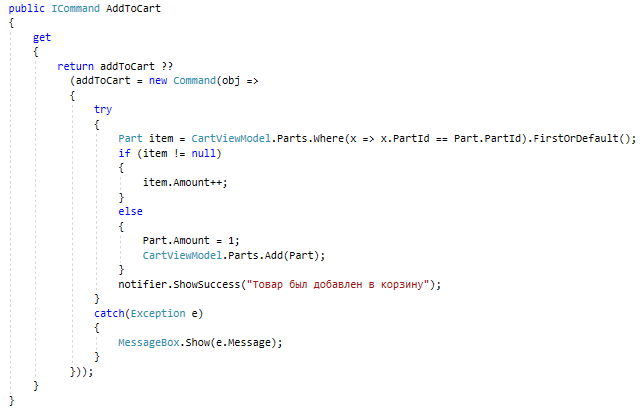


Рисунок 4.7 – Команда, реализующая добавление товара к корзину

Удаление товара из корзины пользователя производится с помощью команды привязанной на странице корзины пользователя. Пример команды, реализующей удаление приведен на рисунке 4.8.



Рисунок 4.8 – Команда, реализующая удаление товара из корзины

## 4.4 Отображение оповещений внутри приложения

Для отображения оповещений внутри приложения используется библиотека ToastNotifications, которая предоставляет класс Notification и методы для отображения оповещений различного типа. В приложении в классе App реализован статический метод, который возвращает объект типа Notifier. Пример метода приведен на рисунке 4.9.

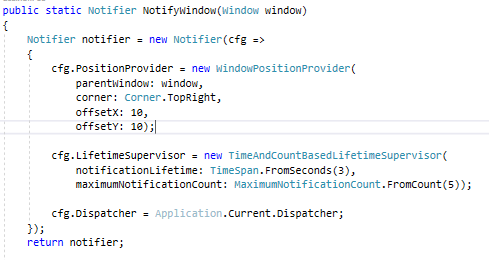


Рисунок 4.9 – Метод предоставляющий объект уведомлений

В объекте уведомлений задаются такие параметры как:

* окно родитель;
* позиция отображения уведомления;
* отступы от краев окна, в котором отображается уведомление;
* время существования уведомления;
* максимальное количество отображаемых уведомлений;

## 4.5 Отправка сообщений на почту и генерация чека

Для отправки сообщений на электронную почту пользователей используются классы, предоставленные пространством имен System.Net.Mail. Данное пространство имен предоставляет классы для работы с SMTP протоколом.

В качестве параметров в первом случае передается адрес электронной почты пользователя, код, тема и тело сообщения, а во втором – адрес электронной почты, тема и тело сообщения.

Примеры отправки кода подтверждения и чека заказа приведены на рисунках 4.10 и 4.11.



Рисунок 4.10 – Метод асинхронной отправки кода подтверждения



Рисунок 4.11 – Метод асинхронной отправки чека заказа

Для создания чека используется статический метод GenerateTicket, в котором используется объединение нескольких таблиц для получение наиболее детальной информации о заказе и товарах в нем. В качестве параметра в метод передается объект заказа. Реализация метода генерации чека приведена на рисунке 4.12.

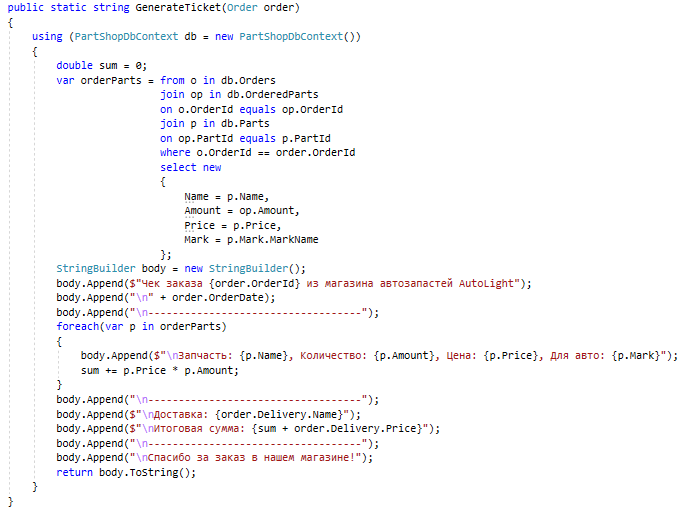


Рисунок 4.12 – Метод генерации чека заказа

## 4.6 Формирование, подтверждение и отмена заказа

Заказ формируется с помощью метода AddOrder в который передается коллекция объектов из корзины. В данном методе так же проводится проверка на наличие средств на счете, наличие выбранной доставки и наличие достаточного количества товаров на складе для оформления заказа.

