Главное управление по образованию

Минского областного исполнительного комитета

Учреждение образования

«Новопольский государственный аграрно-экономический колледж»

Специальность 2-40 01 01 «Программное обеспечение информационных технологий»

Специализация 2-40 01 01 35 «Программное обеспечение обработки экономической и деловой информации»

**ДЕЛО № 3-29**

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**к курсовому проекту по учебному предмету**

**«Конструирование программ и языки программирования»**

**на тему «Игровое приложение** **«UFO Invade»**

|  |  |  |
| --- | --- | --- |
| Разработал  учащийся учебной группы 2120 | \_\_\_\_\_\_\_\_\_\_ | И.В. Карнейчик |
| Руководитель проекта  преподаватель | \_\_\_\_\_\_\_\_\_\_ | В.Л. Захарич |
|  |  |  |
| Курсовой проект защищен с отметкой | \_\_\_\_\_\_\_\_\_\_ |  |

Новое Поле, 2023

**СОДЕРЖАНИЕ**

Изм.

Лист

№ докум.

Подпись

Дата

Лист

2

КП2120.10.106.081ПЗ

Разраб.

Карнейчик И.В.

Провер.

*Захарич В.Л.*

З

Реценз.

Н. Контр.

Утверд.

Игровое приложение «UFO Invade»

Лит.

Листов

32

НГАЭК, 2023

[ВВЕДЕНИЕ 3](#_Toc160223904)

[ОСНОВНАЯ ЧАСТЬ 4](#_Toc160223905)

[1 Назначение и область применения 4](#_Toc160223906)

[1.1 Характеристика предметной области 4](#_Toc160223907)

[1.2 Назначение программного средства 4](#_Toc160223908)

[1.3 Описание аналогов 4](#_Toc160223909)

[2 Технические характеристики 7](#_Toc160223910)

[2.1 Постановка задачи на разработку 7](#_Toc160223911)

[2.2 Описание программного средства 7](#_Toc160223912)

[2.2.1 Общие сведения 7](#_Toc160223913)

[2.2.2 Выбор среды разработки 8](#_Toc160223914)

[2.2.3 Выбор системы управления базами данных 9](#_Toc160223915)

[2.2.4 Построение и описание базы данных 10](#_Toc160223916)

[2.2.5 Построение и описание диаграмм программного средства 15](#_Toc160223917)

[2.2.6 Описание логической струтуры 15](#_Toc160223918)

[2.2.7 Тестирование 17](#_Toc160223919)

[2.2.8 Руководство по использованию 18](#_Toc160223920)

[ЗАКЛЮЧЕНИЕ 26](#_Toc160223921)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 27](#_Toc160223922)

[ПРИЛОЖЕНИЯ 28](#_Toc160223923)

# ВВЕДЕНИЕ

Компьютерная игра – компьютерная программа, служащая для организации игрового процесса (геймплея), связи с партнёрами по игре, или сама выступающая в качестве партнёра. В настоящее время ни один человек не может представить себе жизни без компьютера, его влияние достигло очень больших масштабов. Мы можем заметить, что ни в одной области не обходится без «железного друга», который в несколько раз увеличивает производительность труда. Но компьютер проник не только в рабочую сферу. Сейчас в каждом доме, каждый ребенок и взрослый использует это техническое средство.

В наше время многие любят играть в компьютерные игры. Даже взрослые люди порой не прочь отдохнуть, разложив пасьянс или задумавшись над головоломкой.

Объект исследования – игровое приложение.

Предмет исследования – игровое приложение «UFO Invade».

Цель данного курсового проекта – разработать игровое приложение «UFO Invade».

Задачи проекта:

* изучить предметную область;
* изучить существующие аналоги;
* разработать программное средство
* выполнить отладку программного средства;
* протестировать программное средство, сделать выводы.

Пояснительная записка к курсовому проекту состоит из 32 страниц, 19 рисунков, 2 таблиц, 7 источников литературы, 3 приложений.

Во введении описана актуальность разработки программного модуля, объект и предмет исследования, цель и задачи на курсовой проект.

Основная часть состоит из разделов: «Назначение программного средства», «Характеристика предметной области», «Описание аналогов».

В разделе «Назначение программного средства» было изучено назначение программного средства.

В разделе «Характеристика предметной области» была составленна характеристика прдметной области.

