Министерство цифрового развития государственного управления, информационных технологий и связи Республики Татарстан

Государственное автономное профессиональное образовательное учреждение

«Международный центр компетенций - Казанский техникум

информационных технологий и связи»

|  |  |
| --- | --- |
| Курсовой проект выполнен и защищен с оценкой \_\_\_\_\_\_\_\_\_\_\_  Руководитель КП \_\_\_\_\_\_\_\_\_\_  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_20\_\_\_\_ г |  |

КУРСОВОЙ ПРОЕКТ

по МДК.02.01 Технология разработки программного обеспечения

на тему: Разработка автоматизированной информационной системы для онлайн - покупок автозапчастей

|  |  |  |
| --- | --- | --- |
| Руководитель | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  *подпись, дата* | И.И. Долгова |
| Исполнитель,  студент группы 420 ИСиП(П) | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  *подпись, дата* | А.Ф. Шигапов |

Казань, 2021

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc87533109)

[1. РАЗРАБОТКА СИСТЕМНОГО ПРОЕКТА 4](#_Toc87533110)

[1.1 Анализ предметной области 4](#_Toc87533111)

[1.2 Обоснование выбора языка программирования, среды программирования и СУБД 5](#_Toc87533112)

[2 ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ПРОДУКТА 9](#_Toc87533113)

[2.1 Графическое представление программы 9](#_Toc87533114)

[2.2 UML диаграмма прецедентов (use case diagram) 10](#_Toc87533115)

[2.3 Диаграмма ERD 11](#_Toc87533116)

[2.4 Диаграмма последовательности 12](#_Toc87533117)

[2.5 Диаграмма IDEF 13](#_Toc87533118)

[3. РЕАЛИЗАЦИЯ ПРОЕКТА 17](#_Toc87533119)

[3.1 Создание базы данных 17](#_Toc87533120)

[3.2 Создание приложения 20](#_Toc87533121)

[4. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ 26](#_Toc87533122)

[4.1 Запуск программы 26](#_Toc87533123)

[4.2 Работа с программой 26](#_Toc87533124)

[5. ТЕСТИРОВАНИЕ 32](#_Toc87533125)

[5.1 Виды тестирования 32](#_Toc87533126)

[5.2 Тест-кейсы 33](#_Toc87533127)

[6. СОЗДАНИЕ УСТАНОВЩИКА ПРОГРАММЫ 36](#_Toc87533128)

[ЗАКЛЮЧЕНИЕ 37](#_Toc87533129)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 38](#_Toc87533130)

[ПРИЛОЖЕНИЕ A 39](#_Toc87533131)

[ПРИЛОЖЕНИЕ B 49](#_Toc87533132)

ВВЕДЕНИЕ

В современном постоянно развивающемся мире постоянно растёт количество личных автомобилей. Каждый человек имеющий свои собственный автомобиль стремится достичь максимального комфорта в его использовании посредством установки дополнительного оборудования. А чтобы автомобиль как можно дольше сохранял свои технические качественные показатели и хороший внешний вид необходимо следить за его состоянием. Для того чтобы быть уверенным в своём автомобиле, нужно вовремя менять изношенные запчасти и расходники.

Это действительно актуальная проблема так как в современных условиях ремонта автомобилей возникает потребность в удобном приложении для быстрых покупок автокомпонентов не требующей специальных знаний, не выходя из дома.

Сферой применения информационной системы являются онлайн магазины осуществляющие продажи автозапчастей с помощью сети Интернет. Информационная система позволит хранить большое количество сведений в одной базе данных, экономить рабочее время за счет автоматизации процессов покупки запчастей.

Цель работы разработать программный продукт, предназначенный для автоматизации процесса учёта продаж автозапчастей, который включает в себя все необходимые средства для хранения имеющихся в наличии и проданных запчастей. Разрабатываемый программный продукт должен осуществлять функции фильтрации и поиска имеющихся в наличии автозапчастей.

# РАЗРАБОТКА СИСТЕМНОГО ПРОЕКТА

## Анализ предметной области

Предметной области разрабатываемой программы является интернет-магазин автозапчастей, осуществляющий онлайн продажи товаров с помощью клиентского приложения.

В разрабатываемом проекте выделяются следующие сущности:

1. Brand – хранит список брендов;

2. Category – хранит список категорий товаров;

3. Subcategory – хранит информацию о подкатегориях;

4. Unit – хранит список единиц измерения;

5. User – хранит информацию о клиентах;

6. DataAuthorization – хранит информацию о данных для авторизации клиента;

7. Good – хранит список товаров;

8. GoodSale – хранит информацию о проданном товаре;

9. Picture – хранит список картинок товара;

10. Sale – хранит информацию о проданных товарах.

Связь указывает отношение между двумя сущностями в виде линии. Существует три типа связи:

1. Много ко многим (М:М) – несколько объектов из одной таблицы соответствует нескольким объектам из другой таблицы;

2. Один ко многим (1:М) – одному объекту из одной таблицы соответствует несколько объектов из другой;

3. Один к одному (1:1) – один объект из одной таблицы соответствует одному объекту из другой таблицы.

В разрабатываемой модели указываются следующие связи:

1. Один товар может быть продан по нескольким товарных накладным, а также по одной товарной накладной может быть продано несколько товаров (М:М);

2. У одного товара может быть несколько товаров (1:М);

3. У одного бренда может быть несколько товаров (1:М);

4. У одно клиента могут быть только одни данные для авторизации (1:1);

5. В одной категории есть несколько подкатегорий (1:М);

6. Одна единица измерения может быть у нескольких товаров (1:М).

7. Один клиент может совершить несколько покупок (1:М)

## 1.2 Обоснование выбора языка программирования, среды программирования и СУБД

Язык программирования — это набор формальных правил, по которым пишут программы. Обычный язык нужен для общения людей, а язык программирования — для общения с компьютером. Как и в любом естественном языке, тут есть лексика — слова, функции и операторы, из которых по правилам синтаксиса составляются выражения.

При выборе языка программирования, первым этапом требуется выявить требования к ней.

Существует два типа разработки:

1. Нативная – это создание продукта исключительно для одной операционной системы;
2. Кроссплатформенная – это разработка одного программного продукта, имеющий возможность работать на нескольких операционных системах.

Основной причиной выбора нативной разработки является максимальная оптимизация для работы со своей системой, в отличие от кроссплатформенных. Тем самым приложение, написанное для одной операционной системы будет более быстродействующим и менее ресурса-затратным.

В настоящее время существует большое количество разновидностей языков для разработки: C, C++, C#, Python, Java.

Язык Си, созданный Денисом Ритчи в начале 70-х годов в Bell Laboratory американской корпорации AT&T, является одним из универсальных языков программирования. Язык Си считается языком системного программирования, хотя он удобен и для написания прикладных программ. Среди преимуществ языка Си следует отметить переносимость программ на компьютеры различной архитектуры и из одной операционной системы в другую, лаконичность записи алгоритмов, логическую стройность программ. Программа, написанная на языке Си, состоит из операторов. Каждый оператор вызывает выполнение некоторых действий на соответствующем шаге выполнения программы.

С++ содержит средства для создания программ практически любого назначения, от низкоуровневых до сложных программных комплексов разного назначения. поддерживает разные технологии и стили программирования, включая объектно-ориентированное, обобщенное. Важным достоинством является предсказуемое выполнение программ, что является важным для систем реального времени.

Java  -  это язык программирования общего назначения, который следует парадигме объектно-ориентированного программирования и подходу «Написать один раз и использовать везде» . Java используется для настольных, сетевых, мобильных и корпоративных приложений. Подробная информация:

Язык С# создан как первый компонентно-ориентированный язык программирования семейства С/С++, является объектно-ориентированным языком, в котором любая сущность представляется объектом, но более простым по сравнению с С++, и сохраняющим его мощь и основные конструкции. Главным преимуществом языка C# перед остальными является, работ на платформе .Net Framework позволяющий удобно работать с базой данных, а также удобство написания кода.

В качестве среды разработки был выбран мощнейший инструмент - Visual Studio 2019, оболочка .NetFramework 4.7.

Microsoft SQL Server 2008 призван помочь справиться с этими проблемами. MS SQL Server 2008 представляет собой интегрированное решение по управлению и анализу данных, которое помогает:

- строить, развертывать и управлять промышленными приложениями, которые являются более безопасными, масштабируемыми и надежными.

- увеличивать продуктивность информационных технологий, уменьшая сложность построения, развертывания и управления приложениями.

- разделять данные между платформами, приложениями и устройствами для облегчения соединения внутренних и внешних систем.

- контролировать стоимость, не жертвуя качеством выполнения, доступностью, масштабируемостью и безопасностью.

База данных (БД) обеспечивает хранение информации и представляет собой совокупность данных, организованных по определенным правилам. База данных позволяет структурировать, хранить и обрабатывать данные различного типа.

Система управления базами данных (СУБД) - это совокупность языковых и программных средств, предназначенных для создания, ведения и использования БД.

В рамках разрабатываемого проекта используется реляционная база данных, основным преимуществом которой является то, данные хранятся в формате таблиц и строго структурированы. К современным реляционным СУБД относятся Oracle, PostgreSQL, MySQL, Microsoft SQL Server.

PostgreSQL отличается от остальных СУБД тем, что является объектно-ориентированным, это означает что данные моделируются в виде объектов, их атрибутов. Разрабатываемая модель не предусматривает выполнения сложных операций, а данная модель больше предназначена для обработки нескольких операций, и в простых операциях не отличается быстротой обработки.

MySql простая в использовании СУБД. Данная СУБД имеет хорошую производительность как при сложных операциях, так и при простых. Для работы с MySQL в Visual Studio существуют как официальные, так и пользовательские плагины, упрощающие работу с ним.

MariaDB является ветвлением MySQL. MariaDB предоставляет множество уникальных функций, отсутствующих в MySQL. В рамках создаваемого проекта MariaDB уступает MySQL тем, что время для разработки приложения ограничено, а для освоения данного СУБД требуется большое количество времени.

Специфика работы сервера базы данных MS SQL Server заключается в транзакционной обработке данных. Это означает, что по каждому запросу от СУБД обрабатывается и сохраняется небольшое количество информации, тем самым SQL Server является более подходящим для вставки и удаления данных.

MySQL и SQL Server являются очень похожими по своему синтаксису и параметрам. Ключевым отличием в разработке является возможность работы с средой .NET. Обе СУБД имеют возможность подключения к Visual Studio через Entity Framework, но MySQL в отличие SQL Server для подключения требует установки дополнительных плагинов. В свою же очередь .Net Framework в первую очередь нацелен под работу с SQL Server, тем самым упрощает процесс разработки.

В результате сравнения СУБД SQL Server является самой подходящей при разработке базы данных для данной информационной системы.

# 2 ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ПРОДУКТА

## 2.1 Графическое представление программы

В ходе реализации проекта может возникнуть необходимость внесения изменений в базу данных или в само приложение. Это может привести к большим временным затратам. Для лучшего понимания концепции приложения, его функционала и уменьшения ошибок в ходе реализации проекта, требуется заблаговременное моделирование информационной системы. Одним из вариантов визуализации приложения является представление его в виде диаграммы. Диаграмма позволяет визуально показать как должна работать программа и его структуру.

Одним из способов моделирования является специализированный язык Unified Modeling Language (UML), имеющий все необходимые графические объекты. Язык UML имеет два типа диаграмм: структурную, отражающая из чего состоит система и поведенческие, в этом типе описывается, что делает система.

Структурной является диаграмма модели данных (ER-модель). ER-модель отображает сущности базы данных и связь между ними. ER моделирование помогает анализировать требования к данным для создания хорошо спроектированной базы данных.

Поведенческими являются диаграммы вариантов использования и последовательности. Диаграммы прецедентов отражают динамическое поведение работающей системы. Он моделирует, как пользователь взаимодействует с системой. Диаграмма последовательности раскрывает структуру системы, позволяет реализовать простой сценарий действий, выполняемых пользователем в системе.

## 2.2 UML диаграмма прецедентов (use case diagram)

Диаграмма use case отражает функциональные возможности с использованием действующих лиц и вариантов использования.

Диаграмма use case отображается с помощью таких объектов, как актер, варианта использования, а также стрелка отображающая принадлежность функции к актеру. Актер – это сущность, взаимодействующая с системой. Вариант использования представляет отдельную функциональность системы. Он обозначен овальной формой с названием выполняемой функции, написанной внутри овальной формы.

Разрабатываемая информационная система предназначена для двух типов пользователей: клиента и администратора (Рисунок 2.1). Клиент может просматривать каталог товаров, фильтровать его по необходимым требования, просматривать подробную информацию и картинки товара, просматривать статус своего заказа и авторизироваться. Этот актер может выполнять только указанные в диаграмме взаимодействия с системой, даже если в системе остаются другие варианты использования. Администратор в свою очередь может добавлять новые товары, редактировать и удалять их по необходимости, изменять статус товар, просматривать заказы клиентов.



Рисунок 2.1 Диаграмма Use Case

2.3 Диаграмма ERD

Диаграмма Entity-Relationship (ERD) предоставляет предварительный просмотр того, как все таблицы должны соединиться, какие поля будут в каждой таблице.

В ER-моделях и моделях данных обычно выделяют до трех уровней детализации: концептуальная, физическая, логическая. Для создания модели подходящим вариантом является концептуальная модель. Концептуальная модель имеет схему с минимальным количеством подробностей в отличии от физической и логической, что позволяет отобразить общую структуру модели.

Для создания модели используют два основных элемента: сущность и связь. Сущность – это объект, отображающийся в виде таблицы и содержащий набор атрибутов.

Связь указывает отношение между двумя сущностями в виде линии. Существует три типа связи:

1. Много ко многим (М:М) – несколько объектов из одной таблицы соответствует нескольким объектам из другой таблицы;
2. Один ко многим (1:М) – одному объекту из одной таблицы соответствует несколько объектов из другой;
3. Один к одному (1:1) – один объект из одной таблицы соответствует одному объекту из другой таблицы.

Линия соединяет первичный и вторичный ключ между двумя сущностями, так чтобы стрелка указывала на первичный ключ. На концах линии указывается тип связи между сущностями (Рисунок 1.2).

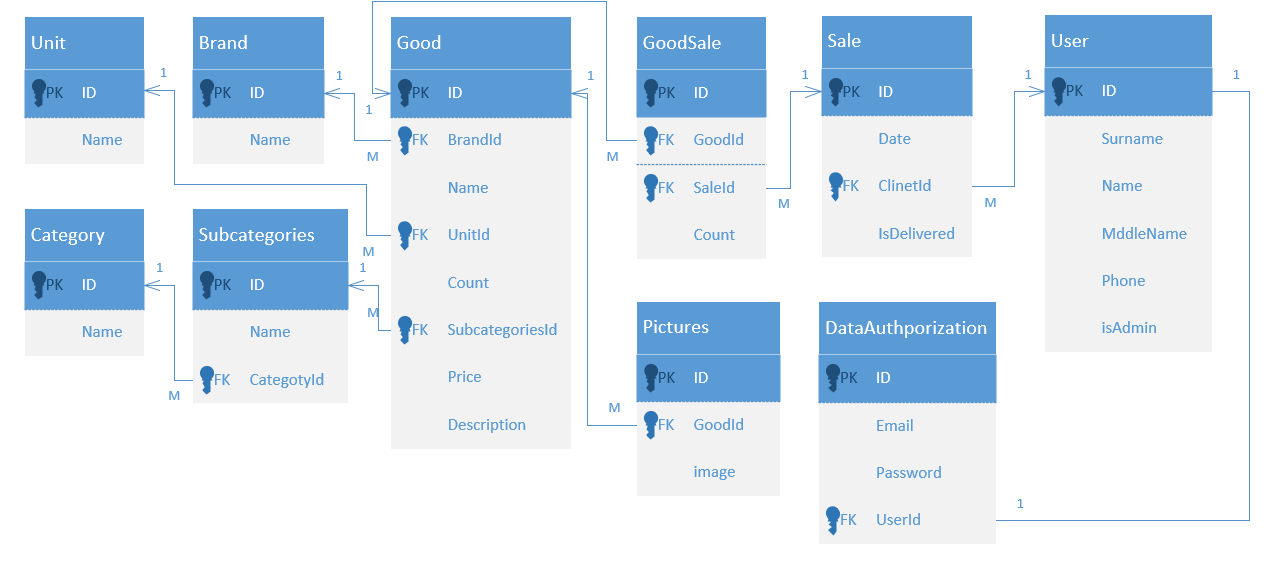


Рисунок 2.2 Диаграмма ERD

## 2.4 Диаграмма последовательности

В ходе разработки приложения возникает необходимость визуального представления последовательности действий клиента, для выполнения определённой функции программы. Представить их можно с помощью диаграммы последовательности.

Диаграмма последовательности моделирует взаимодействие между объектами в едином сценарии использования. Тем самым иллюстрирует, как взаимодействуют между собой отдельные объекты системы для выполнения функции, а также порядок, в котором происходит взаимодействие при выполнении конкретного случая использования.

Первым на диаграмме отображается сам пользователь, в данном случае это клиент автосервиса (Рисунок 2.3). Каждый этап жизни отображается в виде вертикальной пунктирной линии. Каждая линия жизни имеет полосу активности, которая показывает интервал активности участника при взаимодействии. Полоса отображается в виде зеленого выделения линии жизни. Она соответствует временному отрезку от общего времени использования пользователем. Для отображения действий, которые нужно сделать пользователю для авторизации используется стрелка, на которой в краткой форме описано само действие.

После успешной авторизации выполняется переход на главную страницу, на диаграмме отображается с помощью стрелки и описанием выполненного действия. После оформления заказа, подтверждения и оплаты работки отправляет товар клиенту, на диаграмме это отображается в виде стрелки в обратную стрелки к клиенту.



Рисунок 2.3 Диаграмма последовательности

## 2.5 Диаграмма IDEF

Описание системы с помощью IDEF0 называется функциональной моделью. Функциональная модель предназначена для описания существу­ющих бизнес-процессов, в котором используются как естественный, так и графический языки. Для передачи информации о конкретной системе источником графического языка является сама методология IDEF0.

**Методология IDEF0** предписывает построение иерархической системы диаграмм - единичных описаний фрагментов системы. Сначала проводит­ся описание системы в целом и ее взаимодействия с окружающим миром (контекстная диаграмма), после чего проводится функциональная деком­позиция - система разбивается на подсистемы и каждая подсистема опи­сывается отдельно (диаграммы декомпозиции). Затем каждая подсистема разбивается на более мелкие и так далее до достижения нужной степени подробности.

Каждая **IDEF0-диаграмм,** а содержит блоки и дуги. Блоки изображают функции моделируемой системы. Дуги связывают блоки вместе и отобра­жают взаимодействия и взаимосвязи между ними.

Функциональные блоки (работы) на диаграммах изображаются прямоугольниками, означающими поименованные процессы, функции или задачи, которые происходят в течение определенного времени и имеют распознаваемые результаты. Имя работы должно быть выражено отглагольным существительным, обозначающим действие.

**IDEF0** требует, чтобы в диаграмме было не менее трех и не более шести блоков. Эти ограничения поддерживают сложность диаграмм и модели на уровне, доступном для чтения, понимания и использования.

Каждая сторона блока имеет особое, вполне определенное назначение. Левая сторона блока предназначена для входов, верхняя - для управления, правая - для выходов, нижняя - для механизмов. Такое обозначение отражает определенные системные принципы: входы преобразуются в выходы управление ограничивает или предписывает условия выполнения преобразований, механизмы показывают, что и как выполняет функция.