Так как таблицы Order и Parts связаны как один-ко-многим, то в методе формируется новая коллекция объектов промежуточной таблицы, которая прикрепляется к заказу.

Пример метода, реализующего формирование заказа приведен на рисунке 4.13.

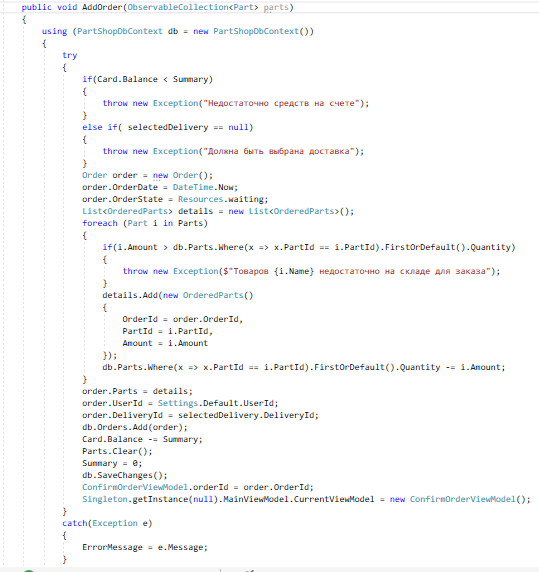


Рисунок 4.13 – Метод, реализующий формирование заказа

Подтверждение заказа осуществляется с помощью команды SubmitCode, которая получает код из поля представления и сравнивает его с кодом, сгенерированным приложением.

Пример команды подтверждения заказа приведен на рисунке 4.14.



Рисунок 4.14 – Команда подтверждения заказа

Отмена заказа осуществляется командой CancelOrder, которая получает код из поля представления и сравнивает его с кодом, сгенерированным приложением. В ходе работы заказ получает метку отмены, товары возвращаются в базу данных, а деньги возвращаются на счет пользователя.

Пример кода команды приведен на рисунке 4.15.



# **5 Тестирование**

В курсовом проекте задействуется обработка ошибок, таким образом, что пользователь будет уведомлен о неудачном выполнении запроса к базе данных, , или недоступности данных в формах. Присутствуют различные элементы, всплывающие окна уведомляющие пользователя.