В разделе «Описание аналогов» исследованы, перечислены и описаны существующие аналоги.

В заключении подведены итоги проделанной работы, указаны перспективы модернизации и сопровождения программного средства.

# ОСНОВНАЯ ЧАСТЬ

### 1 Назначение и область применения

### 1.1 Характеристика предметной области

Программное средство для автоматизации работы складского учёта на примере ОАО «Белвторчермет» представляет собой систему, которая позволяет пользователям взаимодействовать с услугами ОАО «Белвторчермет», самостоятельно оформлять и оплачивать заказы из личного кабинета.

Основная цель веб-приложения – предоставить пользователям возможность удобного сбора и оформления заказов по закупке металлолома, автоматизировать работу склада на предприятии, предоставить администраторам сайта возможности по полной настройке содержимого сайта, которое в свою очередь загружается из базы данных.

Программное средство для автоматизации складского учёта является важным инструментом для клиентов, желающих быстро и легко закупить необходимое количество металлолома который проходит переработку на ОАО «Белвторчермет». Программное средство позволяет проводить транзакции онлайн, что очень важно в нынешнее время.

### 1.2 Назначение программного средства

Назначением игрового приложения является предоставление пользователям увлекательного игрового процесса, который позволит интересно скоротать время и даст посоревноваться с друзьями за новые рекорды .

Данное приложение предназначено для детской аудитории. Оно поможет своим маленьким пользователям быстро развить моторику, умение быстро находить решение в стрессовых ситуациях и развивает внимание. Из-за очень простого интерфейса игрового приложения в нем может разобраться любой ребенок.