Блоки в IDEF0 размещаются по степени важности, как ее понимает автор диаграммы. Этот относительный порядок называется доминированием. Доминирование понимается как влияние, которое один блок оказывает на другие блоки диаграммы. Например, самым доминирующим блоком диаграммы может быть либо первый из требуемой последовательности функций, либо планирующая или контролирующая функция, влияющая на все другие.

Наиболее доминирующий блок обычно размещается в верхнем левом углу диаграммы, а наименее доминирующий - в правом углу.

Расположение блоков на странице отражает авторское определение доминирования. Таким образом, топология диаграммы показывает, какие функции оказывают большее влияние на остальные. Чтобы подчеркнуть это, аналитик может перенумеровать блоки в соответствии с порядком их доминирования. Порядок доминирования может обозначаться цифрой, размещенной в правом нижнем углу каждого прямоугольника: 1 будет указывать на наибольшее доминирование, 2 - на следующее и т. д.

Взаимодействие работ с внешним миром и между собой описывается в виде стрелок, изображаемых одинарными линиями со стрелками на концах. Стрелки представляют собой некую информацию и именуются существительными (Рисунок 2.4).

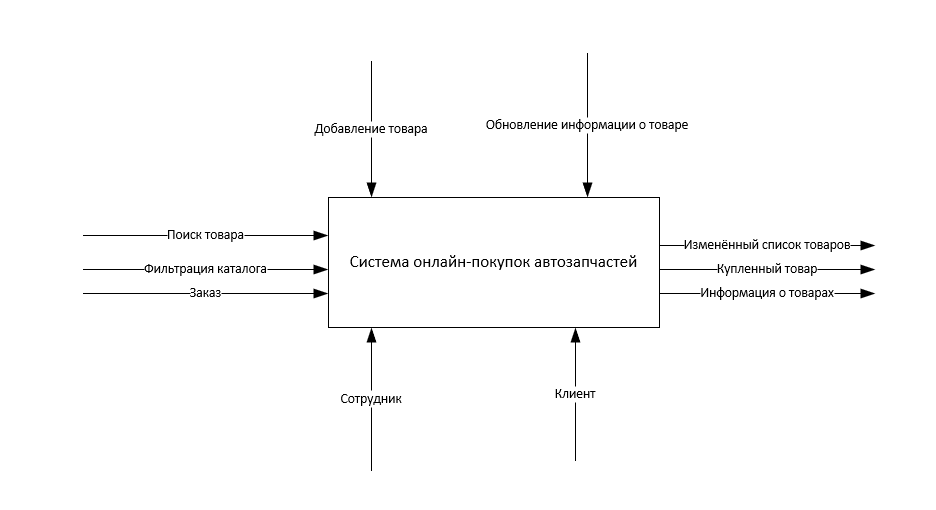


Рисунок 2.4 Диаграмма IDEF0

Стрелками входа являются непосредственно товары и заказы.

В качестве стрелок управления выступает поиск, фильтрация, добавления, удаления, редактирования товаров

К стрелкам механизма относятся клиенты и администраторы.

Стрелки выхода: купленные товары, изменённый список товаров, информация о товарах

Для просмотра более подробной схему последовательности данных создаётся схема IDEF1 (Рисунок 2.5).

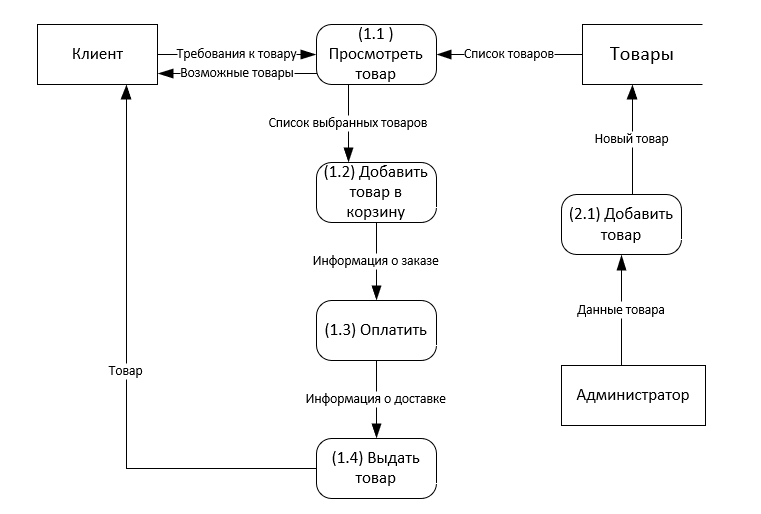


Рисунок 2.5 Диаграмма IDEF0 (нотация Гейна Сарсона)

# 3. РЕАЛИЗАЦИЯ ПРОЕКТА

## 3.1 Создание базы данных

Для быстро и корректной работы программы в начале разработки требуется определить наиболее сбалансированную структуру данных, с помощью анализа предметной области. Требуется разработать такую структуру, которая исключается поверяющиеся объекты и обеспечивает удобный доступ к данным.

В данной предметной области выделаются следующие сановные сущности:

Таблица Brand описывает производителей товаров (Рисунок 3.1).

|  |  |  |  |
| --- | --- | --- | --- |
| Ключ | Наименование | Тип данных | Не пусто |
| PK | Id | int | Not Null |
|  | Name | nvarchar(50) | Null |

Таблица 3.1 Таблица Brand

Таблица Brand описывает единицы измерений продаваемых товаров (Рисунок 3.2).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ключ |  | Наименование | Тип данных | Не пусто |
| PK |  | Id | int | Not Null |
|  |  | Name | nvarchar(20) | Null |

Таблица 3.2 Таблица Unit

Таблица Category описывает категории товаров (Рисунок 3.3).

|  |  |  |  |
| --- | --- | --- | --- |
| Ключ | Наименование | Тип данных | Не пусто |
| PK | Id | int | Not Null |
|  | Name | nvarchar(50) | Null |

Таблица 3.3 Таблица Category

Таблица Subcategory описывает подкатегории товаров (Рисунок 3.4).

|  |  |  |  |
| --- | --- | --- | --- |
| Ключ | Наименование | Тип данных | Не пусто |
| PK | Id | int | Not Null |
|  | Name | nvarchar(20) | Null |
| FK | CategoryId | int | Not Null |

Таблица 3.4 Таблица Subcategory

Таблица User описывает клиентов и администраторов (Рисунок 3.5).

|  |  |  |  |
| --- | --- | --- | --- |
| Ключ | Наименование | Тип данных | Не пусто |
| PK | Id | int | Not Null |
|  | Surname | nvarchar(50) | Not Null |
|  | Name | nvarchar(50) | Not Null |
|  | Phone | char(11) | Not Null |
|  | MiddleName | nvarchar(50) | Null |
|  | IsAdmin | bit | Not Null |

Таблица 3.5 Таблица User

Таблица DataAuthorization описывает данные для авторизации клиента (Рисунок 3.6).

|  |  |  |  |
| --- | --- | --- | --- |
| Ключ | Наименование | Тип данных | Не пусто |
| PK | Id | int | Not Null |
|  | Email | nvarchar(60) | Not Null |
|  | Password | nvarchar(50) | Not Null |

Таблица 3.6 Таблица DataAuthorization

Таблица Good описывает информацию о товаре (Рисунок 3.7).

|  |  |  |  |
| --- | --- | --- | --- |
| Ключ | Наименование | Тип данных | Не пусто |
| PK | Id | int | Not Null |
| FK | BrandId | int | Not Null |
|  | Name | nvarchar(60) | Not Null |
|  | Count | int | Not Null |
| FK | SubcategoriesId | int | Not Null |
|  | Price | decimal(18, 2) | Not Null |
|  | UnitId | int | Not Null |
|  | Description | nvarchar(MAX) | Null |
|  | IsActive | bit | Not Null |

Таблица 3.7 Таблица Good

Таблица Sale описывает товарные накладные (Рисунок 3.8).

|  |  |  |  |
| --- | --- | --- | --- |
| Ключ | Наименование | Тип данных | Не пусто |
| PK | Id | int | Not Null |
|  | Date | date | Not Null |
|  | IsDelivered | bit | Not Null |
| FK | ClientId | int | Not Null |

Таблица 3.8 Таблица Sale

Таблица GoodSale описывает товары из товарной накладной (Рисунок 3.9).

|  |  |  |  |
| --- | --- | --- | --- |
| Ключ | Наименование | Тип данных | Не пусто |
| PK | Id | int | Not Null |
| FK | GoodId | int | Not Null |
| Fk | SaleId | int | Not Null |
|  | Count | int | Not Null |

Таблица 3.9 Таблица GoodSale

Таблица Pictures хранит список картинок товара (Рисунок 3.10).

|  |  |  |  |
| --- | --- | --- | --- |
| Ключ | Наименование | Тип данных | Не пусто |
| PK | Id | int | Not Null |
| FK | GoodId | int | Not Null |
|  | Image | byte | Not Null |

Таблица 3.10 Таблица Picture

## 3.2 Создание приложения

Создание приложение начинается с создания пустого проекта WPF .Net Framwork. После создания интерфейса наступает этап соединения приложения и БД. Подключение осуществляется с помощью модели ADO.NET. Перед созданием модели добавляются две библиотеки EntityFramework и EntityFramework.SqlServer находящиеся в корневой папке VisualStudio Studio\Shared\Packages\EntityFramework.6.2.0\lib\net45. (Рисунок 3.1).

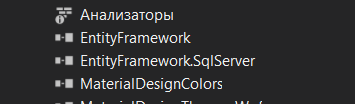


Рисунок 3.1 Библиотеки EF

Далее создаётся модель ADO.NET в окне добавления нового элемента. Преимуществами модели ADO.NЕT является то, что она автоматически генерирует классы тем самым упрощает разработку, а также предоставляет оптимизированный доступ к базам данных SQL Server.

На следующем шаге выбора подключения к данным, добавляется нужное подключение (Рисунок 3.2)

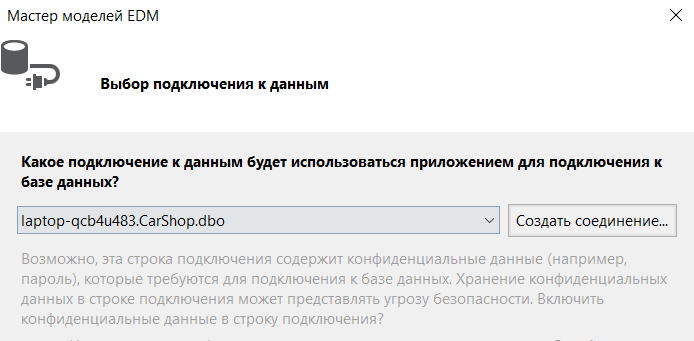


Рисунок 3.2 Выбор подключения к данным

Выбрав подключение на следующем шаге, указываются таблицы для добавления в модель. После выбора таблиц автоматический формируется модель ADO.NET с указанием сущностей и связей между ними (Рисунок 3.3)

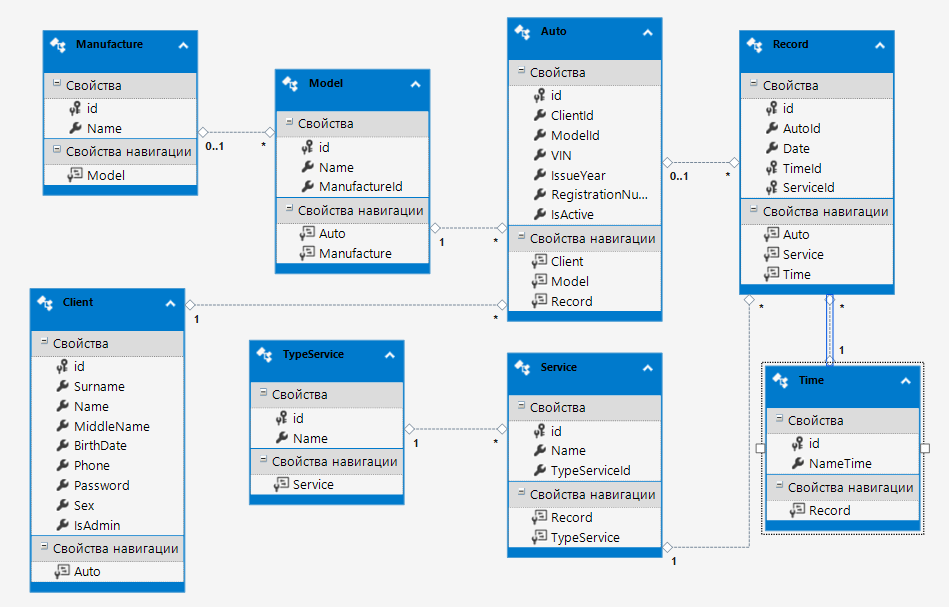


Рисунок 3.3 Модель ADO.NET

Следующим этапом устанавливается NuGet пакет Material Design. Material Design предоставляет дополнительные элементы управления объектов, позволяющих реализовать красивый интерфейс. C помощью диспетчера пакетов NuGet находящийся во кладке «Проект». В диспетчера пакетов выполняется поиск и установка пакета «MaterialDesignThemes» (Рисунок 3.4).

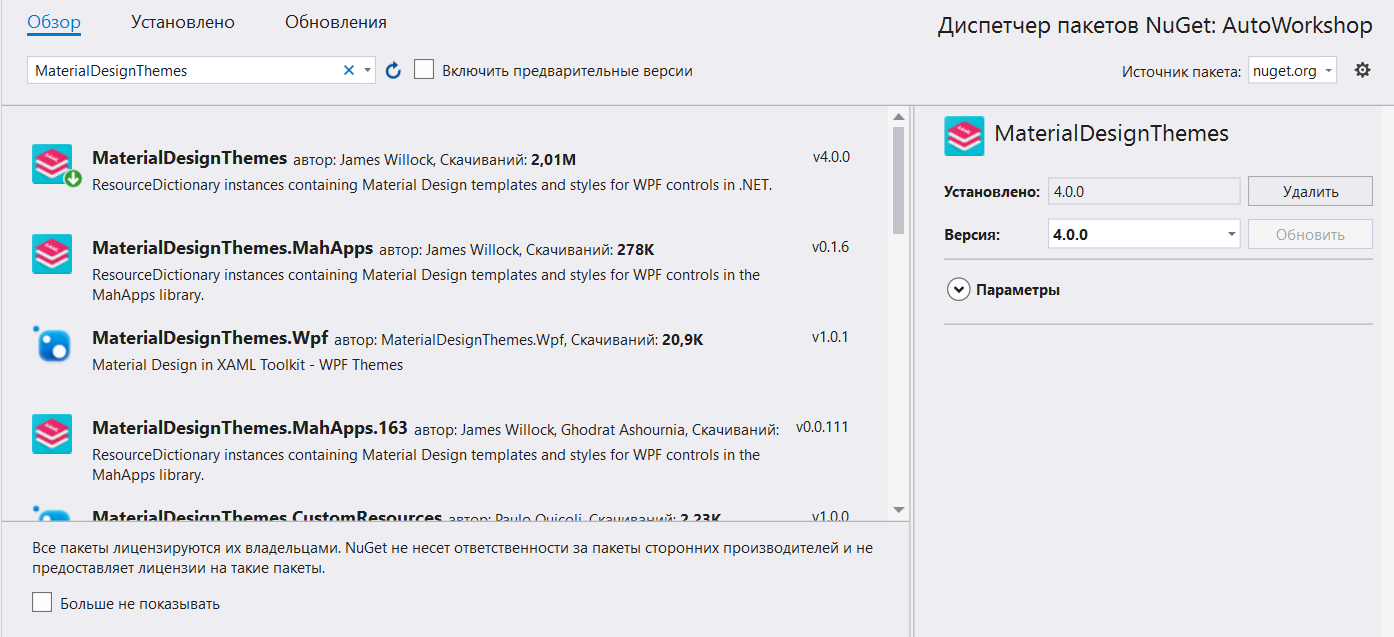


Рисунок 3.4 Диспетчер пакетов NuGet

В файле App.xaml указываются ссылки на компоненты в виде библиотеки ресурсов (Рисунок 3.5.

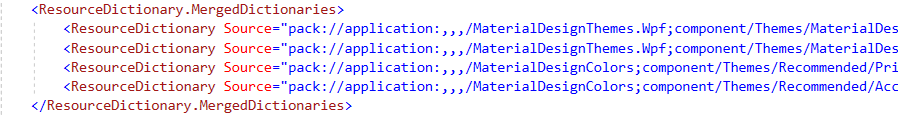


Рисунок 3.5 Ссылки на компоненты Material Design

Для облегчения создания интерфейса следует использовать стили хранящие в себе общие черты объектов (Рисунок 3.6). Стили позволяют централизованно изменить объекты имущие общий стиль, тем самым позволяет быстро вносить изменения на все элементы, упростить разработку и сократить код (Рисунок 3.7).

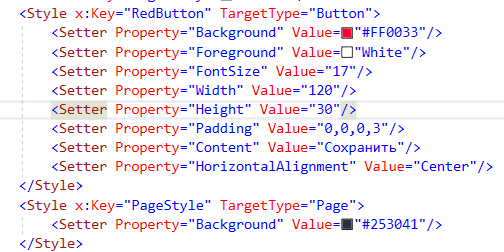


Рисунок 3.6 Style в файле App.xaml

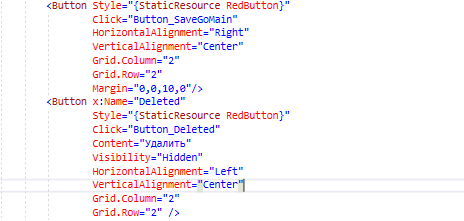


Рисунок 3.7 Использование стилей

Пройдя все этапы подготовки следует этап верстки дизайна на разметки XAML. При разработке интерфейса используются такие элементы как ListBox, ListView, Button, TextBox, TextBlock, CheckBox, Expander. Для изменения внешнего вида элемента использовалось свойство ControlTemplate.

Основной функций в программе является функция корзины товаров. Для её реализации создаётся список SaleProduct содержащий товар и его количество объявленный в виде статичного свойства имеющий модификатор доступа public MainWindow.xaml.cs (Рисунок 3.8). Для осуществление автоматического обновления информации о количества товаров в корзине и общей стоимость используется блок Set, содержащий функции подсчёта и обновления информации на экране.



Рисунок 3.8 Свойство, хранящее список товаров в корзине

Для вывода информации о состоянии корзины в независимости от текущей страницы, следует расположить в MainWindow так как она является навигационной страницей (Рисунок 3.9). А также для быстрого перехода в корзину следует использовать клик по иконке корзины.

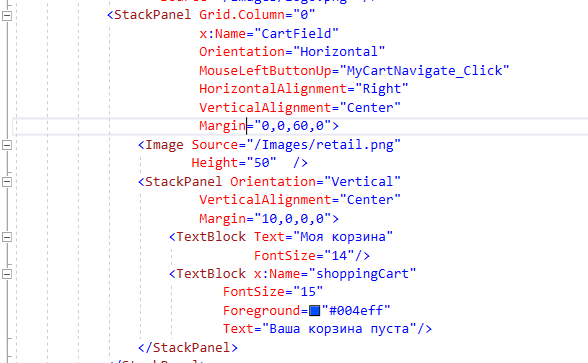


Рисунок 3.9 Код интерфейса состояния корзины

Данные на страницу передаются c помощью свойство DataContext. DataContext – это свойство, поддерживающее привязку, для упрощения сценариев, в которых один контекст может быть привязан к другому. Отображения данных на странице осуществляются с помощью привязок. DataContext является источником привязок по умолчанию. Привязки указываются с помощью ключевого слова Binding после которого следует название свойства для привязки (Рисунок 3.10).