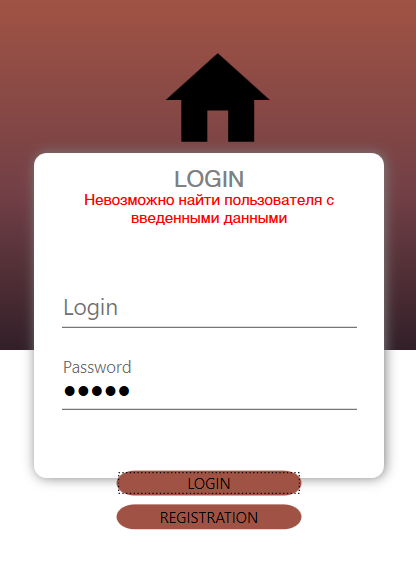


Рисунок 5.1 – отсутствие введенного логина

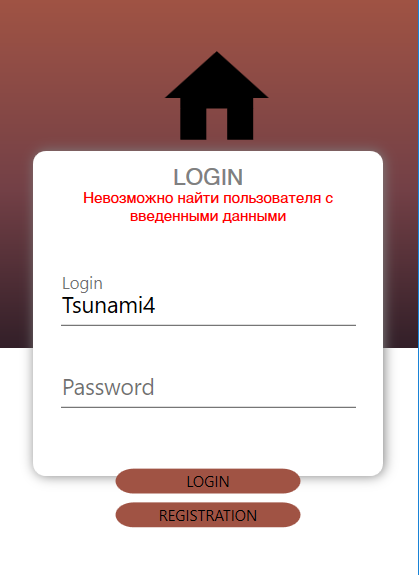


Рисунок 5.2 – отсутствие введенного пароля

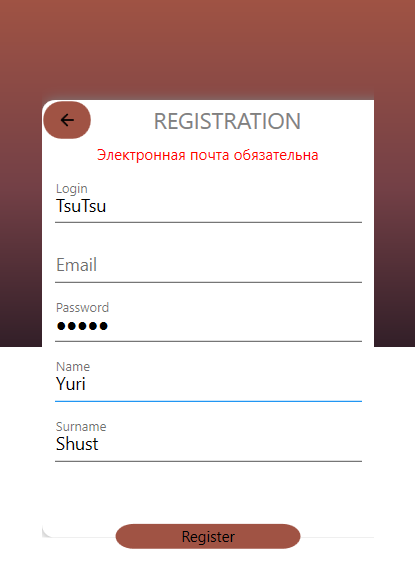


Рисунок 5.3 – отсутствие введенного адреса электронной почты

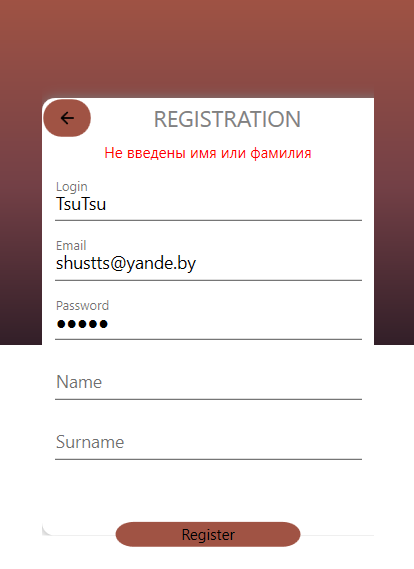


Рисунок 5.4 – отсутствие введенного имени или фамилии

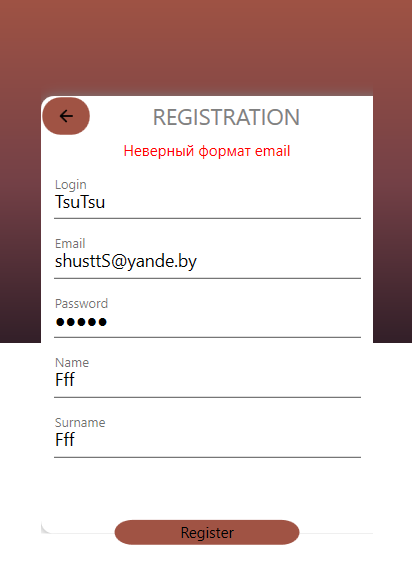


Рисунок 5.5 – неверный формат адреса электронной почты

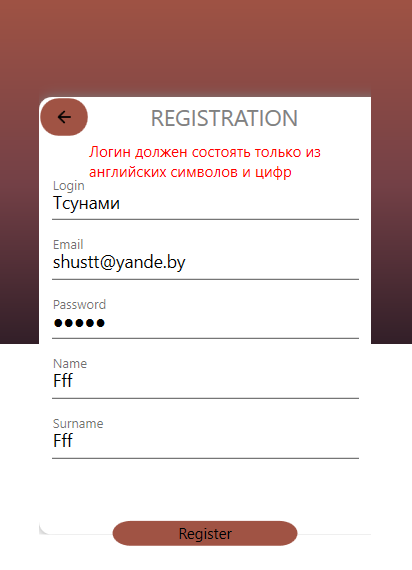


Рисунок 5.6 – логин состоящий из кириллицы

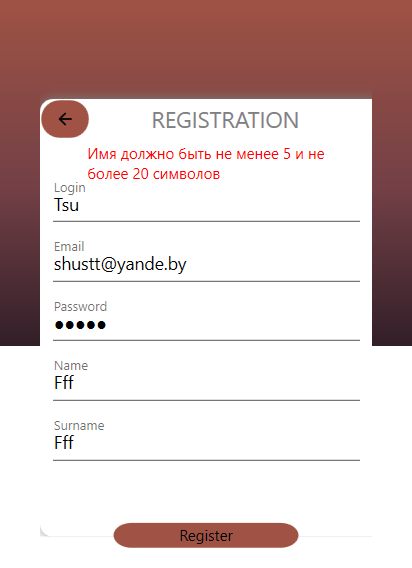


Рисунок 5.7 – логин, состоящий из менее, чем 5 символов

# **6 Руководство по использованию**

## 6.1 Регистрация и авторизация

После запуска приложения пользователь видит главное окно приложения, на котором он может авторизоваться и получить возможность приобретать товары. Окно представлено на рисунке 6.1.