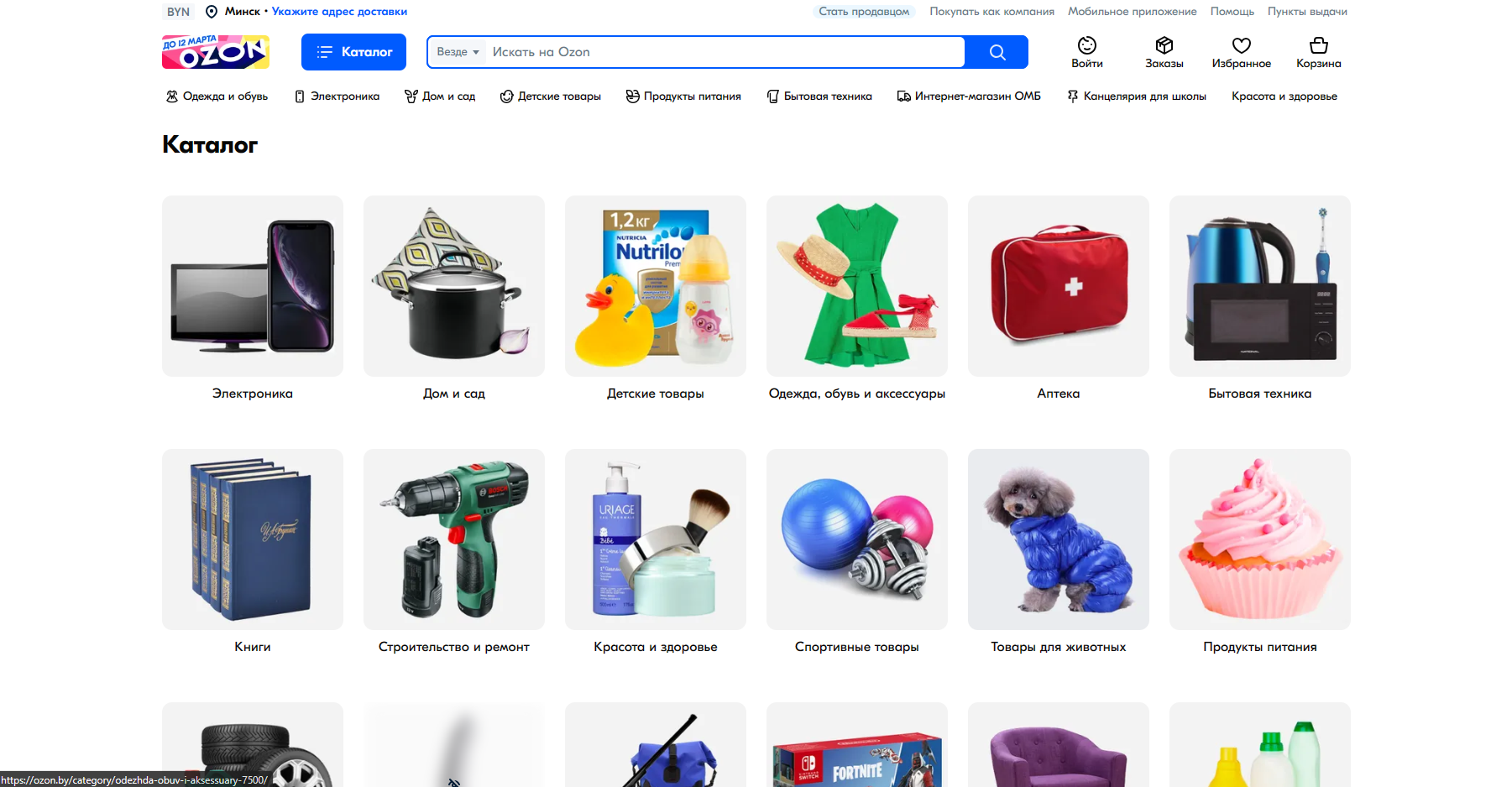
### 1.3 Описание аналогов

У приложения «Автоматизация складского учёта на примере ОАО «Белвторчермет» есть много аналогов и приложений, по своей сути напоминающих его, такие как:

Ozon – российский маркетплейс. Основан в 1998 году как интернет-магазин по продаже книг и видеокассет. Работает по модели онлайн-платформы: продавцы размещают товары на сайте, площадка организует их доставку и продажу покупателям. Помимо торговой площадки, компания развивает экспресс-доставку товаров повседневного спроса Ozon fresh, доставку товаров из-за рубежа Ozon Global, финансовые сервисы от Ozon Банк, а также бронирование авиа и железнодорожных билетов, отелей и туров Ozon Travel. Компания работает в России, Армении, Беларуси, Казахстане, Китае, Кыргызстане, Турции и Узбекистане.

По итогам на 2022 года занимает пятое место по обороту среди российских ритейлеров. Оборот за первые 9 месяцев 2023 года составил 1,12 трлн рублей – маркетплейс доставил 639 млн заказов для 42,4 млн покупателей в России и СНГ. Ассортимент составил 250 млн товаров, число продавцов на платформе – более 400 тысяч.