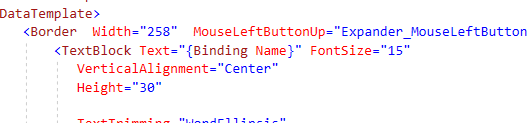


Рисунок 3.10 Добавление привязок

Переход между страницами осуществляется с помощью навигации в Frame расположенный в MainWindow (Рисунок 3.11). Навигация осуществляется с помощью класса NavigationService. Для перехода на новую страницу используется метод Navigate(), принимающий в виде параметра экземпляр страницы, на которую осуществляется переход. Переход на предыдущую страницу осуществляется с помощью метода GoBack().

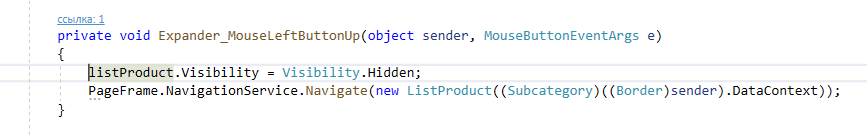


Рисунок 3.11 Функция для перехода cо страницы Main на страницу MainShoppingCart.cs

# 4. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

## 4.1 Запуск программы

Для запуска программы понадобится файл с названием mysetup.exe который находится в папке проекта (Рисунок 4.1).

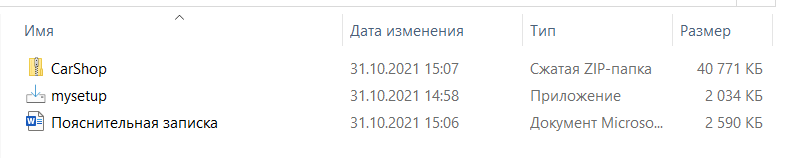


Рисунок 4.1 Папка с установщиком приложения

## 4.2 Работа с программой

После запуска программы открывается главная страница, содержащая список самых покупаемых товаров и информационные баннеры (Рисунок 4.2). В правом верхнем углу находится информацию о состоянии корзины.

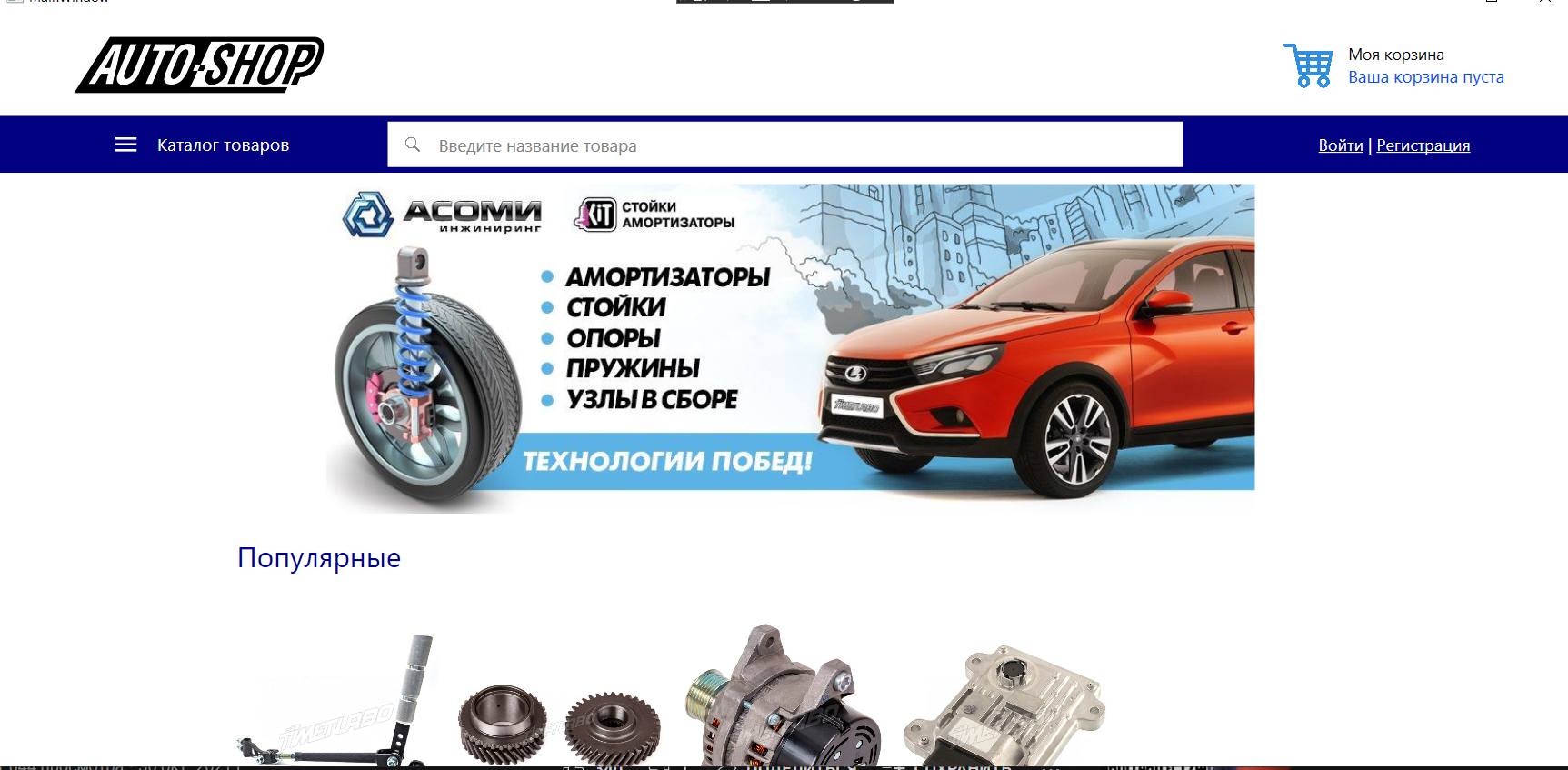


Рисунок 4.2 Главная страница

Для оформления требуется авторизация, в случае если аккаунт отсутствует следует зарегистрироваться. Для перехода в окно регистрации требуется нажать на кнопку «Регистрация» в правом верхнем углу. Для регистрации нового аккаунта требуется заполнить все поля, после нажать кнопку «Регистрация» (Рисунок 4.3). В случае если одно из полей введено не корректно или эл. почта занята высветится соответствующее сообщение.

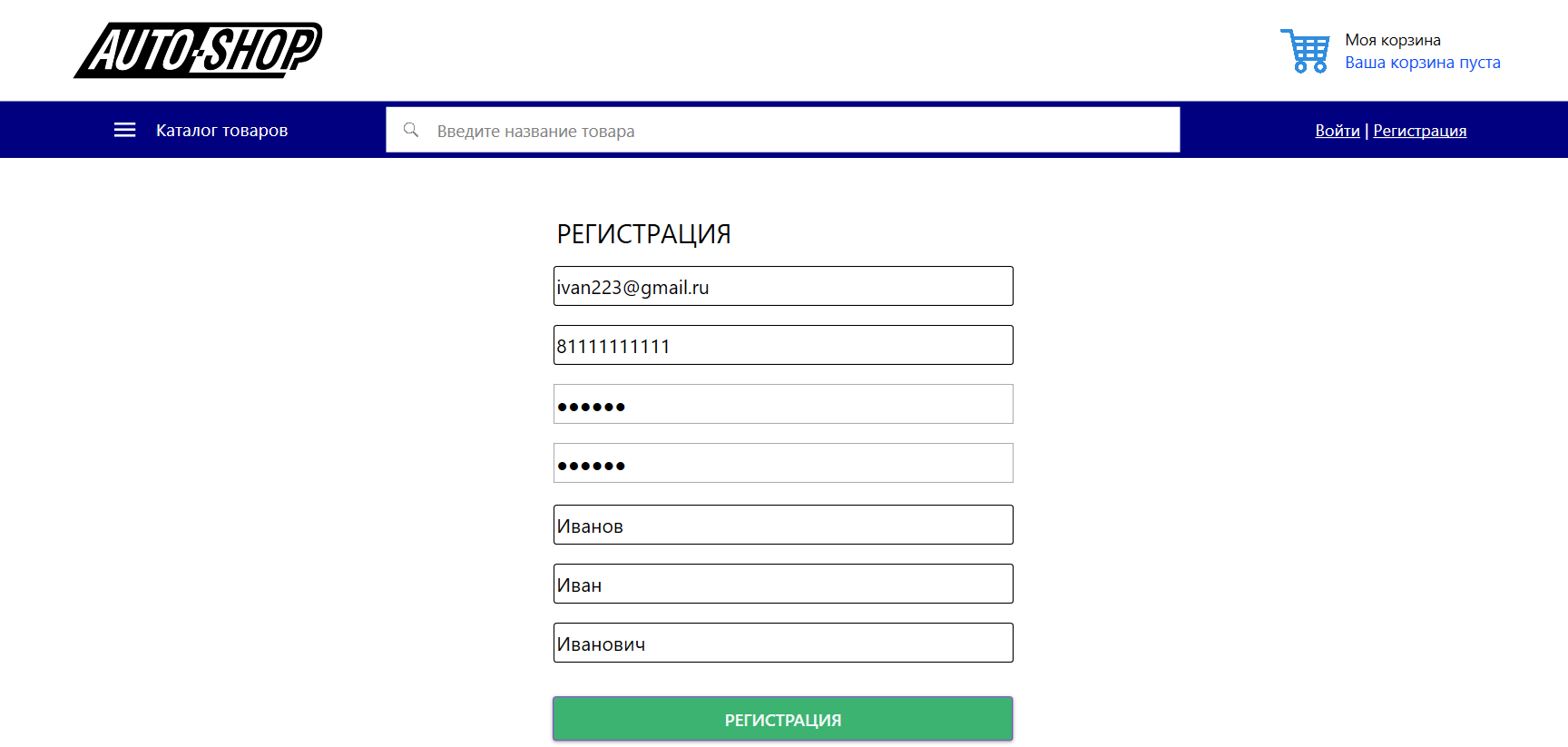


Рисунок 4.3 Страница регистрации

Пройдя этап авторизации, можно приступить к оформлению заказы. Каталог товаров открывается нажатием на «Каталог товаров» в левом верхнем углу. Нажатием на категории товаров выпадает список подкатегорий (Рисунок 4.4).

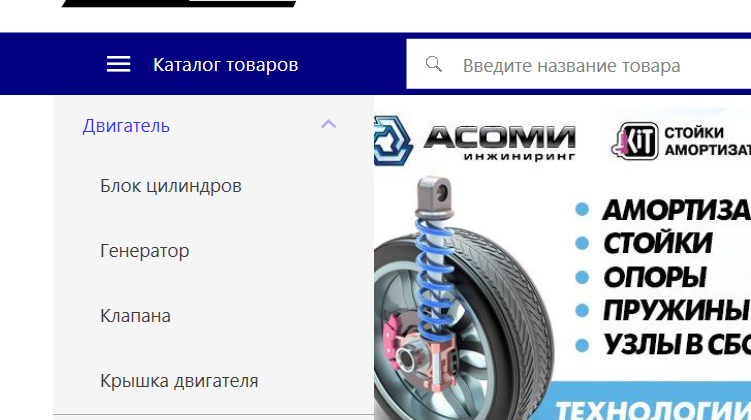


Рисунок 4.4 Каталог товаров

Выбрав необходимую подкатегорию товаров, открывается страница списка товаров (Рисунок 4.5). В левой части страницы находятся категории фильтрации. Так же можно выполнить поиск товаров, введя название товара в поле «Введите название товара» а нажав кнопку «Enter».

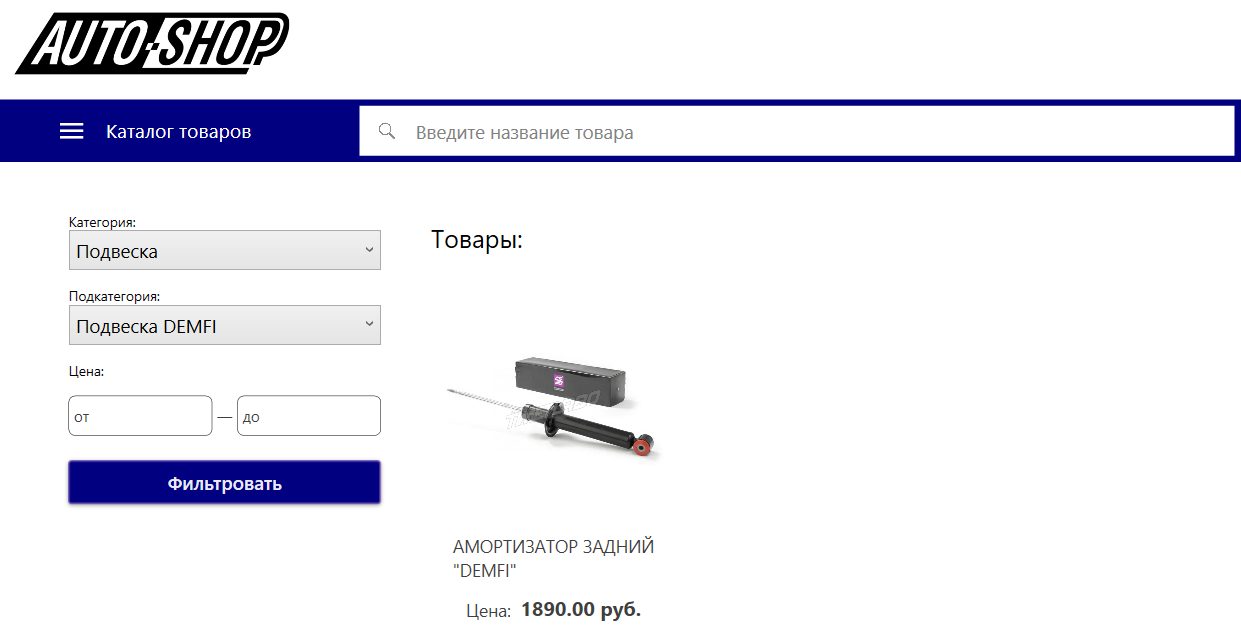


Рисунок 4.5 Каталог товаров

Выбрав необходимый товар, открывается страница подробной информации о товаре (Рисунок 4.6).



Рисунок 4.6 Страница подробной информации о товаре

Для добавления товара в корзину необходимо нажать на кнопку "В корзину». Добавив все необходимые товары следует перейти в корзину для оформления заказа нажав на поле «Моя корзина». Перейдя в корзину, открывается список товаров, находящихся в нём (Рисунок 4.7). При необходимости можно удалить ненужные товары выбрав один или более товаров и нажать на кнопку «Удалить», либо если есть необходимость полностью очистить корзину, то нажать на кнопку «Удалить всё». Так же на данную страницу можно перейти с помощью нажатия на кнопку «Личный кабинет».

И выбрав в закладках «Корзина». Для оформления заказа требуется нажать на кнопку «Оформить».

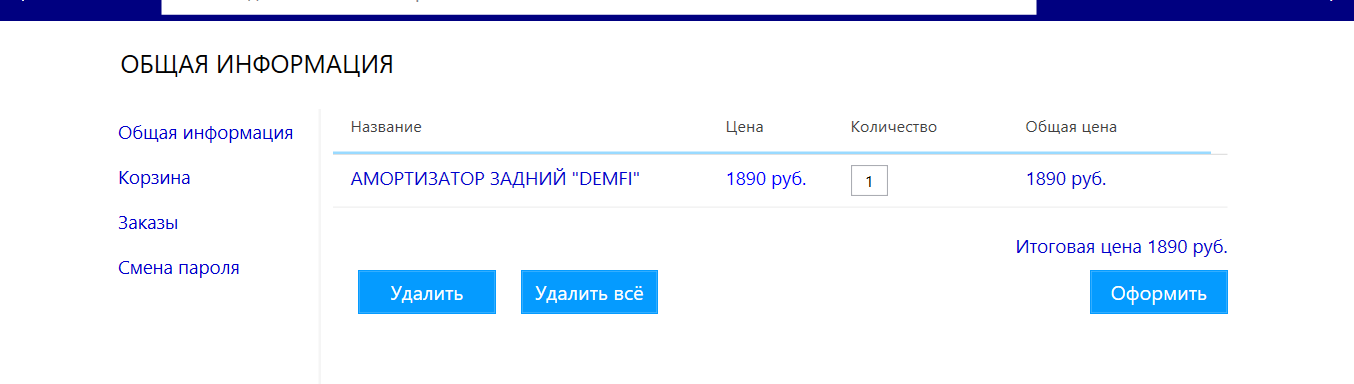


Рисунок 4.7 Корзина товаров

При успешном оформлении заказа высветиться соответствующее сообщение (Рисунок 4.8).

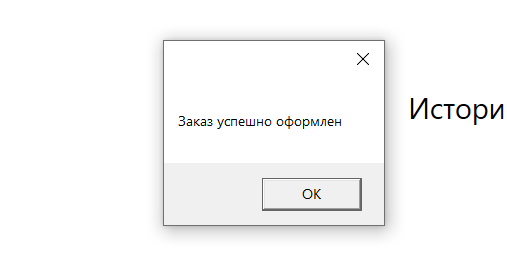


Рисунок 4.8 Сообщение об успешном оформлении заказа

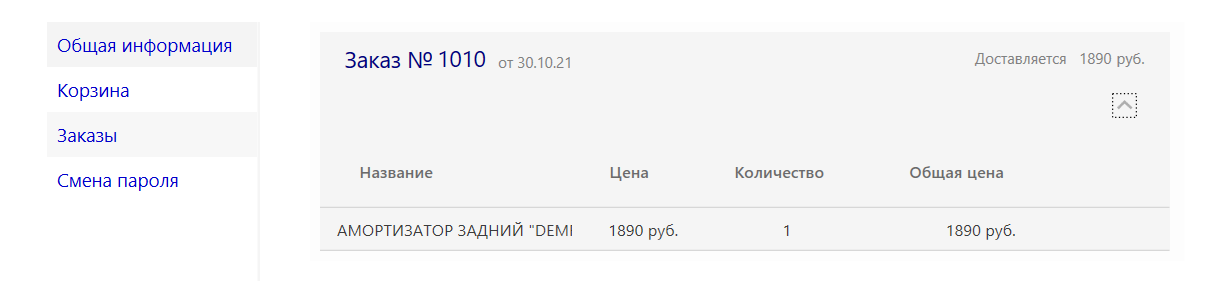
Теперь данный заказ можно отселить во вкладке заказы (Рисунок 4.9) 

Рисунок 4.9 Страница заказов клиента

Так же редактировании данных или смена пароля можно осуществить во вкладках «Общая информация» и «Смена пароля» соответственно.

При работке с программой в роле администратора открывается окно списка товаров (Рисунок 4.10).