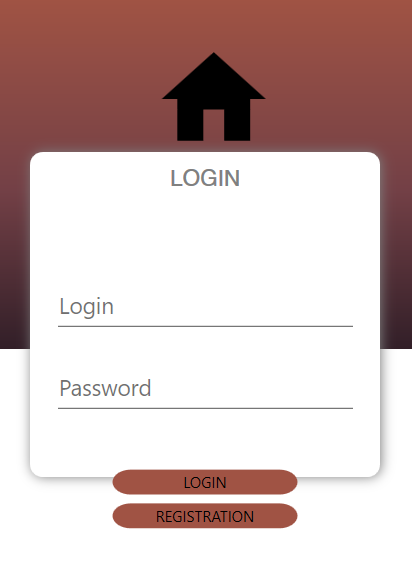


Рисунок 6.1 – Окно входа в аккаунт

Если пользователь ещё не имеет своего аккаунта, то по нажатию на соответствующую кнопку можно перейти к окну регистрации. Пользователь должен придумать логин и пароль, которые будет использовать для авторизации в приложении. Окно регистрации показано на рисунке 6.2.

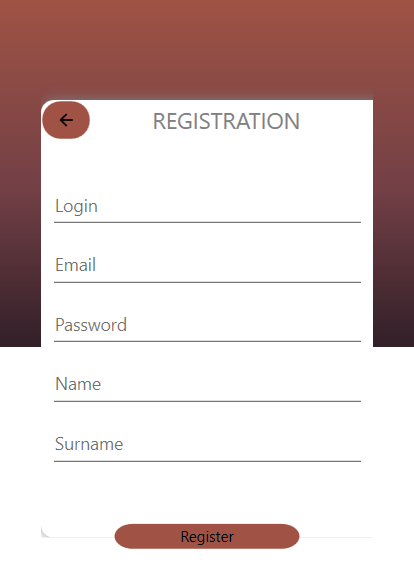


Рисунок 6.2 – Окно регистрации

## 6.2 Использование приложения обычным пользователем

После успешной авторизации пользователь попадает на главный экран, где находятся все товары магазина, а так же предоставлен поиск по названию товара и по выбранной категории. Так же на главном окне имеется 7 кнопок для навигации по страницам и выхода из приложения.

Стартовое окно пользователя приведено на рисунке 6.3.