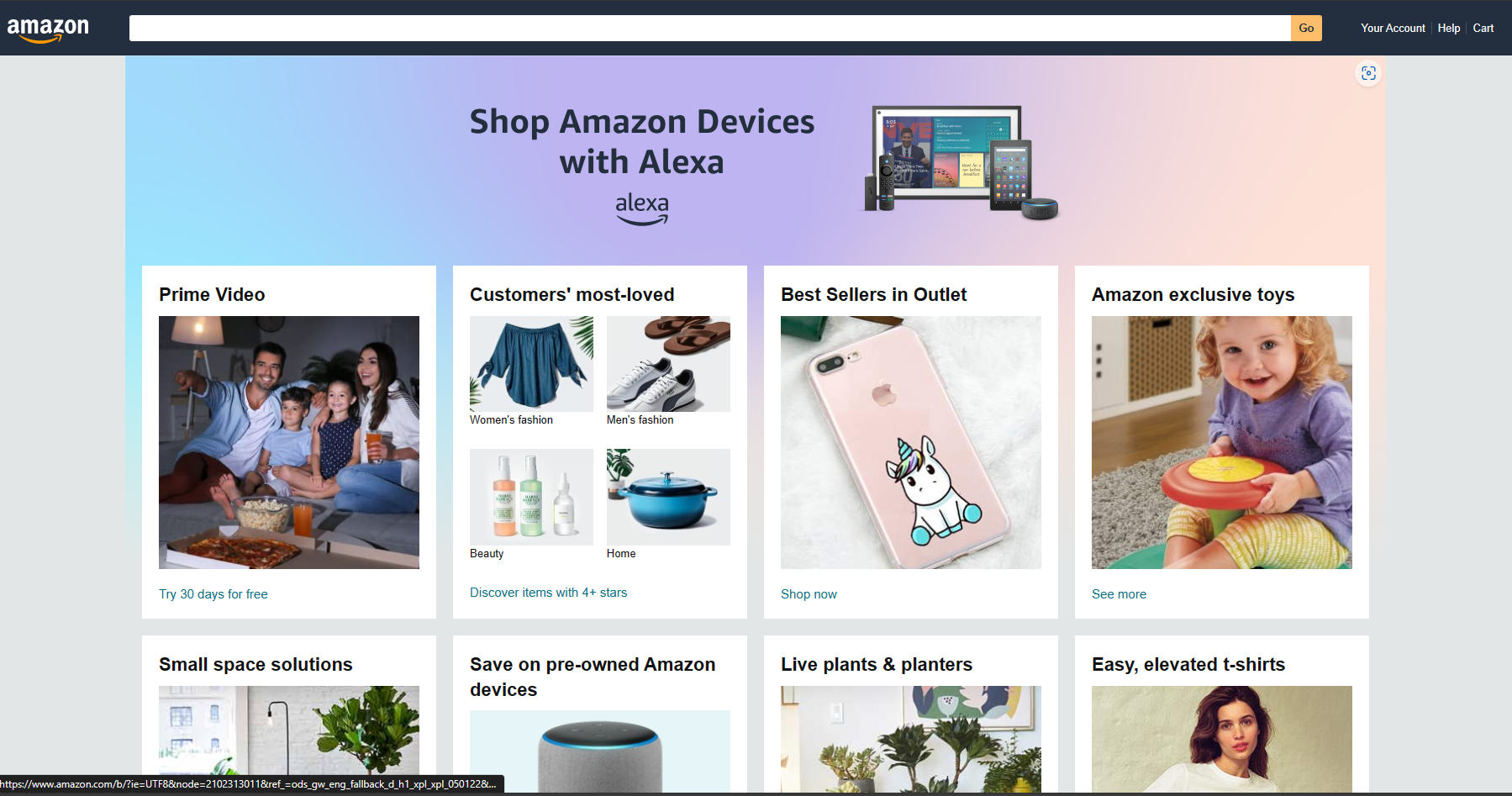
Ozon использует склады и сортировочные центры для хранения и доставки товаров. На сентябрь 2023 года маркетплейс управлял 2 млн м² логистической инфраструктуры в России и СНГ. Сеть пунктов выдачи развивается по партнерской модели, на сентябрь 2023 года работает более 40 тысяч точек.

****

**Рисунок 1.3.1 – «OZON»**

Amazon – американская компания, крупнейшая в мире на рынках платформ электронной коммерции и публично-облачных вычислений по выручке и рыночной капитализации. Штаб-квартира – в Сиэтле.

Основана Джеффом Безосом 5 июля 1994 года как интернет-магазин по продаже книг, но позже диверсифицировалась, начав продавать видео, MP3, аудиокниги (как потоковым способом, так и через возможность загрузки), программное обеспечение, видеоигры, электронику, одежду, мебель, еду, игрушки и ювелирные изделия. Компания также владеет издательским подразделением Amazon Publishing, киностудией Amazon Studios, производит линии бытовой электроники, включая электронные книги Kindle, планшеты Amazon Fire, Fire TV и смарт-динамик Echo, и является крупнейшим в мире поставщиком услуг в моделях IaaS и PaaS (Amazon Web Services). Поддерживает отдельные розничные сайты для некоторых стран, а также предлагает международную доставку своих продуктов в другие страны. Около 200 миллионов человек подписались на Prime Video.

****

**Рисунок 1.3.2 – «Amazon»**

### 2 Технические характеристики

### 2.1 Постановка задачи на разработку

Индивидуальное задание по разработке программного обеспечения:

Программное средство «Автоматизация складского учёта на примере ОАО «Белвторчермет»

Целью написания данного программного средства является разработка удобного в использовании сайта для учёта склада и закупки металлолома, с дружелюбным пользовательским интерфейсом и функциями администрирования, недоступными для обычных пользователей.

Программное средство для автоматизации складского учёта – это программное средство, позволяющее просматривать позиции на складе, создавать и оплачивать онлайн-заказ, создавать и настраиваить позиции на складе.

Программное средство для автоматизации складского учёта должно иметь следующие функции:

* быстрый и удобный доступ к оформлению заказов;
* опции администрирования базы данных склада;
* удобная работу с транзакциями и заказами;
* функционал создания и настройки личного кабинета как администратора, так и пользователя;
* интуитивная навигация по сайту и простая структура сайта;
* удобную и несложную административную панель.