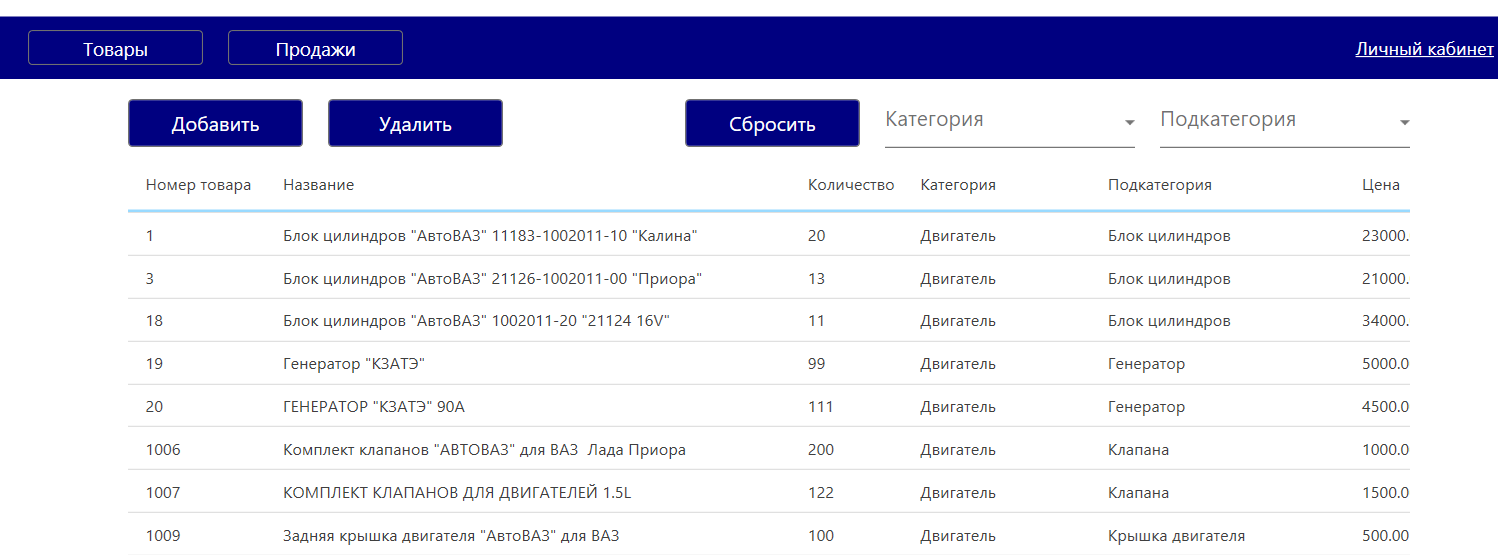


Рисунок 4.10 Страница редактирования списка товаров.

Для удобного поиска товаров есть фильтрации по категориям и под категории.

Для добавления нового продукта требуется нажать кнопку «Добавить».

В отрывшемся окне следует заполнить все поля и добавить картинки товара. Заполнив все поля для заверения добавления требуется нажать на кнопку (Рисунок 4.11). После нажатия кнопки «Сохранить» программа вернётся на предыдущую страницу.

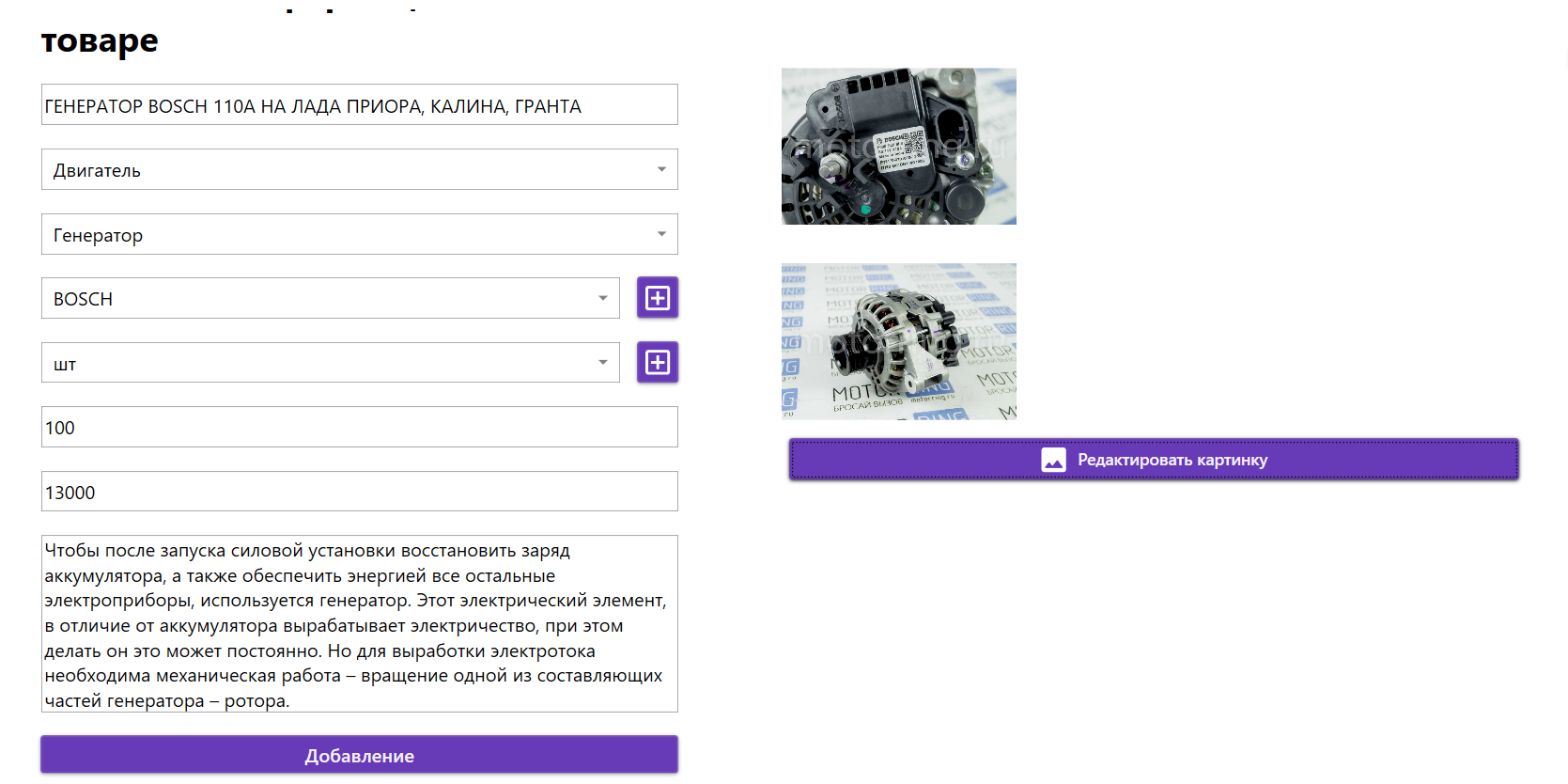


Рисунок 4.11 Страница добавления/редактирования товара

Для удаления товара необходимо выбрать один или несколько товаров и нажать кнопку удалить.

Для просмотра списка заказов необходимо нажать на вкладку «Заказы» (Рисунок 4.12). Откроется список заказов для того чтобы указать что данный заказ доставлен необходимо перевести поле флажка «Доставка» в истинное значение путём нажатия на него.

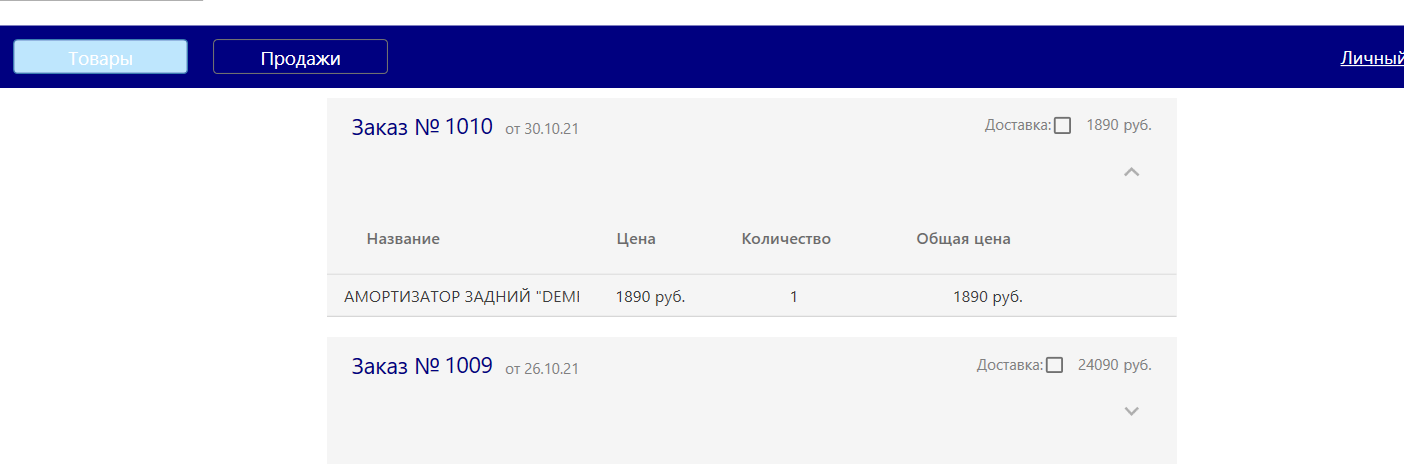


Рисунок 4.12 Список заказов для администратора

5. ТЕСТИРОВАНИЕ

## 5.1 Виды тестирования

Целью тестирования программного обеспечения является выявление ошибок, пробелов или недостающих требований в отличие от фактических требований. Для тестирования проводится с помощью тест - кейса.

Модульное тестирование выполняется на ранних этапах, когда готовятся отдельные куски приложения (классы, компоненты, функции). В этот момент тестировщики скрупулезно пишут автоматические тесты для каждой функции будущей программы. Это необходимо потому, что проверить «софт» в графическом интерфейсе пока нереально, да и автоматика дает лучший результат.

Интеграционное тестирование проводится на следующем этапе, когда некоторые модули объединяются и превращаются в более крупный компонент, более приближенный к готовой программе. Тестировщики проверяют, как ведут себе ранее разобщенные модули, совмещенные в единый продукт, и как этот готовый продукт функционирует сам по себе. Также на стадии интеграции проверяется совместимость создаваемого ПО с операционной системой, на которой оно будет работать, а иногда еще и с аппаратной частью, чтобы создаваемый продукт нормально функционировал на большем количестве устройств.

Системное тестирования тесно привязана к функциональному и нефункциональному типу. Именно в ходе системного тестирования специально обученные люди проверяют, соответствует ли детище программистов тому, что было заявлено с точки зрения возможностей, и стандартам качества компании с позиции производительности, отзывчивости, отказоустойчивости и прочих технических аспектов.

Тест-кейсы – конкретные действия или наборы действий, выполняемые тестировщиками, чтобы оценить работоспособность ПО. Здесь важно учесть те сценарии, которые будут наиболее близки к реальности.

Для проверки разрабатываемого приложения идеально подходит unit- тест, так как не требуют специальных знаний, а так проверяю весь функционал со стороны пользователя.

## 5.2 Тест-кейсы

Тест-кейсы разрабатывается для основных функций программ:

1. Регистрация клиента (Таблица 5.1)
2. Оформление заказа (Таблица 5.2)
3. Добавление товара (Таблица 5.3)

|  |  |
| --- | --- |
| **Test Case #** | 1 |
| **Приоритет теста** | Высокий |
| **Название тестирования/Имя** | Регистрация нового пользователя, почта которого уже занята |
| **Резюме испытания** | Необходимо чтобы программа выводила сообщение об ошибке при недоступной почте. |
| **Шаги тестирования** | 1. Нажать на вкладку “Регистрация”;  2. В отрывшемся окне заполнить поля: Почта, Номер телефона, Фамилия, Имя, Отчество;  3. Нажать на кнопку “Регистрация”; |
| **Данные тестирования** | Email: rail@mail.ru; Phone: +77777777777; Фамилия: Абдраков; Имя: Раиль ; Отчество: Ильдарович |
| **Ожидаемый результат** | Появление сообщения о недоступности почты |
| **Фактический результат** | Появление сообщения о недоступности почты |
| **Предпосылки** | Запустить программу |
| **Постусловия** | Возврат на страницу регистрации |
| **Статус (Pass/Fail)** | Pass |
| **Комментарии** | Отсутствует |

Таблица 5.1 – Тест-кейс для проверки работоспособности регистрации нового пользователя

|  |  |
| --- | --- |
| **Test Case #** | 2 |
| **Приоритет теста** | Высокий |
| **Название тестирования/Имя** | Оформление заказа |
| **Резюме испытания** | Необходимо чтобы программа успешно оформила заказ |
| **Шаги тестирования** | 1. Нажать на список категория товаров; 2. Выбрать любую подкатегорию товаров; 3. Нажать на товар; 4. Нажать на кнопку “Добавить в корзину”; 5. Нажать на поле “Моя корзина” 6. В открывшемся окне нажать кнопку “Оформить заказ” 7. Нажать кнопку “Подтвердить” |
| **Данные тестирования** | Клиент: Email: airat@gmail.ru Password: 123qwe |
| **Ожидаемый результат** | Сообщение об успешном оформлении заказа |
| **Фактический результат** | Сообщение об успешном оформлении заказа |
| **Предпосылки** | Запустить программу |
| **Постусловия** | Возврат на главную страницу |
| **Статус (Pass/Fail)** | Pass |
| **Комментарии** | Отсутствует |

Таблица 5.2 – Тест-кейс для проверки работоспособности оформления заказа

|  |  |
| --- | --- |
| **Test Case #** | 2 |
| **Приоритет теста** | Высокий |
| **Название тестирования/Имя** | Добавление товара |
| **Резюме испытания** | Необходимо чтобы программа успешно добавила товар в базу данных |
| **Шаги тестирования** | 1. Авторизироваться под администратором; 2. Нажать на кнопку товары 3. В открывшемся окне нажать кнопку “Добавить” 4. Заполнить все поля; 5. Нажать на кнопку “Добавить в корзину”; 6. Нажать на поле “Моя корзина” 7. В открывшемся окне нажать кнопку “Оформить заказ” 8. Нажать кнопку “Подтвердить” |
| **Данные тестирования** | Администратор: Email: admin Password: admin  Товары: Название: ПТФ Sal-Man; Категория: Автоэлектроника; Подкатегория: автосвет; Количество: 100; Единица измерения: кг; Бренд: Sal - man |
| **Ожидаемый результат** | Сообщение об успешном добавлении товара |
| **Фактический результат** | Сообщение об успешном добавлении товара |
| **Предпосылки** | Запустить программу |
| **Постусловия** | Возврат на главную страницу |
| **Статус (Pass/Fail)** | Pass |
| **Комментарии** | Отсутствует |

Таблица 5.3 – Тест-кейс для проверки добавления нового товара

6. СОЗДАНИЕ УСТАНОВЩИКА ПРОГРАММЫ

Для создания установщика была выбрана программа InnoSetup, имеющая удобный и простой интерфейс.

Для начала требуется получить сам exe файл, путём запуска программы в режиме Release. Следующим этапом в программе InnoSetup создаётся новый проект указывается ссылка на exe файл, а также на папку самого проекта (Рисунок 6.1).

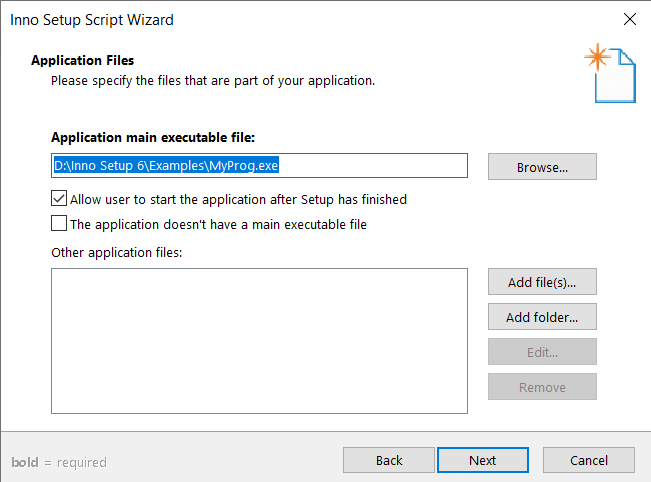


Рисунок 6.1 Создание установщика

На следующем шаге указывается место, в котором будет создан файл установщик. После прохождения всех этапом формируется скрипт установщика, после запуска скрипта создаётся сам установщик.

ЗАКЛЮЧЕНИЕ

Результатом курсовой работы является создание информационной системы для онлайн – покупок автозапчастей, основным назначением которой является продажа автомобильных запчастей и комплектующих. Разработанный информационная система помогает систематизировать учёт продажи товаров.

В ходе разработки информационной системы для онлайн - покупок автозапчастей были приобретены практические навыки анализа предметной области, логической и физической разработки, проанализированы функциональные возможности онлайн - магазинов, отдельные компоненты применены на практике.

При выполнении проекта были пройдены все этапы разработки специализированного прикладного программного обеспечения:

* постановка задачи;
* сбор необходимых исходных данных;
* разработка и отладка программного продукта.

С помощью СУБД MS SQL Server была разработана структура базы данных, предоставляющая всю информацию в удобном для просмотра виде, позволяя быстро ориентироваться в дальнейшей работе с ней.

Для разработки приложения была выбрана платформа WPF позволяющая удобно работать с СУБД, был спроектирован консольный интерфейс, который обеспечивает удобную работу клиента магазина с приложением. Интерфейс реализован в виде страниц, в которых есть наборы элементов, с помощью которых можно взаимодействовать с базы данных магазина.

В результате было получено клиентское приложение имеющий весь необходимы функционал для удобных покупок автозапчастей.

# СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Джеральд М. Идеальное программное обеспечение и другие иллюзии в тестирования / М. Джеральд: Gerald M. Weinberg, 2010. – 209 с.
2. Дастин Э. Автоматизированное тестирование программного обеспечения/ Элфрид Дастин, Джефф Рэшка, Джон Пол: ЛОРИ, 2003. – 689 с.
3. Мак-Дональд М. WPF. в .NET 4.5 с примерами на C# 5.0 для профессионалов/ М. Мак-Дональд – 4-е издание: Вильямс, 2013. – 1442 с.
4. Петцольд Ч. Microsoft Windows Presentation Foundation. Базовый курс/ Ч. Петцольд: Русская Редакция, Санкт – Петербург, 2008. – 945 с.
5. Савин Р. Тестирование DOT COM./ Р. Савин: «Депо» Москва, 2007. – 316 с.
6. Книга «Microsoft SQL Server 2014/ Александр Бондарь: БХВ-Петербург, 2015. – 589с.
7. Рафалович В. Data mining, или Интеллектуальный анализ данных для занятых/ В. Рафалович: СмартБук, 2014. – 96 с.
8. Виссер Д. Разработка обслуживаемых программ на языке C# / Джус Виссер: пер. с англ. Р. Н. Рагимова. — М.: ДМК Пресс, 2017. – 192 с.
9. Мартин Р. Принципы, паттерны и методики гибкой разработки на языке C#/ Роберт Мартин, Мика Мартин: Пер. с англ. – СПб.: Символ-Плюс, 2011. – 768 с.
10. Фаулер М.UML, основы: краткое руководство по стандартному языку объектного моделирования/ М. Фаулер: Символ-Плюс, 2011. – 181 с.
11. Правила оформления технического задания. [Электронный курс]. [Электронный ресурс]. – Режим доступа: https://habr.com/ru/post/507414/
12. Отличительные черты WPF и Window Form. [Электронный курс]. [Электронный ресурс]. – Режим доступа: https://www.wpf-tutorial.com/ru/2/wpf-vs-winforms/

# ПРИЛОЖЕНИЕ A

Техническое задание к программе "Информационная система для онлайн - покупок автозапчастей"

СОДЕРЖАНИЕ

1. Введение

1.1. Наименование программы

1.3. Основание для разработки

1.3. Назначение и область применения

2. Требования к программе

2.1. Требования к функциональным характеристикам

2.2. Требования к надежности

2.2.1. Требования к обеспечению надежного функционирования программы

2.2.2. Время восстановления после отказа

2.2.3. Отказы из-за некоректных действий оператора

3. Условия эксплуатации

3.1. Климатические условия эксплуатации

3.2. Требования к квалификации и численности персонала

3.3. Требования к составу и параметрам технических средств

3.4. Требования к информационной и программной совместимости

3.4.1. Требования к информационным структурам и методам решения

3.4.2. Требования к исходным кодам и языкам программирования

3.4.3. Требования к программным средствам, используемым программой

3.4.4. Требования к защите информации и программ

3.5. Специальные требования

4. Требования к программной документации

4.1. Предварительный состав программной документации

5. Технико-экономические показатели

5.1. Экономические преимущества разработки

6. Стадии и этапы разработки

6.1. Стадии разработки

6.2. Этапы разработки

6.3. Содержание работ по этапам

7. Порядок контроля и приемки

7.1. Общие требования к приемке работы

1. ВЕДЕНИЕ

1.1. Наименование программы

Наименование программы: "Информационная система онлайн - покупок автозапчастей"

1.2. Основание для разработки

Основание для разработки является учебный план по специальности 09.02.07 Информационные системы и программирование, квалификация программист.

Плановые сроки начала работы: 09.09.21

Плановый срок окончания работы: 05.11.21

1.3. Назначение и область применения

Продажа автозапчастей через сеть, учёт прихода и ухода запчастей. Удобный пользовательский интерфейс пользователя для быстрого и комфортного просмотра каталога автозапчастей.

2. ТРЕБОВАНИЯ К ПРОГРАММЕ

2.1. Требования к функциональным характеристикам

Программа должна обеспечивать возможность выполнения перечисленных ниже функций:

* Добавление/редактирования/удаления товаров
* Учёт товаров
* Покупку товаров с использованием безналичной оплаты
* Иметь приятный дизайн интерфейса
* Авторизацию пользователей
* Сортировка/фильтрация товаров
* Поиск товаров

2.2. Требования к надежности

2.2.1 Требования к обеспечению надежного функционирования программы

Надежное (устойчивое) функционирование программы должно быть обеспечено выполнением Заказчиком совокупности организационно-технических мероприятий, перечень которых приведен ниже:

а) организацией бесперебойного питания технических средств;

б) использованием лицензионного программного обеспечения;

в) регулярным выполнением рекомендаций Министерства труда и социального развития РФ, изложенных в Постановлении от 23 июля 1998 г.

Об утверждении межотраслевых типовых норм времени на работы по сервисному обслуживанию ПЭВМ и оргтехники и сопровождению программных средств»;

г) регулярным выполнением требований ГОСТ 51188-98. Защита информации. Испытания программных средств на наличие компьютерных вирусов

2.2.2. Время восстановления после отказа

Время восстановления после отказа, вызванного сбоем электропитания технических средств (иными внешними факторами), не фатальным сбоем (не крахом) операционной системы,

не должно превышать 30-ти минут при условии соблюдения условий эксплуатации технических и программных средств.

Время восстановления после отказа, вызванного неисправностью технических средств, фатальным сбоем (крахом) операционной системы, не должно превышать времени, требуемого на устранение неисправностей технических средств и переустановки программных средств.

2.2.3. Отказы из-за некорректных действий оператора

Отказы программы возможны вследствие некорректных действий оператора (пользователя) при взаимодействии с операционной системой.

Во избежание возникновения отказов программы по указанной выше причине следует обеспечить работу конечного пользователя без предоставления ему административных привилегий

3. УСЛОВИЯ ЭКСПЛУАТАЦИИ

3.1. Климатические условия эксплуатации

Климатические условия эксплуатации, при которых должны обеспечиваться заданные характеристики, должны удовлетворять требованиям,

предъявляемым к техническим средствам в части условий их эксплуатации

3.2. Требования к квалификации и численности персонала

Минимальное количество персонала, требуемого для работы программы, должно составлять не менее 1 штатных единиц – оператор.

Системный администратор должен иметь высшее профильное образование и сертификаты компании-производителя операционной системы. В перечень задач, выполняемых системным администратором, должны входить:

а) задача поддержания работоспособности технических средств;

б) задачи установки (инсталляции) и поддержания работоспособности системных программных средств — операционной системы;

в) задача установки (инсталляции) программы.

г) задача создания резервных копий базы данных.

3.3. Требования к составу и параметрам технических средств

3.3.1. В состав технических средств должен входить IВМ-совместимый персональный компьютер (ПЭВМ), выполняющий роль сервера, включающий в себя:

3.3.1.1. процессор Pentium-2.0Hz, не менее;

3.3.1.2. оперативную память объемом, 256 Мбайт, не менее;

3.3.1.3. свободного пространства на жестком диске, 1 Гигабайт, не менее;

3.3.1.4. операционную систему Windows 2000 Server;

3.4. Требования к информационной и программной совместимости

3.4.1. Требования к информационным структурам и методам решения

База данных работает под управлением Microsoft SQL Server. Среда разработки Microsoft Visual Studio. Используется много поточный доступ к базе данных. Необходимо обеспечить одновременную работу с программой с той же базой данной модулей экспорта внешних данных.

3.4.2. Требования к исходным кодам и языкам программирования

Дополнительные требования не предъявляются

3.4.3. Требования к программным средствам, используемым программой

Системные программные средства, используемые программой, должны быть представлены операционной системы Windows 2000 Server и Microsoft SQL Server 2000

3.4.4. Требования к защите информации и программ

Требования к защите информации и программ не предъявляются

3.5. Специальные требования

Специальные требования к данной программе не предъявляются

4. ТРЕБОВАНИЯ К ПРОГРАММНОЙ ДОКУМЕНТАЦИИ

4.1. Предварительный состав программной документации

Состав программной документации должен включать в себя:

4.1.1. техническое задание;

4.1.2. руководство пользователя;

5. ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

5.1. Экономические преимущества разработки

Ориентировочная экономическая эффективность не рассчитываются. Аналогия не проводится ввиду уникальности предъявляемых требований к разработке.

6. СТАДИИ И ЭТАПЫ РАЗРАБОТКИ

6.1. Стадии разработки

Разработка должна быть проведена в три стадии:

1. разработка технического задания;

2. рабочее проектирование;

3. внедрение.

6.2. Этапы разработки

На стадии разработки технического задания должен быть выполнен этап разработки, согласования и утверждения настоящего технического задания.

На стадии рабочего проектирования должны быть выполнены перечисленные ниже этапы работ:

1. разработка программы;

2. разработка программной документации;

3. испытания программы.

На стадии внедрения должен быть выполнен этап разработки подготовка и передача программы

6.3. Содержание работ по этапам

На этапе разработки технического задания должны быть выполнены перечисленные ниже работы:

1. постановка задачи;

2. определение и уточнение требований к техническим средствам;

3. определение требований к программе;

4. определение стадий, этапов и сроков разработки программы и документации на неё;

5. согласование и утверждение технического задания.

На этапе разработки программы должна быть выполнена работа по программированию (кодированию) и отладке программы.

На этапе разработки программной документации должна быть выполнена разработка программных документов в соответствии с требованиями к составу документации.

На этапе испытаний программы должны быть выполнены перечисленные ниже виды работ:

1. разработка, согласование и утверждение и методики испытаний;

2. проведение приемо-сдаточных испытаний;

3. корректировка программы и программной документации по результатам испытаний.

На этапе подготовки и передачи программы должна быть выполнена работа по подготовке и передаче программы и программной документации в эксплуатацию на объектах Заказчика.

7. ПОРЯДОК КОНТРОЛЯ И ПРИЕМКИ

7.1. Общие требования к приемке работы

На основании Протокола проведения испытаний Исполнитель совместно с Заказчиком подписывает Акт приемки-сдачи программы в эксплуатацию.

# ПРИЛОЖЕНИЕ B

App.xaml:

<Application x:Class="CarShop.App"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:local="clr-namespace:CarShop"

StartupUri="/Windows/MainWindow.xaml">

<Application.Resources>

<ResourceDictionary>

<ResourceDictionary.MergedDictionaries>

<ResourceDictionary Source="pack://application:,,,/MaterialDesignThemes.Wpf;component/Themes/MaterialDesignTheme.Light.xaml" />

<ResourceDictionary Source="pack://application:,,,/MaterialDesignThemes.Wpf;component/Themes/MaterialDesignTheme.Defaults.xaml" />

<ResourceDictionary Source="pack://application:,,,/MaterialDesignColors;component/Themes/Recommended/Primary/MaterialDesignColor.DeepPurple.xaml" />

<ResourceDictionary Source="pack://application:,,,/MaterialDesignColors;component/Themes/Recommended/Accent/MaterialDesignColor.Lime.xaml" />

</ResourceDictionary.MergedDictionaries>

</ResourceDictionary>

</Application.Resources>

</Application>

AddProduct.xaml:

<Page x:Class="CarShop.AddProduct"

xmlns:materialDesign="http://materialdesigninxaml.net/winfx/xaml/themes"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:CarShop"

mc:Ignorable="d"

d:DesignHeight="600" d:DesignWidth="1400"

Background="White"

Title="AddProduct">

<Page.Resources>

<Style TargetType="TextBox">

<Setter Property="FontSize" Value="16"/>

<Setter Property="Height" Value="35"/>

<Setter Property="VerticalContentAlignment" Value="Center"/>

<Setter Property="Padding" Value="100,0,1,0"/>

<Setter Property="Foreground" Value="Gray"/>

</Style>

</Page.Resources>

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition/>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<Button x:Name="closeButton"

FontSize="16"

Click="closeButton\_Click"

Margin="0,30,40,0"

VerticalAlignment="Top"

HorizontalAlignment="Right"

Width="200"

Grid.Column="1"

Content="Назад"/>

<StackPanel Margin="70,0,80,60" VerticalAlignment="Top" Grid.Column="0">

<TextBlock

Margin="0,30,140,0"

TextWrapping="Wrap"

FontWeight="Bold"

FontSize="30"

Text="Заполните информацию о товаре"/>

<TextBox x:Name="nameProduct"

Margin="0,20,0,0"

Padding="30,0,0,0"

MaxLength="70"

GotFocus="nameProduct\_GotFocus"

LostFocus="nameProduct\_LostFocus"

Text="Название товара"/>

<ComboBox x:Name="categoryList"

Height="35"

Margin="0,20,0,0"

SelectionChanged="categoryList\_SelectionChanged"

VerticalContentAlignment="Center"

Padding="10,0,10,0"

BorderThickness="0.5"

BorderBrush="Gray"

DisplayMemberPath="Name"

Background="White"

Foreground="Black"

FontSize="16"

materialDesign:HintAssist.Hint="Категория" >

</ComboBox>

<ComboBox x:Name="subcategoryList"

Height="35"

VerticalContentAlignment="Center"

Padding="10,0,10,0"

Margin="0,20,0,0"

BorderThickness="0.5"

BorderBrush="Gray"

DisplayMemberPath="Name"

Background="White"

Foreground="Black"

FontSize="16"

materialDesign:HintAssist.Hint="Подкатегория" >

</ComboBox>

<TextBlock x:Name="WarName"

Visibility="Hidden"

Text="Поле не заполнено"

Foreground="Red"

FontSize="11"/>

<Grid Margin="0,5,0,0">

<Grid.ColumnDefinitions>

<ColumnDefinition/>

<ColumnDefinition Width="50"/>

</Grid.ColumnDefinitions>

<ComboBox x:Name="BrandName"

Height="35"

VerticalContentAlignment="Center"

Padding="10,0,10,0"

BorderThickness="0.5"

BorderBrush="Gray"

DisplayMemberPath="Name"

Background="White"

Foreground="Black"

FontSize="16"

materialDesign:HintAssist.Hint="Производитель" >

</ComboBox>

<Button Click="AddNewBrand\_Button"

Padding="0"

Margin="15,0,0,0"

Height="35"

Width="35"

Grid.Column="1" >

<Image Source="/Images/addIcons.png"

Height="28"

Width="28" />

</Button>

</Grid>

<TextBlock x:Name="WarBrand"

Visibility="Hidden"

Text="Производитель не выбран"

Foreground="Red"

FontSize="11"/>

<Grid Margin="0,5,0,0">

<Grid.ColumnDefinitions>

<ColumnDefinition/>

<ColumnDefinition Width="50"/>

</Grid.ColumnDefinitions>

<ComboBox x:Name="UnitField"

Height="35"

VerticalContentAlignment="Center"

BorderBrush="Gray"

Padding="10,0,10,0"

BorderThickness="0.5"

Background="White"

Foreground="Black"

DisplayMemberPath="Name"

FontSize="16"

materialDesign:HintAssist.Hint="Единица измерения">

</ComboBox>

<Button Click="AddNewUntir\_Button"

Padding="0"

Margin="15,0,0,0"

Height="35"

Width="35"

Grid.Column="1" >

<Image Source="/Images/addIcons.png"

Height="28"

Width="28" />

</Button>

</Grid>

<TextBlock x:Name="WarUnit"

Visibility="Hidden"

Text="Единица измерения не выбрана не выбран"

Foreground="Red"

FontSize="11"/>

<TextBox x:Name="CountProductField"

Margin="0,5,0,0"

Padding="30,0,0,0"

GotFocus="CountProductField\_GotFocus"

LostFocus="CountProductField\_LostFocus"

Text="Количество" PreviewTextInput="CountProductField\_PreviewTextInput"/>

<TextBox x:Name="PriceField"

Margin="0,20,0,0"

Padding="30,0,0,0"

PreviewTextInput="PriceField\_PreviewTextInput"

GotFocus="PriceField\_GotFocus"

LostFocus="PriceField\_LostFocus"

Text="Цена"/>

<TextBox x:Name="Description"

MinHeight="100"

Height="Auto"

Padding="4"

GotFocus="Discription\_GotFocus"

LostFocus="Discription\_LostFocus"

Text="Подробное описание товара"

TextWrapping="Wrap"

VerticalContentAlignment="Top"

Margin="0,20,0,0"

Width="Auto"

HorizontalAlignment="Stretch"/>

<Button x:Name="GoBackButton"

Click="GoBackButton\_Click"

Margin="0,20,0,0"

FontSize="16"

VerticalAlignment="Center"

HorizontalAlignment="Stretch"

Content="Добавление" />

</StackPanel>

<ScrollViewer Grid.Column="1" Margin="0,100,0,0">

<StackPanel Orientation="Vertical" >

<ListBox x:Name="ImageList" ItemsSource="{Binding Pictures}">

<ListBox.ItemTemplate>

<DataTemplate>

<Image Width="200" Source="{Binding Image}"/>

</DataTemplate>

</ListBox.ItemTemplate>

</ListBox>

<Button Padding="0"

Margin="15,0,40,0"

Height="35"

Click="NewImage\_Click"

Grid.Column="1" >

<StackPanel Orientation="Horizontal">

<Path Data = "M43 38.2222V4.77778C43 2.15 40.85 0 38.2222 0H4.77778C2.15 0 0 2.15 0 4.77778V38.2222C0 40.85 2.15 43 4.77778 43H38.2222C40.85 43 43 40.85 43 38.2222ZM14.0944 26.23L19.1111 32.2739L26.5167 22.7422C26.9944 22.1211 27.95 22.1211 28.4278 22.7661L36.8128 33.9461C37.41 34.7344 36.8367 35.8572 35.8572 35.8572H7.21444C6.21111 35.8572 5.66167 34.7106 6.28278 33.9222L12.2311 26.2778C12.685 25.6567 13.5928 25.6328 14.0944 26.23Z"

Fill="white"

Stretch="Fill"

Height="21"

Width="21" />

<TextBlock Text="Редактировать картинку" Margin="10,0,0,0"/>

</StackPanel>

</Button>

<TextBlock x:Name="WarImage"

Visibility="Hidden"

HorizontalAlignment="Center"

Text="Выбрите минимум 1 картинку"

Foreground="Red"

FontSize="11"/>

</StackPanel>

</ScrollViewer>

</Grid>

</Page>

AddProduct.xaml.cs:

public partial class AddProduct : Page

{

Good goodValue;

public AddProduct()

{

InitializeComponent();

UnitField.ItemsSource = MainWindow.carShopEntities.Units.ToList();

BrandName.ItemsSource = MainWindow.carShopEntities.Brands.ToList();

categoryList.ItemsSource = MainWindow.carShopEntities.Categories.ToList();

}

public AddProduct(Good good) : this()

{

goodValue = good;

Description.Text = good.Description;

Description.Foreground = Brushes.Black;

nameProduct.Text = good.Name;

nameProduct.Foreground = Brushes.Black;

UnitField.SelectedItem = good.Unit;

categoryList.SelectedItem = good.Subcategory.Category;

subcategoryList.SelectedItem = good.Subcategory;

UnitField.Foreground = Brushes.Black;

BrandName.SelectedItem = good.Brand;

BrandName.Foreground = Brushes.Black;

PriceField.Text = Convert.ToString(good.Price);

PriceField.Foreground = Brushes.Black;

CountProductField.Text = Convert.ToString(good.Count);

CountProductField.Foreground = Brushes.Black;

ImageList.ItemsSource = good.Pictures.ToList();

GoBackButton.Content = "Сохранить";

}

List<string> filePaths;

private void NewImage\_Click(object sender, RoutedEventArgs e)

{

OpenFileDialog openDialog = new OpenFileDialog

{

Multiselect = true

};

openDialog.ShowDialog();

filePaths = openDialog.FileNames.ToList();

List<Picture> pictures = new List<Picture>();

foreach (var item in filePaths)

{

pictures.Add(new Picture

{

Image = System.IO.File.ReadAllBytes(item),

GoodId = 1,

});

}

ImageList.ItemsSource = pictures;

}

private void ImageLoad(Good good)

{

foreach (var item in filePaths)

{

MainWindow.carShopEntities.Pictures.Add(new Picture { Image = System.IO.File.ReadAllBytes(item), Good = good });

}

}

private void nameProduct\_GotFocus(object sender, RoutedEventArgs e)

{

if (nameProduct.Text == "Название товара")

{

nameProduct.Text = "";

nameProduct.Foreground = Brushes.Black;

}

}

private void nameProduct\_LostFocus(object sender, RoutedEventArgs e)

{

if (nameProduct.Text == "")

{

nameProduct.Text = "Название товара";

nameProduct.Foreground = Brushes.Gray;

}

}

private void CountProductField\_GotFocus(object sender, RoutedEventArgs e)

{

if (CountProductField.Text == "Количество")

{

CountProductField.Text = "";

CountProductField.Foreground = Brushes.Black;

}

}

private void CountProductField\_LostFocus(object sender, RoutedEventArgs e)

{

if (CountProductField.Text == "")

{

CountProductField.Text = "Количество";

CountProductField.Foreground = Brushes.Gray;

}

}

private void Discription\_GotFocus(object sender, RoutedEventArgs e)