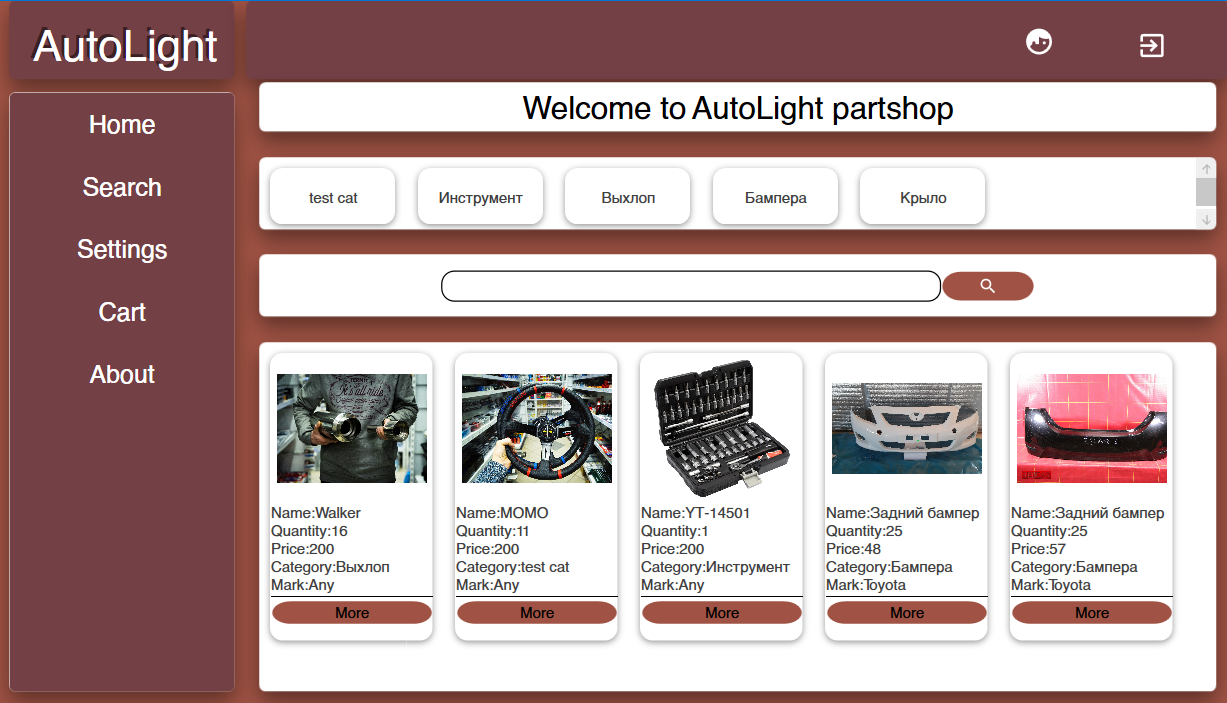


Рисунок 6.3 – Главное окно пользователя

Далее пользователь имеет возможность просматривать и добавлять товары в корзину, а так же оформлять заказ.

## 6.3 Использование приложения администратором

После успешной авторизации администратор попадает в главное окно администратора, где видит 10 для навигации по страницам и выхода их приложения. Администратор имеет возможность добавлять такие объекты как:

* товары
* доставки
* категории
* марки

Так же администратор имеет возможность вносить в изменения любых объектов находящихся в базе данных.

Пример главного окна администратора приведен на рисунке 6.4.