### 2.2 Описание программного средства

### 2.2.1 Общие сведения

Программное средство разработано при помощи среды разработки Visual Studio 2022. Бекенд программного средства разработан при помощи ASP.NET Core на языке программирования C#, c использованием Entity Framework для работы с базой данных. Фронтенд веб приложения разработан при помощи HTML и CSS, с использованием фреймворка Bootstrap для создания адаптивного и мобильно-ориентированного дизайна. Работа с транзакциями осуществляется при помощи серсвиса BrainTree. Для создания авторизации, регистрации и настройки личного кабинета используются Razor Pages.

Для корректного функционирования программы необходимо следующее программное обеспечение:

‒ операционная система Windows или Linux;

‒ версия .NET Core не ниже 3.1;

‒ должны быть установлены все необходимые библиотеки.

### 2.2.2 Выбор среды разработки

Для выполнения курсового проекта был выбран Visual Studio.

Visual Studio – Интегрированная среда разработки (IntegratedDevelopmentEnvironment – IDE) Visual Studio предлагает ряд высокоуровневых функциональных возможностей, которые выходят за рамки базового управления кодом.

Ниже перечислены основные преимущества IDE-среды Visual Studio.

Встроенный Web-сервер. Для обслуживания Web-приложения ASP.NET необходим Web-сервер, который будет ожидать Web-запросы и обрабатывать соответствующие страницы. Наличие в Visual Studio интегрированного Web-сервера позволяет запускать Web-сайт прямо из среды проектирования, а также повышает безопасность, исключая вероятность получения доступа к тестовомуWeb-сайту с какого-нибудь внешнего компьютера, поскольку тестовый сервер может принимать соединения только с локального компьютера.

Поддержка множества языков при разработке. Visual Studio позволяет писать код на своем языке или любых других предпочитаемых языках, используя все время один и тот же интерфейс (IDE). Более того, Visual Studio также еще позволяет создавать Web-страницы на разных языках, но помещать их все в одно и то же Web-приложение. Единственным ограничением является то, что в каждой Web-странице можно использовать только какой-то один язык (очевидно, что в противном случае проблем при компиляции было бы просто не избежать).

Меньше кода для написания. Для создания большинства приложений требуется приличное количество стандартного стереотипного кода, и Web-страницы ASP. NET тому не исключение. Например, добавление Web-элемента управления, присоединение обработчиков событий и корректировка форматирования требует установки в разметке страницы ряда деталей. В Visual Studio такие детали устанавливаются автоматически.

Интуитивный стиль кодирования. По умолчанию Visual Studio форматирует код по мере его ввода, автоматически вставляя необходимые отступы и применяя цветовое кодирование для выделения элементов типа комментариев. Такие незначительные отличия делают код более удобным для чтения и менее подверженным ошибкам. Применяемые Visual Studio автоматически параметры форматирования можно даже настраивать, что очень удобно в случаях, когда разработчик предпочитает другой стиль размещения скобок (например, стиль K&R, при котором открывающая скобка размещается на той же строке, что и объявление, которому она предшествует).

Более высокая скорость разработки. Многие из функциональных возможностей Visual Studio направлены на то, чтобы помогать разработчику делать свою работу как можно быстрее. Удобные функции, вроде функции IntelliSense (которая умеет перехватывать ошибки и предлагать правильные варианты), функции поиска и замены (которая позволяет отыскивать ключевые слова как в одном файле, так и во всем проекте) и функции автоматического добавления и удаления комментариев (которая может временно скрывать блоки кода), позволяют разработчику работать быстро и эффективно.

Возможности отладки. Предлагаемые в Visual Studio инструменты отладки являются наилучшим средством для отслеживания загадочных ошибок и диагностирования странного поведения. Разработчик может выполнять свой код по строке за раз, устанавливать интеллектуальные точки прерывания, при желании сохраняя их для использования в будущем, и в любое время просматривать текущую информацию из памяти.

Visual Studio также имеет и множество других функций: возможность управления проектом; встроенная функция управления исходным кодом; возможность рефакторизации кода; мощная модель расширяемости. Более того, в случае использования Visual Studio 2008 Team System разработчик получает расширенные возможности для модульного тестирования, совместной работы и управления версиями кода (что значительно больше того, что предлагается в более простых инструментах вроде Visual SourceSafe).