{

if (Description.Text == "Подробное описание товара")

{

Description.Text = "";

Description.Foreground = Brushes.Black;

}

}

private void Discription\_LostFocus(object sender, RoutedEventArgs e)

{

if (Description.Text == "")

{

Description.Text = "Подробное описание товара";

Description.Foreground = Brushes.Gray;

}

}

private void GoBackButton\_Click(object sender, RoutedEventArgs e)

{

if (BrandName.SelectedItem != null && nameProduct.Text != ""

&& categoryList.SelectedItem != null && subcategoryList.SelectedItem != null

&& nameProduct.Text != "Название товара" && CountProductField.Text != ""

&& CountProductField.Text != "Количество" && UnitField.SelectedItem != null

&& Description.Text != "" && Description.Text != "Подробное описание товара"

&& PriceField.Text != "" && PriceField.Text != "Цена")

{

if (ImageList.ItemsSource != null)

{

if (goodValue == null)

{

goodValue = new Good();

}

goodValue.Brand = (Brand)BrandName.SelectedItem;

goodValue.Name = nameProduct.Text;

goodValue.Unit = (Unit)UnitField.SelectedItem;

goodValue.Subcategory = (Subcategory)subcategoryList.SelectedItem;

goodValue.Price = Convert.ToDecimal(PriceField.Text);

goodValue.Count = Convert.ToInt32(CountProductField.Text);

goodValue.Description = Description.Text;

goodValue.IsActive = true;

if(filePaths != null)

{

ImageLoad(goodValue);

}

if (goodValue == null)

{

MainWindow.carShopEntities.Goods.Add(goodValue);

}

MainWindow.carShopEntities.SaveChanges();

this.NavigationService.GoBack();

}

else

{

WarImage.Visibility = Visibility.Visible;

}

}

else

{

MessageBox.Show("Заполните все поля");

}

}

private void CountProductField\_PreviewTextInput(object sender, TextCompositionEventArgs e)

{

Regex intControl = new Regex(@"^([1-9]{1})([0-9]{0,})$");

if (!intControl.IsMatch(CountProductField.Text+e.Text))

{

e.Handled = true;

}

}

private void AddNewBrand\_Button(object sender, RoutedEventArgs e)

{

AddValue newBrand = new AddValue();

if(newBrand.ShowDialog() == true)

{

BrandName.ItemsSource = MainWindow.carShopEntities.Brands.ToList();

}

}

private void AddNewUntir\_Button(object sender, RoutedEventArgs e)

{

AddValue newBrand = new AddValue(false);

if (newBrand.ShowDialog() == true)

{

UnitField.ItemsSource = MainWindow.carShopEntities.Units.ToList();

}

}

private void PriceField\_GotFocus(object sender, RoutedEventArgs e)

{

if(PriceField.Text == "Цена")

{

PriceField.Text = "";

PriceField.Foreground = Brushes.Black;

}

}

private void PriceField\_LostFocus(object sender, RoutedEventArgs e)

{

if (PriceField.Text == "")

{

PriceField.Text = "Цена";

PriceField.Foreground = Brushes.Gray;

}

}

private void PriceField\_PreviewTextInput(object sender, TextCompositionEventArgs e)

{

Regex r = new Regex(@"^((([0]{1})((,)([0-9]{0,}))?)||([1-9]{1}([0-9]{0,})((,)([0-9]{0,}))?))$");

if (!r.IsMatch((PriceField.Text + e.Text)))

{

e.Handled = true;

}

}

private void closeButton\_Click(object sender, RoutedEventArgs e)

{

this.NavigationService.GoBack();

}

private void categoryList\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

subcategoryList.ItemsSource = ((Category)categoryList.SelectedItem).Subcategories.ToList();

}

}

Authorization.xaml:

<Page x:Class="CarShop.Authorization"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:CarShop"

mc:Ignorable="d"

d:DesignHeight="500" d:DesignWidth="1400"

Background="White"

Title="Authorization">

<Page.Resources>

<ResourceDictionary>

<Style TargetType="TextBox" >

<Setter Property="FontSize" Value="17"/>

<Setter Property="Foreground" Value="Gray"/>

<Setter Property="Height" Value="35"/>

<Setter Property="VerticalContentAlignment" Value="Center"/>

</Style>

</ResourceDictionary>

</Page.Resources>

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="490"/>

<ColumnDefinition Width="1\*"/>

<ColumnDefinition Width="490"/>

</Grid.ColumnDefinitions>

<StackPanel Orientation="Vertical"

Grid.Column="1"

Margin="0,80,0,0"

HorizontalAlignment="Stretch">

<TextBlock Text="АВТОРИЗАЦИЯ"

Margin="3,0,0,0"

HorizontalAlignment="Left"

FontSize="23" />

<TextBox x:Name="emailName"

Text="E-mail"

GotFocus="emailName\_GotFocus"

LostFocus="emailName\_LostFocus"

Margin="0,25,0,0">

<TextBox.Resources>

<Style TargetType="{x:Type Border}">

<Setter Property="CornerRadius" Value="1"/>

</Style>

</TextBox.Resources>

</TextBox>

<TextBlock x:Name="emailWar"

Text="Пользователь не найдет"

Visibility="Hidden"

Foreground="Red"

FontSize="11"/>

<Border x:Name="passBorder"

Visibility="Visible"

BorderBrush="Black"

Height="35"

CornerRadius="1"

BorderThickness="0.2"

Margin="0,22,0,0">

<Grid>

<PasswordBox x:Name="passwordfield"

BorderThickness="0"

Visibility="Hidden"

FontSize="15"

Height="25"

Margin="0,2,0,0"

VerticalAlignment="Center"

LostFocus="passwordfield\_LostFocus">

<PasswordBox.Resources>

<Style TargetType="PasswordBox"/>

</PasswordBox.Resources>

</PasswordBox>

<TextBox x:Name="passHelper"

Text="Пароль"

Foreground="Gray"

GotFocus="passwordfield\_GotFocus">

</TextBox>

</Grid>

</Border>

<TextBlock x:Name="PassWar"

Text="Пароль не верный "

Visibility="Hidden"

Foreground="Red"

FontSize="11"/>

<Button Content="ВОЙТИ"

Click="Authorization\_Click"

Height="39"

Margin="0,22,0,0"

Background="MediumSeaGreen"

FontSize="15">

</Button>

<TextBlock x:Name="fieladNullWar"

Text="Заполните все поля"

Visibility="Hidden"

Foreground="Red"

FontSize="11"/>

</StackPanel>

</Grid>

</Page>

Authorization.xaml.cs:

public partial class Authorization : Page

{

MainWindow mainWindow;

public Authorization(MainWindow mainWindow)

{

InitializeComponent();

this.mainWindow = mainWindow;

}

private void emailName\_GotFocus(object sender, RoutedEventArgs e)

{

if(emailName.Text == "E-mail")

{

emailName.Text = "";

emailName.Foreground = Brushes.Black;

emailName.BorderBrush = Brushes.Black;

}

}

private void emailName\_LostFocus(object sender, RoutedEventArgs e)

{

if (emailName.Text == "")

{

emailName.Text = "E-mail";

emailName.Foreground = Brushes.Gray;

emailName.BorderBrush = Brushes.Gray;

}

}

private void passwordfield\_GotFocus(object sender, RoutedEventArgs e)

{

passwordfield.Visibility = Visibility.Visible;

passHelper.Visibility = Visibility.Hidden;

passwordfield.Focus();

}

private void passwordfield\_LostFocus(object sender, RoutedEventArgs e)

{

if (passwordfield.Password == "")

{

passwordfield.Visibility = Visibility.Hidden;

passHelper.Visibility = Visibility.Visible;

}

}

private void Authorization\_Click(object sender, RoutedEventArgs e)

{

PassWar.Visibility = Visibility.Hidden;

emailWar.Visibility = Visibility.Hidden;

if (emailName.Text != "" && emailName.Text != "E-mail")

{

if (passwordfield.Password != "" && passwordfield.Password != "Пароль")

{

DateAuthorization clientInfoAuthorization = (DateAuthorization)MainWindow.carShopEntities.DateAuthorizations.

Where(x => x.Email == emailName.Text).ToList().FirstOrDefault();

if(clientInfoAuthorization != null)

{

if(clientInfoAuthorization.Password == passwordfield.Password)

{

MainWindow.client = clientInfoAuthorization.User;

if (clientInfoAuthorization.User.IsAdmin)

{

this.NavigationService.Navigate(new ListProductAdminView());

mainWindow.AdminPanelVisible();

}

else

{

mainWindow.authAccount.Visibility = Visibility.Hidden;

mainWindow.personalAccount.Visibility = Visibility.Visible;

this.NavigationService.GoBack();

}

}

else

{

PassWar.Visibility = Visibility.Visible;

passBorder.BorderBrush = Brushes.Red;

}

}

else

{

emailWar.Visibility = Visibility.Visible;

emailName.BorderBrush = Brushes.Red;

}

}

else

{

fieladNullWar.Visibility = Visibility.Visible;

passBorder.BorderBrush = Brushes.Red;

}

}

else

{

fieladNullWar.Visibility = Visibility.Visible;

emailName.BorderBrush = Brushes.Red;

}

}

}

Registration.xaml:

<Page x:Class="CarShop.Registration"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:CarShop"

mc:Ignorable="d"

d:DesignHeight="500" d:DesignWidth="1400"

Background="White"

Title="Registration">

<Page.Resources>

<ResourceDictionary>

<Style TargetType="TextBox" >

<Setter Property="FontSize" Value="17"/>

<Setter Property="Foreground" Value="Gray"/>

<Setter Property="Height" Value="35"/>

<Setter Property="VerticalContentAlignment" Value="Center"/>

<Style.Resources>

<Style TargetType="Border">

<Setter Property="CornerRadius" Value="2"/>

</Style>

</Style.Resources>

</Style>

<Style TargetType="PasswordBox">

<Setter Property="VerticalContentAlignment" Value="Center"/>

</Style>

</ResourceDictionary>

</Page.Resources>

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="490"/>

<ColumnDefinition Width="1\*"/>

<ColumnDefinition Width="490"/>

</Grid.ColumnDefinitions>

<StackPanel Orientation="Vertical"

Grid.Column="1"

Margin="0,50,0,0"

HorizontalAlignment="Stretch">

<TextBlock Text="РЕГИСТРАЦИЯ"

Margin="3,0,0,0"

HorizontalAlignment="Left"

FontSize="23" />

<TextBox x:Name="emailName"

Text="E-mail"

GotFocus="emailName\_GotFocus"

LostFocus="emailName\_LostFocus"

Margin="0,15,0,0">

</TextBox>

<TextBlock x:Name="emailWar"

Text="Данная почта уже занята"

Visibility="Hidden"

Foreground="Red"

FontSize="11"/>

<TextBox x:Name="phonefield"

Text="Номер телефона"

MaxLength="11"

GotFocus="phonefield\_GotFocus"

LostFocus="phonefield\_LostFocus"

Margin="0,2,0,0">

</TextBox>

<Grid Margin="0,17,0,0">

<TextBox x:Name="passwordfield"

Text="Пароль"

BorderBrush="Gray"

GotFocus="passwordfield\_GotFocus">

</TextBox>

<PasswordBox x:Name="password2"

BorderThickness="1"

Visibility="Hidden"

LostFocus="password2\_LostFocus"

Foreground="Black"

FontSize="17"/>

</Grid>

<Grid Margin="0,17,0,0">

<TextBox x:Name="passwordfield2"

Text="Повторите пароль"

BorderBrush="Gray"

GotFocus="passwordfield2\_GotFocus">

</TextBox>

<PasswordBox x:Name="password1"

BorderThickness="1"

Visibility="Hidden"

LostFocus="password1\_LostFocus"

Foreground="Black"

FontSize="17"/>

</Grid>

<TextBlock x:Name="WarPass"

Visibility="Hidden"

Text="Пароли не совподает"

Foreground="Red"

FontSize="11"/>

<TextBox x:Name="surnamefield"

Text="Фамилия"

GotFocus="surnamefield\_GotFocus"

LostFocus="surnamefield\_LostFocus"

Margin="0,5,0,0">

</TextBox>

<TextBox x:Name="namefield"

Text="Имя"

GotFocus="namefield\_GotFocus"

LostFocus="namefield\_LostFocus"

Margin="0,17,0,0">

</TextBox>

<TextBox x:Name="middlenamefield"

Text="Отчество"

GotFocus="middlenamefield\_GotFocus"

LostFocus="middlenamefield\_LostFocus"

Margin="0,17,0,0">

</TextBox>

<Button Content="РЕГИСТРАЦИЯ"

Height="39"

Click="Registration\_Click"

Margin="0,30,0,0"

Background="MediumSeaGreen"

FontSize="15">

</Button>

<TextBlock x:Name="fieladNullWar"

Text="Заполните все поля"

Visibility="Hidden"

Foreground="Red"

FontSize="11"/>

</StackPanel>

</Grid>

</Page>

Registration.xaml.cs:

public partial class Registration : Page

{

MainWindow mainWindow;

public Registration(MainWindow mainWindow)

{

InitializeComponent();

this.mainWindow = mainWindow;

}

private void emailName\_GotFocus(object sender, RoutedEventArgs e)

{

if (emailName.Text == "E-mail")

{

emailName.Text = "";

emailName.Foreground = Brushes.Black;

emailName.BorderBrush = Brushes.Black;

}

}

private void emailName\_LostFocus(object sender, RoutedEventArgs e)

{

if (emailName.Text == "")

{

emailName.Text = "E-mail";

emailName.Foreground = Brushes.Gray;

emailName.BorderBrush = Brushes.Gray;

}

}

private void phonefield\_GotFocus(object sender, RoutedEventArgs e)

{

if (phonefield.Text == "Номер телефона")

{

phonefield.Text = "";

phonefield.Foreground = Brushes.Black;

phonefield.BorderBrush = Brushes.Black;

}

}

private void phonefield\_LostFocus(object sender, RoutedEventArgs e)

{

if (phonefield.Text == "")

{

phonefield.Text = "Номер телефона";

phonefield.Foreground = Brushes.Gray;

phonefield.BorderBrush = Brushes.Gray;

}

}

private void surnamefield\_GotFocus(object sender, RoutedEventArgs e)

{

if (surnamefield.Text == "Фамилия")

{

surnamefield.Text = "";

surnamefield.Foreground = Brushes.Black;

surnamefield.BorderBrush = Brushes.Black;

}

}

private void surnamefield\_LostFocus(object sender, RoutedEventArgs e)

{

if (surnamefield.Text == "")

{

surnamefield.Text = "Фамилия";

surnamefield.Foreground = Brushes.Gray;

surnamefield.BorderBrush = Brushes.Gray;

}

}

private void namefield\_GotFocus(object sender, RoutedEventArgs e)

{

if (namefield.Text == "Имя")

{

namefield.Text = "";

namefield.Foreground = Brushes.Black;

namefield.BorderBrush = Brushes.Black;

}

}

private void namefield\_LostFocus(object sender, RoutedEventArgs e)

{

if (namefield.Text == "")

{

namefield.Text = "Имя";

namefield.Foreground = Brushes.Gray;

namefield.BorderBrush = Brushes.Gray;

}

}

private void middlenamefield\_GotFocus(object sender, RoutedEventArgs e)

{

if (middlenamefield.Text == "Отчество")

{

middlenamefield.Text = "";

middlenamefield.Foreground = Brushes.Black;

middlenamefield.BorderBrush = Brushes.Black;

}

}

private void middlenamefield\_LostFocus(object sender, RoutedEventArgs e)

{

if (middlenamefield.Text == "")

{

middlenamefield.Text = "Отчество";

middlenamefield.Foreground = Brushes.Gray;

middlenamefield.BorderBrush = Brushes.Gray;

}

}

private void passwordfield\_GotFocus(object sender, RoutedEventArgs e)

{

if (passwordfield.Text == "Пароль")

{

password2.Visibility = Visibility.Visible;

passwordfield.Visibility = Visibility.Hidden;

password2.Focus();

}

}

private void password2\_LostFocus(object sender, RoutedEventArgs e)

{

if (password2.Password == "")

{

password2.Visibility = Visibility.Hidden;

passwordfield.Visibility = Visibility.Visible;

}

}

private void passwordfield2\_GotFocus(object sender, RoutedEventArgs e)

{

password1.Visibility = Visibility.Visible;

passwordfield2.Visibility = Visibility.Hidden;

password1.Focus();

}

private void password1\_LostFocus(object sender, RoutedEventArgs e)

{

if (password1.Password == "")

{

password1.Visibility = Visibility.Hidden;

passwordfield2.Visibility = Visibility.Visible;

}

}

private void Registration\_Click(object sender, RoutedEventArgs e)

{

fieladNullWar.Visibility = Visibility.Hidden;

emailWar.Visibility = Visibility.Hidden;

emailName.BorderBrush = Brushes.Black;

password1.BorderBrush = Brushes.Gray;

password2.BorderBrush = Brushes.Gray;

if (emailName.Text != "" && emailName.Text != "E-mail" &&

phonefield.Text != "" && phonefield.Text != "Номер телефона" &&

password1.Password != "" && password2.Password != "" &&

surnamefield.Text != "" && surnamefield.Text != "Фамилия" &&

namefield.Text != "" && namefield.Text != "Имя" &&

middlenamefield.Text != "" && middlenamefield.Text != "Отчество")

{

DateAuthorization clientInfoAuth = MainWindow.carShopEntities.DateAuthorizations.