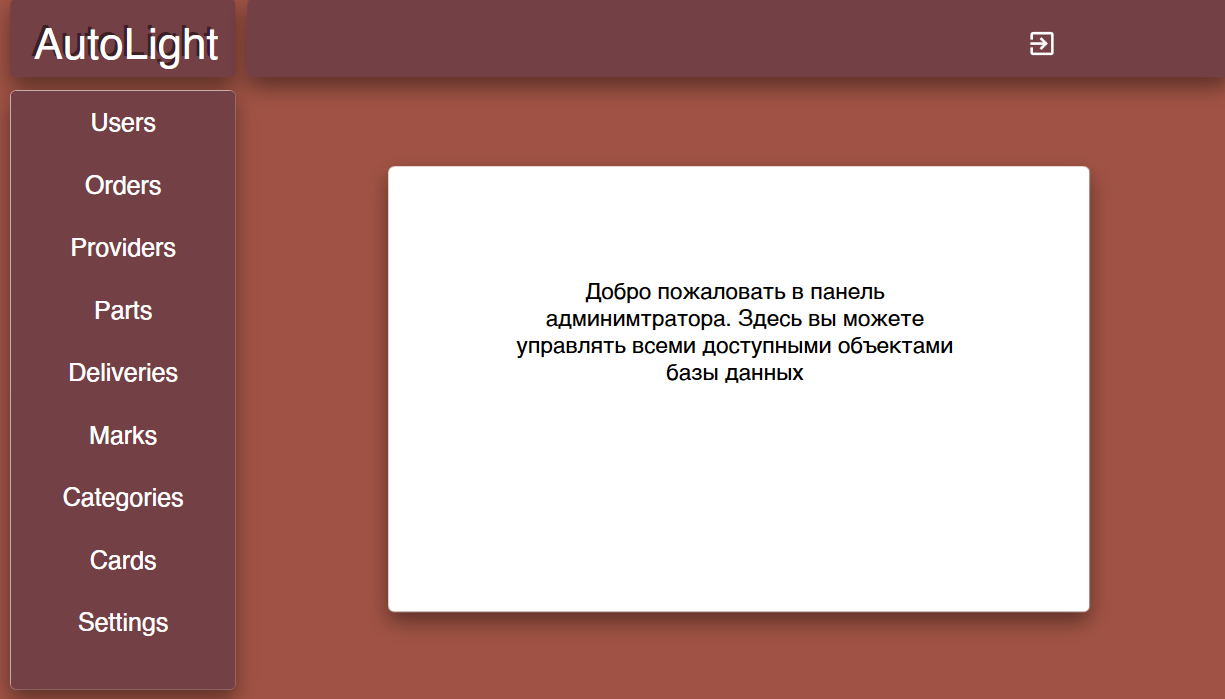


Рисунок 6.4 – Главное окно администратора

# **ЗАКЛЮЧЕНИЕ**

В итоге выполнения курсового проекта было разработано программное средство «Магазин автозапчастей», предоставляющее возможность добавлять товары в корзину, а так же косвенно покупать товары, находящиеся в данной корзине. Так же приложение поддерживает отправку кода подтверждения по электронной почте и смену языка. Основные достоинства – простота использования, оригинальный дизайн и гибкий функционал.

В процессе реализации программного средства были достигнуты выше указанные требования и задачи.

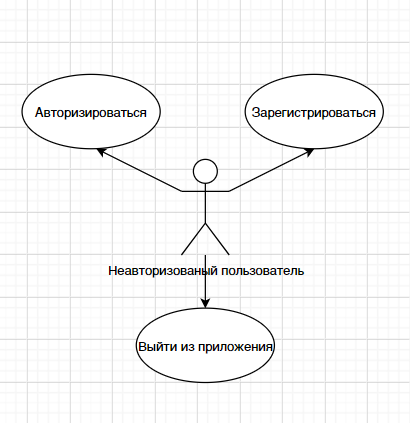
Также в процессе выполнения данного курсового проекта были закреплены навыки в программировании на языке C#, создании приложений на WPF, использование Entity Framework 6, работа с современным паттерном MVVM, проектирование базы данных и реализация их в СУБД MS SQL Server 2019.

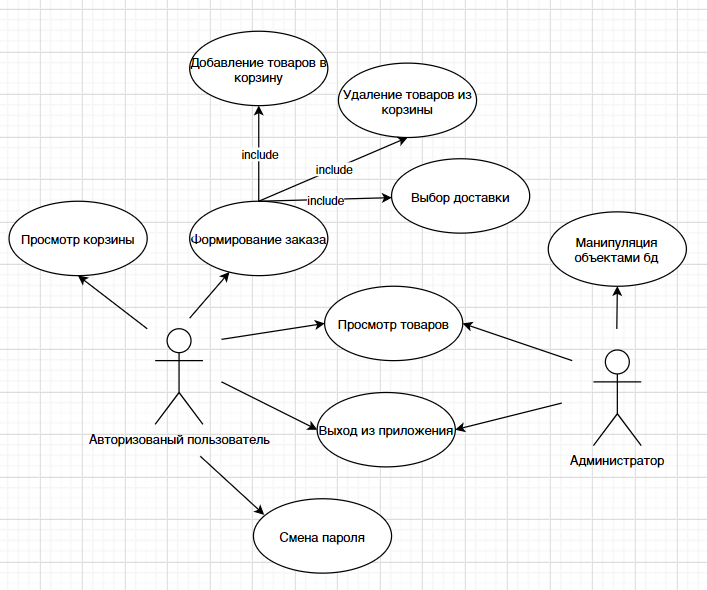
Был получен опыт взаимодействия со сторонними библиотеками, изучения их и внедрения в свой проект.

**СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ**

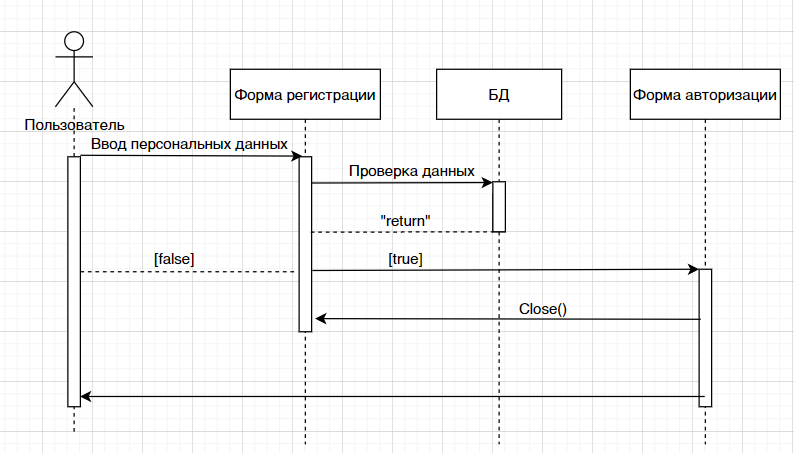
1. MSDN сеть разработчиков в Microsoft [Электронный ресурс] – <http://msdn.microsoft.com/library/rus/>. Дата доступа: 10.05.2021
2. WPF | Определение паттерна MVVM [Электронный ресурс] – <https://metanit.com/sharp/wpf/22.1.php>. Дата доступа: 24.03.2021
3. Пацей, Н.В. Курс лекций по языку программирования С# / Н.В. Пацей. – Минск: БГТУ, 2016. – 175 с.
4. Entity Framework 6 [Электронный ресурс] – <https://metanit.com/sharp/entityframework/3.13.php>. Дата доступа: 05.04.2021
5. Сайт о программировании: MVVM [Электронный ресурс] / medium.com – Режим доступа: <https://medium.com/@losiennik/mvvm-architectural-pattern-on-ios-using-swift-a7b039e3101>. Дата доступа: 24.03.2021.

## Приложение А

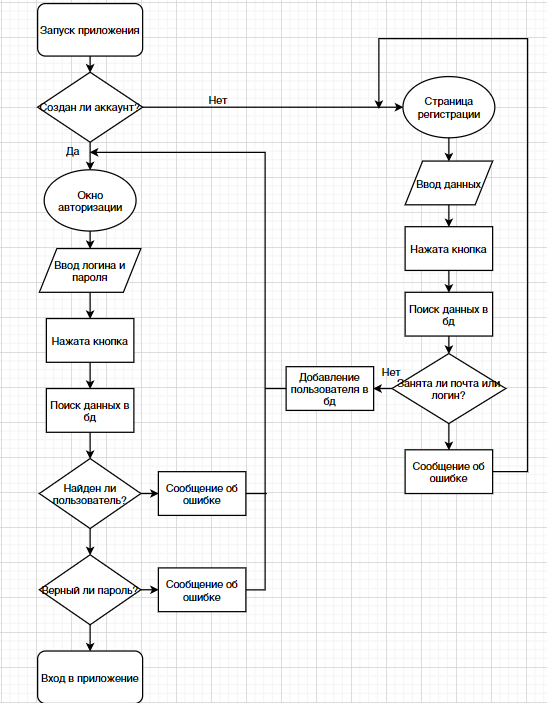




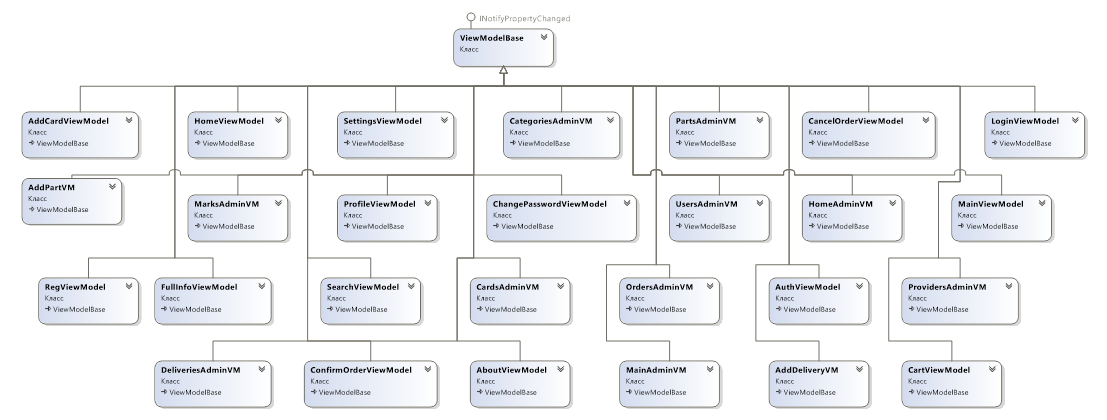
## Приложение Б



## Приложение В



## Приложение Г



## Приложение Д