Where(x => x.Email == emailName.Text).ToList().FirstOrDefault();

Regex phoneRegex = new Regex(@"^(8)[0-9]{10}$");

if (!phoneRegex.IsMatch(phonefield.Text))

{

MessageBox.Show("Номер телефона введён не корректно");

return;

}

Regex emailControl = new Regex("[a-z]{1,20}[0-9]{0,10}(@)(gmail.com || @mail.ru || bk.ru || yandex.ru ||outlook.com)");

if (clientInfoAuth == null)

{

if (emailControl.IsMatch(emailName.Text))

{

if (password1.Password == password2.Password && password2.Password != "")

{

User newClient = new User

{

Surname = surnamefield.Text,

Name = namefield.Text,

MiddleName = namefield.Text,

Phone = phonefield.Text,

DateAuthorizations = new List<DateAuthorization>

{

new DateAuthorization

{

Email = emailName.Text,

Password = password1.Password

}

}

};

MainWindow.carShopEntities.Users.Add(newClient);

MainWindow.carShopEntities.SaveChanges();

MainWindow.client = newClient;

this.NavigationService.Navigate(new MainRecommendationProduct());

mainWindow.authAccount.Visibility = Visibility.Hidden;

mainWindow.personalAccount.Visibility = Visibility.Visible;

}

else

{

WarPass.Visibility = Visibility.Visible;

password2.BorderBrush = Brushes.Red;

password1.BorderBrush = Brushes.Red;

}

}

else

{

emailWar.Visibility = Visibility.Visible;

emailName.BorderBrush = Brushes.Red;

emailWar.Text = "Некорректный ввод эл. почты";

}

}

else

{

emailWar.Visibility = Visibility.Visible;

emailName.BorderBrush = Brushes.Red;

}

}

else

{

fieladNullWar.Visibility = Visibility.Visible;

}

}

}

MyShoppingCart.xaml:

<Page x:Class="CarShop.MyShoppingСart"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:CarShop"

mc:Ignorable="d"

Background="White"

Loaded="Page\_Loaded"

d:DesignHeight="700" d:DesignWidth="1000"

Title="MyShoppingСart">

<Page.Resources>

<ResourceDictionary>

<Style TargetType="TextBlock">

<Setter Property="FontSize" Value="15"/>

<Setter Property="Height" Value="20"/>

<Setter Property="VerticalAlignment" Value="Center"/>

<Setter Property="Foreground" Value="#0000CD"/>

</Style>

<Style TargetType="ListViewItem">

<Setter Property="HorizontalContentAlignment" Value="Center" />

<Setter Property="Foreground" Value="#212121"/>

<Setter Property="Height" Value="30"/>

</Style>

<Style TargetType="Button">

<Setter Property="Background" Value="#059bff"/>

<Setter Property="BorderBrush" Value="#059bff"/>

<Setter Property="Foreground" Value="White"/>

<Setter Property="FontSize" Value="16"/>

<Setter Property="Height" Value="35"/>

<Setter Property="Width" Value="110"/>

</Style>

<local:GoodsConverter x:Key="myPriceGoodConverter"></local:GoodsConverter>

<local:TotalPriceConverter x:Key="myTotalPriceConverter"></local:TotalPriceConverter>

<local:DeliveryConverter x:Key="myDeliveryConverter"></local:DeliveryConverter>

</ResourceDictionary>

</Page.Resources>

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="470"/>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="70"/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock x:Name="infoChapter"

Text="ОБЩАЯ ИНФОРМАЦИЯ"

Grid.Row="0"

Grid.ColumnSpan="2"

Height="30"

Margin="310,0,0,0"

HorizontalAlignment="Left"

VerticalAlignment="Center"

Foreground="Black"

FontSize="20"

/>

<ListBox Grid.Row="1"

BorderThickness="0,0,2,0"

BorderBrush="#F5F5F5"

Height="230"

VerticalAlignment="Top"

Margin="0,0,0,0"

HorizontalAlignment="Right"

Grid.Column="0" Width="170">

<Border x:Name="InfoClientButton"

MouseLeftButtonDown="InfoClient\_Visible"

Width="200">

<TextBlock Text="Общая информация" />

</Border>

<Border x:Name="CartButton"

MouseLeftButtonDown="Cart\_Visible"

Width="200">

<TextBlock Text="Корзина"/>

</Border>

<Border x:Name="OrderButton"

MouseLeftButtonDown="Order\_Visible"

Width="200">

<TextBlock Text="Заказы"/>

</Border>

<Border x:Name="passUpdateButton"

MouseLeftButtonDown="Border\_MouseLeftButtonDown"

Width="200">

<TextBlock Text="Смена пароля"/>

</Border>

</ListBox>

<Grid x:Name="InfoClient"

Grid.Row="1"

Visibility="Hidden"

Margin="30,0,0,0"

Grid.Column="1">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="170"/>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="40"/>

<RowDefinition Height="40"/>

<RowDefinition Height="40"/>

<RowDefinition Height="40"/>

<RowDefinition Height="40"/>

<RowDefinition Height="80"/>

</Grid.RowDefinitions>

<Grid.Resources>

<Style TargetType="TextBlock" >

<Setter Property="VerticalAlignment" Value="Center"/>

<Setter Property="FontSize" Value="16"/>

<Setter Property="Foreground" Value="Black"/>

</Style>

<Style TargetType="TextBox" >

<Setter Property="FontSize" Value="15"/>

<Setter Property="Height" Value="28"/>

<Setter Property="IsEnabled" Value="False"/>

<Setter Property="Width" Value="380"/>

<Setter Property="VerticalAlignment" Value="Center"/>

<Setter Property="HorizontalAlignment" Value="Left"/>

<Setter Property="Foreground" Value="Black"/>

</Style>

</Grid.Resources>

<TextBlock Text="Эл. почта" Grid.Row="0"/>

<TextBlock Text="Номер телефона" Grid.Row="1"/>

<TextBlock Text="Фамилия" Grid.Row="2"/>

<TextBlock Text="Имя" Grid.Row="3"/>

<TextBlock Text="Отчество" Grid.Row="4"/>

<TextBox x:Name="email"

Text="{Binding Email}"

Grid.Row="0" Grid.Column="1"/>

<TextBox x:Name="phone"

Text="{Binding User.Phone}"

MaxLength="11"

PreviewTextInput="phone\_PreviewTextInput"

Grid.Row="1" Grid.Column="1"/>

<TextBox x:Name="surname"

Text="{Binding User.Surname}"

Grid.Row="2" Grid.Column="1"/>

<TextBox x:Name="name"

Text="{Binding User.Name}"

Grid.Row="3" Grid.Column="1"/>

<TextBox x:Name="middleName"

Text="{Binding User.MiddleName}"

Grid.Row="4" Grid.Column="1"/>

<Button x:Name="updateButton"

Content="Изменить"

Grid.Row="5"

Click="updateButton\_Click"

Margin="270,5,0,0"

HorizontalAlignment="Left"

Grid.Column="1"/>

<Button x:Name="cancleButton"

Content="Отмена"

Visibility="Hidden"

Click="cancleButton\_Click"

Margin="270,5,0,0"

HorizontalAlignment="Left"

Grid.Row="5"

Grid.Column="1"/>

<Button x:Name="saveButton"

Content="Сохранить"

Grid.Row="5"

Click="saveButton\_Click"

Visibility="Hidden"

Margin="0,5,0,0"

HorizontalAlignment="Left"

Grid.Column="0"/>

</Grid>

<Grid x:Name="Cart"

Grid.Row="1"

Visibility="Hidden"

Margin="10,0,190,0"

Grid.Column="1">

<StackPanel Orientation="Vertical">

<ListView x:Name="CartList"

Width="1000"

ItemsSource="{Binding GoodSales}"

HorizontalAlignment="Left">

<ListView.Resources>

<Style TargetType="{x:Type GridViewColumnHeader}">

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="{x:Type GridViewColumnHeader}">

<Border BorderThickness="0,0,0,2" BorderBrush="#99DAFF" Background="Transparent">

<Label x:Name="ContentHeader" Content="{TemplateBinding Content}" Padding="14,5,5,12"

Width="{TemplateBinding Width}" HorizontalAlignment="Left" />

</Border>

</ControlTemplate>

</Setter.Value>

</Setter>

</Style>

</ListView.Resources>

<ListView.View>

<GridView>

<GridViewColumn Header="Название" DisplayMemberBinding="{Binding Good.Name}" Width="300"></GridViewColumn>

<GridViewColumn Header="Цена" Width="100">

<GridViewColumn.CellTemplate>

<DataTemplate>

<StackPanel Orientation="Horizontal">

<TextBlock Text="{Binding Good.Price, StringFormat={}{0:#.}}" Foreground="Blue" FontSize="15"/>

<TextBlock Text=" руб." Foreground="Blue" FontSize="15"/>

</StackPanel>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

<GridViewColumn Header="Количество" Width="140">

<GridViewColumn.CellTemplate>

<DataTemplate>

<StackPanel Orientation="Horizontal" >

<TextBox x:Name="Count"

Height="25"

Width="30"

LostFocus="Count\_LostFocus"

PreviewTextInput="Count\_PreviewTextInput"

VerticalContentAlignment="Center"

HorizontalContentAlignment="Center"

Text="{Binding Count}">

<TextBox.Style>

<Style TargetType="TextBox"/>

</TextBox.Style>

</TextBox>

</StackPanel>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

<GridViewColumn Header="Общая цена"

DisplayMemberBinding="{Binding Converter={StaticResource myPriceGoodConverter}, StringFormat={}{0:#.} руб.}"

Width="160">

</GridViewColumn>

</GridView>

</ListView.View>

</ListView>

<TextBlock x:Name="SumCost"

Margin="0,20,0,0"

HorizontalAlignment="Right"

FontSize="15"

Text="{Binding Converter={StaticResource myTotalPriceConverter}, StringFormat={}Итоговая цена {0} руб.}"/>

<DockPanel>

<StackPanel Orientation="Horizontal"

Margin="20,10,0,0">

<Button Content="Удалить" Click="Cart\_Deleted"/>

<Button Content="Удалить всё"

Click="Cart\_FullDeleted"

Margin="20,0,0,0"/>

</StackPanel>

<Button Content="Оформить" Click="Arrange\_Click" HorizontalAlignment="Right" Margin="0,10,0,0"/>

</DockPanel>

</StackPanel>

</Grid>

<Grid x:Name="Order"

Grid.Row="1"

Visibility="Hidden"

Grid.Column="1">

<ListBox x:Name="ListOrders"

HorizontalAlignment="Left"

Margin="40,0,0,0"

Width="700"

VerticalAlignment="Top">

<ListBox.ItemTemplate>

<DataTemplate>

<Grid MinHeight="115">

<Border>

<Grid HorizontalAlignment="Right"

Background="#F5F5F5"

Width="680" >

<Grid.RowDefinitions>

<RowDefinition Height="35"/>

<RowDefinition Height="1\*"/>

</Grid.RowDefinitions>

<StackPanel Orientation="Horizontal"

Margin="20,10,0,5"

Grid.Row="0"

VerticalAlignment="Top">

<TextBlock Text="Заказ №"

Padding="0"

Margin="0,0,0,0"

Foreground="#000080"

FontSize="18"/>

<TextBlock Text="{Binding Id}"

Foreground="#000080"

VerticalAlignment="Bottom"

Margin="4,0,0,-3"

FontSize="18"/>

<TextBlock Text="от "

Foreground="Gray"

Margin="10,6,0,0" />

<TextBlock Text="{Binding Date , StringFormat={}{0:dd.MM.yy}}"

Foreground="Gray"

Margin="0,6,0,0"/>

</StackPanel>

<StackPanel Orientation="Horizontal"

VerticalAlignment="Top"

Margin="0,13,20,0"

Grid.Row="0"

HorizontalAlignment="Right">

<TextBlock Text="{Binding IsDelivered, Converter={StaticResource myDeliveryConverter}}"

Foreground="Gray" />

<TextBlock Text="{Binding Converter={StaticResource myTotalPriceConverter}}"

Margin="10,0,00,0"

Foreground="Gray"/>

<TextBlock Text="руб."

Margin="4,0,00,0"

Foreground="Gray"/>

</StackPanel>

<Expander Grid.Row="1" Background="#F5F5F5">

<ListView x:Name="ListPastOrders"

ItemsSource ="{Binding GoodSales}"

Width="680"

VerticalContentAlignment="Center"

Grid.Row="1"

HorizontalAlignment="Left">

<ListView.View>

<GridView>

<GridViewColumn Header="Название"

Width="200">

<GridViewColumn.CellTemplate>

<DataTemplate>

<TextBlock Text ="{Binding Good.Name}" />

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

<GridViewColumn Header="Цена" Width="100">

<GridViewColumn.CellTemplate>

<DataTemplate>

<Grid Width="90">

<StackPanel Orientation="Horizontal" HorizontalAlignment="Center" >

<TextBlock Text ="{Binding Good.Price, StringFormat={}{0:#.}}" />

<TextBlock Text =" руб." />

</StackPanel>

</Grid>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

<GridViewColumn Header="Количество"

Width="140">

<GridViewColumn.CellTemplate>

<DataTemplate>

<Grid Width="120">

<TextBlock Text ="{Binding Count}"

HorizontalAlignment="Center" />

</Grid>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

<GridViewColumn Header="Общая цена"

Width="160" >

<GridViewColumn.CellTemplate>

<DataTemplate>

<Grid Width="150">

<StackPanel Orientation="Horizontal"

HorizontalAlignment="Center">

<TextBlock Text ="{Binding StringFormat={}{0:#.},

Converter={StaticResource myPriceGoodConverter}}" />

<TextBlock Text =" руб." />

</StackPanel>

</Grid>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

</GridView>

</ListView.View>

</ListView>

</Expander>

</Grid>

</Border>

</Grid>

</DataTemplate>

</ListBox.ItemTemplate>

</ListBox>

</Grid>

<Grid x:Name="OrderNull"

Grid.Row="1"

Visibility="Hidden"

Grid.Column="1">

<TextBlock Text="Истории заказов пуста"

FontSize="23"

x:Name="OrderText"

Margin="0,100,0,60"

Height="30"

VerticalAlignment="Top"

HorizontalAlignment="Center"

Foreground="Black"/>

</Grid>

<Grid x:Name="UpdatePassword"

Grid.Row="1"

Visibility="Hidden"

Grid.Column="1">

<StackPanel Orientation="Vertical"

Margin="40,20,0,0"

HorizontalAlignment="Left"

VerticalAlignment="Top">

<StackPanel.Resources>

<Style TargetType="TextBox">

<Setter Property="Width" Value="300"/>

<Setter Property="Height" Value="30"/>

<Setter Property="BorderBrush" Value="Gray"/>

<Setter Property="Foreground" Value="Gray"/>

<Setter Property="FontSize" Value="16"/>

<Setter Property="VerticalContentAlignment" Value="Center"/>

</Style>

<Style TargetType="PasswordBox">

<Setter Property="Width" Value="300"/>

<Setter Property="Height" Value="30"/>

<Setter Property="BorderBrush" Value="Black"/>

<Setter Property="Foreground" Value="Black"/>

<Setter Property="FontSize" Value="16"/>

<Setter Property="VerticalContentAlignment" Value="Center"/>

</Style>

</StackPanel.Resources>

<Grid Margin="0,17,0,0">

<TextBox x:Name="passwordfield1"

Text="Старый пароль"

GotFocus="passwordfield\_GotFocus">

</TextBox>

<PasswordBox x:Name="passwordPast"

BorderThickness="1"

Visibility="Hidden"

LostFocus="passwordPasr\_LostFocus"/>

</Grid>

<TextBlock x:Name="passPastWar"

Text="Неверный старый пароль"

Visibility="Hidden"

Foreground="Red"

FontSize="11"/>

<Grid Margin="0,13,0,0">

<TextBox x:Name="passwordfield2"

Text="Новый пароль"

GotFocus="passwordfield2\_GotFocus">

</TextBox>

<PasswordBox x:Name="passwordNew"

BorderThickness="1"

Visibility="Hidden"

LostFocus="passwordNew\_LostFocus"/>

</Grid>

<TextBlock x:Name="passNewWar"

Text="Минимальная длинна 6 символов"

Visibility="Hidden"

Foreground="Red"

FontSize="11"/>

<Button Content="Изменить"

Margin="0,13,0,0"

Width="100"

Click="Button\_Update"

HorizontalAlignment="Right"/>

</StackPanel>

</Grid>

</Grid>

</Page>

MyShoppingCart.xaml.cs:

public partial class MyShoppingСart : Page

{

private DateAuthorization client;

public MyShoppingСart()

{

InitializeComponent();

}

private void Page\_Loaded(object sender, RoutedEventArgs e)

{

if (MainWindow.client != null)

{

client = (DateAuthorization)MainWindow.client.DateAuthorizations.FirstOrDefault().Clone();

DataContext = client;

ListOrders.ItemsSource = MainWindow.client.Sales.OrderByDescending(x=>x.Date).ToList();

InfoClientButton.IsEnabled = true;

OrderButton.IsEnabled = true;

passUpdateButton.IsEnabled = true;

}

else

{

InfoClientButton.IsEnabled = false;

OrderButton.IsEnabled = false;

passUpdateButton.IsEnabled = false;

}

Cart.DataContext = new Sale { GoodSales = MainWindow.Instance.GoodCart.ToList() };

}

public MyShoppingСart(bool CartVisible):this()

{

InitializeComponent();

if (MainWindow.Instance.GoodCart.Count != 0)

{

Cart.Visibility = Visibility.Visible;

}

else

{

OrderText.Text = "Корзина пуста";

OrderNull.Visibility = Visibility.Visible;

}

}

private void Cart\_Visible(object sender, MouseButtonEventArgs e)

{

InfoClient.Visibility = Visibility.Hidden;

Order.Visibility = Visibility.Hidden;

UpdatePassword.Visibility = Visibility.Hidden;

OrderNull.Visibility = Visibility.Hidden;

Cart.Visibility = Visibility.Hidden;

if (MainWindow.Instance.GoodCart.Count != 0)

{

Cart.Visibility = Visibility.Visible;

}

else

{

OrderText.Text = "Корзина пуста";

OrderNull.Visibility = Visibility.Visible;

}

}

private void InfoClient\_Visible(object sender, MouseButtonEventArgs e)

{

Cart.Visibility = Visibility.Hidden;

Order.Visibility = Visibility.Hidden;

InfoClient.Visibility = Visibility.Visible;

UpdatePassword.Visibility = Visibility.Hidden;

OrderNull.Visibility = Visibility.Hidden;

}

private void Order\_Visible(object sender, MouseButtonEventArgs e)

{

Cart.Visibility = Visibility.Hidden;

InfoClient.Visibility = Visibility.Hidden;

UpdatePassword.Visibility = Visibility.Hidden;

OrderNull.Visibility = Visibility.Hidden;

Order.Visibility = Visibility.Hidden;

if (MainWindow.client.Sales.Count != 0)

{

Order.Visibility = Visibility.Visible;

}

else

{

OrderNull.Visibility = Visibility.Visible;

OrderText.Text = "Истории заказов пуста";

}

}

private void Border\_MouseLeftButtonDown(object sender, MouseButtonEventArgs e)

{

Cart.Visibility = Visibility.Hidden;

Order.Visibility = Visibility.Hidden;

InfoClient.Visibility = Visibility.Hidden;

UpdatePassword.Visibility = Visibility.Visible;

OrderNull.Visibility = Visibility.Hidden;

}

private void passwordfield\_GotFocus(object sender, RoutedEventArgs e)

{

if (passwordfield1.Text == "Старый пароль")

{

passwordPast.Visibility = Visibility.Visible;

passwordfield1.Visibility = Visibility.Hidden;

passwordPast.Focus();

}

}

private void passwordPasr\_LostFocus(object sender, RoutedEventArgs e)

{

if (passwordPast.Password == "")

{

passwordPast.Visibility = Visibility.Hidden;

passwordfield1.Visibility = Visibility.Visible;

}

}

private void passwordfield2\_GotFocus(object sender, RoutedEventArgs e)

{

if (passwordfield2.Text == "Новый пароль")

{

passwordNew.Visibility = Visibility.Visible;

passwordfield2.Visibility = Visibility.Hidden;

passwordNew.Focus();

}

}

private void passwordNew\_LostFocus(object sender, RoutedEventArgs e)

{

if (passwordNew.Password == "")

{

passwordNew.Visibility = Visibility.Hidden;

passwordfield2.Visibility = Visibility.Visible;

}

}

private void Button\_Update(object sender, RoutedEventArgs e)

{

passPastWar.Visibility = Visibility.Hidden;

passNewWar.Visibility = Visibility.Hidden;

if (passwordPast.Password == MainWindow.client.DateAuthorizations.First().Password)

{

if(passwordNew.Password.Length >= 6)

{

MainWindow.client.DateAuthorizations.First().Password = passwordNew.Password;

MainWindow.carShopEntities.SaveChanges();

MessageBox.Show("Пароль успешно изменён");

passwordNew.Password = "";

passwordPast.Password = "";

passwordNew.Visibility = Visibility.Hidden;

passwordPast.Visibility = Visibility.Hidden;

passwordfield1.Visibility = Visibility.Visible;

passwordfield2.Visibility = Visibility.Visible;

}

else

{

passNewWar.Visibility = Visibility.Visible;

}

}

else

{

passPastWar.Visibility = Visibility.Visible;

}

}

private void updateButton\_Click(object sender, RoutedEventArgs e)

{

email.IsEnabled = true;

phone.IsEnabled = true;

surname.IsEnabled = true;

name.IsEnabled = true;

middleName.IsEnabled = true;

cancleButton.Visibility = Visibility.Visible;

saveButton.Visibility = Visibility.Visible;

}

private void VisibleFields()

{

email.IsEnabled = false;

phone.IsEnabled = false;

surname.IsEnabled = false;

name.IsEnabled = false;

middleName.IsEnabled = false;

cancleButton.Visibility = Visibility.Hidden;

saveButton.Visibility = Visibility.Hidden;

updateButton.Visibility = Visibility.Visible;

}

private void cancleButton\_Click(object sender, RoutedEventArgs e)

{

client = (DateAuthorization)MainWindow.client.DateAuthorizations.FirstOrDefault().Clone();

DataContext = client;

VisibleFields();

}

private void saveButton\_Click(object sender, RoutedEventArgs e)

{

if (email.Text != "" && phone.Text != "" &&

surname.Text != "" && name.Text != "" && middleName.Text != "")

{

Regex r = new Regex(@"^(8)[0-9]{10}$");

if (!r.IsMatch(phone.Text))

{

MessageBox.Show("Номер телефона не корректен");

return;

}

Regex emailControl = new Regex("[a-z]{1,20}[0-9]{0,}(@)(gmail.com || @mail.ru || bk.ru || yandex.ru ||outlook.com)");

if (emailControl.IsMatch(email.Text))

{

MessageBox.Show("Почта введена не корректно");

return;

}

VisibleFields();

MainWindow.client.Surname = client.User.Surname;

MainWindow.client.Name = client.User.Name;

MainWindow.client.MiddleName = client.User.MiddleName;

MainWindow.client.Phone = client.User.Phone;

MainWindow.client.DateAuthorizations.First().Email = client.Email;

MainWindow.carShopEntities.SaveChanges();

}

else

{

MessageBox.Show("Поля не могут быть пустыми");

}

}

private void Count\_PreviewTextInput(object sender, TextCompositionEventArgs e)

{

Regex r = new Regex(@"^[1-9]{1}[0-9]{0,}$");

if (!r.IsMatch(((TextBox)sender).Text + e.Text))

{

e.Handled = true;

}

}

private void Cart\_FullDeleted(object sender, RoutedEventArgs e)

{

MainWindow.Instance.GoodCart = new List<GoodSale>();

Cart.DataContext = new Sale { GoodSales = MainWindow.Instance.GoodCart.ToList() };

}

private void Cart\_Deleted(object sender, RoutedEventArgs e)

{

if (CartList.SelectedItems.Count > 0)

{

var gsValue = CartList.SelectedItems;

foreach (var item in gsValue)

{

MainWindow.Instance.GoodCart.Remove((GoodSale)item);

}

Cart.DataContext = new Sale { GoodSales = MainWindow.Instance.GoodCart};

MainWindow.Instance.GoodCart = MainWindow.Instance.GoodCart;

if (MainWindow.Instance.GoodCart.Count <= 0)

{

Cart.Visibility = Visibility.Hidden;

OrderText.Text = "Корзина пуста";

OrderNull.Visibility = Visibility.Visible;

}

}

}

private void Arrange\_Click(object sender, RoutedEventArgs e)

{

if (MainWindow.client != null)

{

Sale SaleNew = new Sale();

SaleNew.Date = DateTime.Now;

SaleNew.UserId = MainWindow.client.Id;

SaleNew.IsDelivered = false;

foreach (var item in MainWindow.Instance.GoodCart)

{

if (item.Good.Count >= item.Count)

{

item.Sale = SaleNew;

item.SaleId = SaleNew.Id;

}

else

{

MessageBox.Show($"{item.Good.Name} имеется в налиxии только {item.Good.Count} шт");

return;

}

}

SaleNew.GoodSales = MainWindow.Instance.GoodCart;

MainWindow.carShopEntities.Sales.Add(SaleNew);

foreach (var item in MainWindow.Instance.GoodCart)

{

Good g = MainWindow.carShopEntities.Goods.Where(x => x.Id == item.GoodId).ToList().FirstOrDefault();

g.Count = g.Count - item.Count;

}

MainWindow.carShopEntities.SaveChanges();

MainWindow.Instance.GoodCart = new List<GoodSale>();

Cart.DataContext = new Sale { GoodSales = MainWindow.Instance.GoodCart.ToList() };

MainWindow.Instance.GoodCart = new List<GoodSale>();

Cart.Visibility = Visibility.Hidden;

OrderNull.Visibility = Visibility.Visible;

MessageBox.Show("Заказ успешно оформлен");

OrderText.Text = "Корзина пуста";

ListOrders.ItemsSource = MainWindow.client.Sales.OrderByDescending(x=>x.Date).ToList();

}

else

{

this.NavigationService.Navigate(new Authorization(MainWindow.Instance));

}

}

private void Count\_LostFocus(object sender, RoutedEventArgs e)

{

if(((TextBox)sender).Text == "")

{

((TextBox)sender).Text = "0";

}

MainWindow.Instance.GoodCart = MainWindow.Instance.GoodCart;

Cart.DataContext = new Sale { GoodSales = MainWindow.Instance.GoodCart };

}

private void phone\_PreviewTextInput(object sender, TextCompositionEventArgs e)

{

Regex r = new Regex(@"^(8)[0-9]{0,}$");

if(!r.IsMatch(phone.Text + e.Text))

{

e.Handled = true;

}

}

}

MainWindow.xaml:

<Window x:Class="CarShop.MainWindow"

xmlns:materialDesign="http://materialdesigninxaml.net/winfx/xaml/themes"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:CarShop"

mc:Ignorable="d"

WindowStartupLocation="CenterScreen"

Title="MainWindow" Height="700" Width="1400">

<Window.Resources>

<ResourceDictionary>

<Style x:Key="TextStyle"

TargetType="TextBlock">

<Setter Property="FontSize"

Value="14.5"/>

</Style>

</ResourceDictionary>

</Window.Resources>

<ScrollViewer VerticalScrollBarVisibility="Hidden"

HorizontalScrollBarVisibility="Disabled">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="90"/>

<RowDefinition Height="50" MaxHeight="50"/>

<RowDefinition Height="251\*"/>

</Grid.RowDefinitions>

<Image Height="220"

Margin="30,0,0,0"

MouseLeftButtonUp="GoRecommendProduct\_Click"

VerticalAlignment="Center"

HorizontalAlignment="Left"

Grid.Column="0"

Source="/Images/logo.png" />

<StackPanel Grid.Column="0"

x:Name="CartField"

Orientation="Horizontal"

MouseLeftButtonUp="MyCartNavigate\_Click"

HorizontalAlignment="Right"

VerticalAlignment="Center"

Margin="0,0,60,0">

<Image Source="/Images/retail.png"

Height="50" />

<StackPanel Orientation="Vertical"

VerticalAlignment="Center"

Margin="10,0,0,0">

<TextBlock Text="Моя корзина"

FontSize="14"/>

<TextBlock x:Name="shoppingСart"

FontSize="15"

Foreground="#004eff"

Text="Ваша корзина пуста"/>

</StackPanel>

</StackPanel>

<Rectangle Grid.Row="1" Fill="#000080"/>

<StackPanel x:Name="AdminButton"

Orientation="Horizontal"

Grid.Row="1"

Visibility="Hidden"

Margin="100,0,0,0">

<StackPanel.Resources>

<Style TargetType="Button">

<Setter Property="FontSize" Value="15"/>

<Setter Property="Height" Value="28"/>

<Setter Property="Width" Value="140"/>

<Setter Property="Foreground" Value="White"/>

<Setter Property="Background" Value="#000080"/>

<Style.Resources>

<Style TargetType="Border">

<Setter Property="CornerRadius" Value="3"/>

</Style>

</Style.Resources>

</Style>

</StackPanel.Resources>

<Button Content="Товары" Click="AdminGoProduct\_Click"/>

<Button Content="Продажи" Margin="20,0,0,0" Click="AdminGoSale\_Click"/>

</StackPanel>

<Grid x:Name="searchGrid"

Background="White"

Grid.Row="1"

Width="700"

Height="40">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="40"/>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<Image Source="/Images/Search-ico.png"

Margin="4,0,0,0"

Height="13"

Grid.Column="0"

VerticalAlignment="Center" />

<TextBox x:Name="searchBox"

Margin="4,0,0,0"

FontSize="15"

BorderThickness="0"

Foreground="Gray"

BorderBrush="#0000"

GotFocus="searchBox\_GotFocus"

LostFocus="searchBox\_LostFocus"

VerticalContentAlignment="Center"

Text="Введите название товара"

Grid.Column="1"

VerticalAlignment="Stretch" KeyDown="searchBox\_KeyDown" />

</Grid>

<Grid Grid.Row="1"

Margin="0,0,60,0"

HorizontalAlignment="Right">

<TextBlock x:Name="authAccount"

Foreground="White"

VerticalAlignment="Center"

Style="{StaticResource TextStyle}">

<Hyperlink Foreground="White" Click="Login\_Click">Войти</Hyperlink>

|

<Hyperlink Foreground="White" Click="Registration\_Click">Регистрация</Hyperlink>

</TextBlock>

<TextBlock x:Name="personalAccount"

Foreground="White"

Visibility="Hidden"

VerticalAlignment="Center"

Style="{StaticResource TextStyle}">

<Hyperlink Foreground="White" Click="PersonalAccount\_Click">Личный кабинет</Hyperlink>

|

<Hyperlink Foreground="White" Click="Close\_Click">Выйти</Hyperlink>

</TextBlock>

</Grid>

<Frame x:Name="PageFrame" Grid.Row="2"

Source="/Pages/MainRecommendationProduct.xaml"

NavigationUIVisibility="Hidden"/>

<Border Width="200"

Grid.Row =" 1"

MouseLeftButtonDown="CatalogVisible"

HorizontalAlignment="Left"

Margin="100,0,0,0" >

<StackPanel x:Name="CategoryStack"

Orientation="Horizontal" >

<Image Source="/Images/menu.png"

Margin="0,0,0,0"

Height="25" />

<TextBlock Foreground="White"

FontSize="15.5"

VerticalAlignment="Center"

Margin="15,0,0,2"

Text="Каталог товаров"/>

</StackPanel>

</Border>

<ListView x:Name="listProduct"

Visibility="Hidden"

Width="258"

Margin="60,0,0,100"

BorderBrush="Transparent" Grid.Row="2"

Padding="0"

Background="Transparent"

HorizontalAlignment="Left" >

<ListView.ItemTemplate>

<DataTemplate>

<Grid HorizontalAlignment="Stretch"

Background="#F5F5F5"

Width="258"

MinHeight="35"

Margin="-9,-9,-10,-8" >

<Expander Padding="0"

Header="{Binding Name}"

BorderBrush="Gray"

Expanded="Expander\_Expanded"

BorderThickness="0.5"

Background="#F5F5F5"

Foreground="#1719FD"

HorizontalAlignment="Stretch" >

<ListBox ItemsSource="{Binding Subcategories}">

<ListBox.ItemTemplate>

<DataTemplate>

<Border Width="258" MouseLeftButtonUp="Expander\_MouseLeftButtonUp" >

<TextBlock Text="{Binding Name}" FontSize="15"

VerticalAlignment="Center"

Height="30"

TextTrimming="WordEllipsis"

Margin="30,6,0,0"/>

</Border>

</DataTemplate>

</ListBox.ItemTemplate>

</ListBox>

</Expander>

</Grid>

</DataTemplate>

</ListView.ItemTemplate>

</ListView>

</Grid>

</ScrollViewer>

</Window>

MainWindow.xaml.cs:

public partial class MainWindow : Window

{

public static CarShopEntities carShopEntities = new CarShopEntities();

public static User client;

public static MainWindow Instance { get; private set; }

Expander exp = new Expander();

public MainWindow()

{

InitializeComponent();

listProduct.ItemsSource = carShopEntities.Categories.ToList();

Instance = this;

}

private List<GoodSale> goodCart = new List<GoodSale>();

public List<GoodSale> GoodCart

{

get

{

return goodCart;

}

set

{

goodCart = value;

if (goodCart.Count == 0)

{

this.shoppingСart.Text = "Ваша корзина пуста";

}

else

{

double sum = 0;

foreach (var item in goodCart)

{

sum = sum + Convert.ToDouble(item.Count \* item.Good.Price);

}

this.shoppingСart.Text = $"{goodCart.Count} товар на {sum} руб.";

}

}

}

private void Expander\_Expanded(object sender, RoutedEventArgs e)

{

if(exp != ((Expander)sender))

exp.IsExpanded = false;

exp = ((Expander)sender);

}

private void searchBox\_GotFocus(object sender, RoutedEventArgs e)

{

if(searchBox.Text == "Введите название товара")

{

searchBox.Clear();

searchBox.Foreground = Brushes.Black;

}

}

private void searchBox\_LostFocus(object sender, RoutedEventArgs e)

{

if (searchBox.Text.Length == 0)

{

searchBox.Text = "Введите название товара";

}

searchBox.Foreground = Brushes.Gray;

}

private void searchBox\_KeyDown(object sender, KeyEventArgs e)

{

if (e.Key == Key.Enter)

{

PageFrame.NavigationService.Navigate(new ListProduct(searchBox.Text));

}

}

private void Login\_Click(object sender, RoutedEventArgs e)

{

Authorization authorization = new Authorization(this);

CategoryHidden();

PageFrame.NavigationService.Navigate(authorization);

}

private void Registration\_Click(object sender, RoutedEventArgs e)

{

Registration registration = new Registration(this);

CategoryHidden();

PageFrame.NavigationService.Navigate(registration);

}

private void PersonalAccount\_Click(object sender, RoutedEventArgs e)

{

MyShoppingСart myShoppingСart = new MyShoppingСart();

CategoryHidden();

PageFrame.NavigationService.Navigate(myShoppingСart);

}

private void Close\_Click(object sender, RoutedEventArgs e)

{

MainRecommendationProduct mainRecommendationProduct = new MainRecommendationProduct();

PageFrame.NavigationService.Navigate(mainRecommendationProduct);

personalAccount.Visibility = Visibility.Hidden;

MainWindow.client = null;

authAccount.Visibility = Visibility.Visible;

if (client.IsAdmin == true)

{

AdminPanelHidden();

}

}

private void MyCartNavigate\_Click(object sender, MouseButtonEventArgs e)

{

PageFrame.NavigationService.Navigate(new MyShoppingСart(true));

CategoryHidden();

}

private void GoRecommendProduct\_Click(object sender, MouseButtonEventArgs e)

{

PageFrame.NavigationService.Navigate(new MainRecommendationProduct());

}

private void CatalogVisible(object sender, MouseButtonEventArgs e)

{

if (listProduct.Visibility == Visibility.Hidden)

{

CategoryVisibible();

}

else

{

CategoryHidden();

}

}

private void CategoryVisibible()

{

listProduct.Visibility = Visibility.Visible;

}

private void CategoryHidden()

{

listProduct.Visibility = Visibility.Hidden;

}

private void Expander\_MouseLeftButtonUp(object sender, MouseButtonEventArgs e)

{

listProduct.Visibility = Visibility.Hidden;

PageFrame.NavigationService.Navigate(new ListProduct((Subcategory)((Border)sender).DataContext));

}

internal void AdminPanelVisible()

{

CategoryStack.Visibility = Visibility.Hidden;

searchGrid.Visibility = Visibility.Hidden;

CartField.Visibility = Visibility.Hidden;

authAccount.Visibility = Visibility.Hidden;

personalAccount.Visibility = Visibility.Visible;

AdminButton.Visibility = Visibility.Visible;

}

internal void AdminPanelHidden()

{

CategoryStack.Visibility = Visibility.Visible;

searchGrid.Visibility = Visibility.Visible;

CartField.Visibility = Visibility.Visible;

authAccount.Visibility = Visibility.Visible;

personalAccount.Visibility = Visibility.Hidden;

AdminButton.Visibility = Visibility.Hidden;

}

private void AdminGoProduct\_Click(object sender, RoutedEventArgs e)

{

PageFrame.NavigationService.Navigate(new ListProductAdminView());

}

private void AdminGoSale\_Click(object sender, RoutedEventArgs e)

{

PageFrame.NavigationService.Navigate(new ListProductAdminView(false));

}

private void Border\_LostFocus(object sender, RoutedEventArgs e)

{

}

}

Министерство цифрового развития государственного управления,

информационных технологий и связи Республики Татарстан

государственное автономное профессиональное образовательное

учреждение

«МЕЖДУНАРОДНЫЙ ЦЕНТР КОМПЕТЕНЦИЙ – КАЗАНСКИЙ

ТЕХНИКУМ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И СВЯЗИ»

ЗАДАНИЕ

на курсовой проект

студента Шигапов Айрат Фаргатович

группы 420

по специальности 09.02.07 «Информационные системы и программирование» квалификация «Программист»

1. Тема курсового проекта: Разработка автоматизированной информационной системы онлайн - покупок автозапчастей, утверждена приказом по техникуму от « » 20 г. № \_\_\_\_

2. Срок сдачи студентом законченного проекта « » 20 г.

3. Исходные данные к курсовому проекту: Методические рекомендации к написанию документации, теоретический материал.

4. Перечень подлежащих разработке вопросов:

Пояснительная записка должна содержать следующие разделы:

1. введение;
2. разработка системного проекта;
3. проектирование программного продукта;
4. реализация проекта;
5. руководство пользователя;
6. тестирование программного продукта;
7. создание файла установщика приложения;
8. заключение;
9. список использованных источников.

Система должна обеспечивать возможность добавления, изменения и поиска данных в базе данных. Программный продукт имеет приятный и удобный интерфейс для работы пользователей. Для создания пользовательского интерфейса должна использоваться мягкая цветовая гамма.

План – график выполнения курсового проекта (КП):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № п/п | Наименование этапов КП | Срок выполнения этапов КП | | | Подпись руководителя, консультантов КП |
| План | Факт | |
| 1 | Введение | 09.09.21- 11.09.21 | 09.09.21- 12.09.21 |  | |
| 2 | Глава 1. Выбор и изучение среды разработки | 12.09.21- 16.09.21 | 13.09.21- 16.09.21 |  | |
| 3 | Глава 2. Изучение теоретических основ | 17.09.21- 19.09.21 | 17.09.21- 20.09.21 |  | |
| 4 | Глава 3. Разработка приложения | 20.09.21- 15.10.21 | 21.09.21- 15.10.21 |  | |
| 5 | Глава 4. Техническое обоснование проекта и тестирование | 15.10.21- 20.10.21 | 15.10.21- 20.10.21 |  | |
| 6 | Заключение | 21.10.21- 30.10.21 | 21.10.21- 31.10.21 |  | |
| 7 | Сдача КП | 05.11.21 |  |  | |

Рассмотрено на заседании ЦК Программирования

Протокол № \_\_\_ от « » 20 г.

Председатель ЦК \_\_\_\_\_\_\_\_\_\_

Дата выдачи задания:

« » 20 г.

Руководитель КП:

преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ И.И. Долгова

*инициалы и фамилия*

*подпись*

*должность*

Задание принял к исполнению: \_\_\_\_\_\_\_\_\_\_\_\_\_ А.Ф Шигапов

*инициалы и фамилия*

*подпись*