### 2.2.3 Выбор системы управления базами данных

Для данного курсового проекта в качестве СУБД был выбран MS SQL SERVER

SQL Server – это программа, которая предназначена для хранения и обработки данных. При взаимодействии с ней пользователи могут отправлять запросы и получать ответы – причем как локально, так и по сети. Функционирует программа следующим образом: открывает сетевой порт, принимает команды и выдает результат.

Для работы по локальной сети СУБД устанавливается на рабочем устройстве юзера. Режим работы зависит от выбранной системы. По умолчанию пользователи взаимодействуют с базой данных MS Access. Такой режим подходит в тех случаях, если с БД будут работать не более 10-12 человек.

Для работы в сетевой среде файл базы данных заливается на общедоступный ресурс. И пользователи подключаются к нему со своих ПК (на каждом устройстве устанавливается специальное программное обеспечение).

Если с базой данных будут работать более 12 человек одновременно или если обнаружится, что сервер не выдерживает нагрузки, лучше перевести БД в формат Microsoft SQL. В данном режиме все вычисления происходят на стороне сервера, а не на компьютере сотрудника, что гарантирует высокое быстродействие даже при одновременном подключении большого количества пользователей.

Купить Microsoft SQL Server вы можете не выходя из дома, заказав его по ссылке в специализированном магазине SoftMagazin. Компания на IT-рынке более 17 лет, является авторизированным партнером Microsoft, поэтому продает только лицензионные решения от ведущего разработчика.

Преимущества и недостатки программы

Рассмотрим, какие у Microsoft SQL Server преимущества и недостатки. Это нужно, чтобы вы понимали, насколько данная СУБД подходит для ваших целей.

Основные достоинства:

– масштабирование системы. Взаимодействовать с ней можно как на простых ноутбуках, так и на ПК с мощным процессором, который способен обрабатывать большой объем запросов;

– размер страниц – до 8 Кб. Данные извлекаются быстро, а сложную информацию удобнее хранить. Система обрабатывает транзакции в интерактивном режиме, есть динамическая блокировка;

– автоматизация рутинных административных задач. Например, управление блокировками и памятью, редактура размеров файлов. В программе продуманы настройки, можно создавать профили пользователей;

– удобный поиск. Его можно осуществлять по фразам, словам, тексту либо создавать ключевые индексы;

– поддержка работы с другими решениями Майкрософт, в том числе с Excel, Access.

Также в программе предусмотрена синхронизация, есть репликации через интернет, службы преобразования информации и полноценный web-ассистент для форматирования страниц. Дополнительно в нее интегрирован сервис интерактивного анализа (можно принимать решения, создавать корпоративные отчеты).

### 2.2.4 Построение и описание базы данных

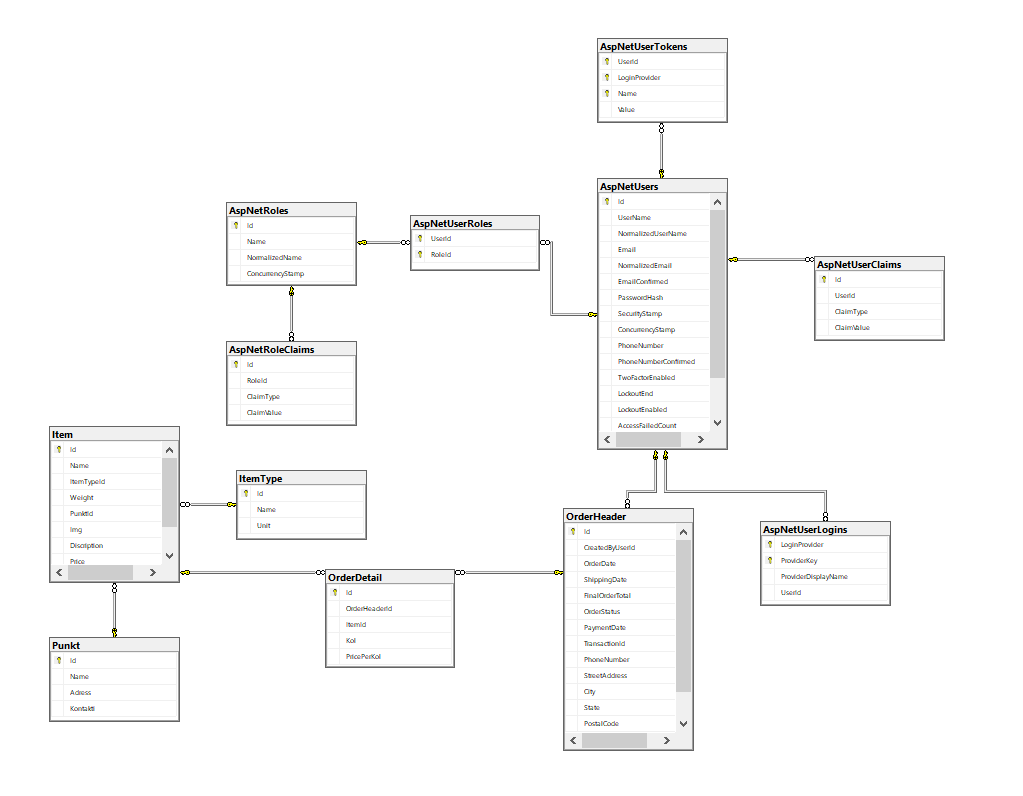
Проектирование баз данных (БД) – процесс создания схемы базы данных и определения необходимых ограничений целостности. Проектирование базы данных сводится к трем последовательным этапам:

* Концептуальное проектирование – сбор, анализ и редактирование требований к данным.
* Логическое проектирование – преобразование требований к данным в структуры данных.
* Физическое проектирование – определение особенностей хранения данных, методов доступа и т.д.

Концептуальное (инфологическое) моделирование, или концептуальное проектирование базы данных – это процесс конструирования модели использования информации на некотором предприятии, он не зависит от таких подробностей, как используемая СУБД, прикладные программы, языки программирования и т.д.

Логическая модель данных – описание объектов предметной области, их атрибутов и взаимосвязей между ними в том объеме, в котором они подлежат непосредственному хранению в базе данных системы.

Целью построения логической модели является получение графического представления логической структуры исследуемой предметной области. Логическая модель предметной области иллюстрирует сущности, а также их взаимоотношения между собой.

****

**Рисунок 2.2.4.1 – Логическая модель базы данных**

* В веб-приложении используется база данных, содержащая следующие таблицы:
* «Item» – содержит информацию о товарах;
* «Punkt» – содержит информацию о пунктах хранения товара;
* «ItemType» – содержит информацию о типах товаров;
* «OrderHeader» – содержит основную информацию о заказах пользователей сайта;
* «OrederDetail» – содержит дополнительную информацию о заказах;

Физическое проектирование БД

Физическое моделирование базы данных – создание схемы базы данных для конкретной СУБД. Проектирование физической структуры заключается в определении места хранения БД, а также форматов, хранимых данных на уровне отдельных полей.

База данных состоит из 10 таблиц, связи между которыми обеспечивают наименьшую избыточность данных.

Сущность «Item» предназначена для хранения информации о товарах на складе.

Таблица 2.2.4.1 – Атрибуты сущности «Item»

|  |  |  |
| --- | --- | --- |
| Атрибут | Тип данных | Назначение |
| ld | int | Уникальный ключ товара |
| Name | nvarchar(MAX) | Наименование товара |
| ItemTypeld | int | Id типа товара |
| Weight | int | Вес товара |
| Punktld | int | Id пункта нахождения товара на складе |
| Img | nvarchar(MAX) | Хранение путей к изображениям |
| Discription | nvarchar(MAX) | Описание товара |
| Price | decimal(18, 2) | Цена товара |
| ShortDesc | nvarchar(MAX) | Короткое описание товара |

Сущность «Punkt» предназначена для хранения информации о пунктах хранения товаров.

Таблица 2.2.4.2 – Атрибуты сущности «Punkt»

|  |  |  |
| --- | --- | --- |
| Атрибут | Тип данных | Назначение |
| Id | int | Уникальный ключ пункта |
| Name | nvarchar(MAX) | Наименование пункта |
| Adress | nvarchar(MAX) | Адресс пункта |
| Kontakti | nvarchar(MAX) | Контакты пункта |

Сущность «ItemType» предназначена для хранения информации о типах товаров.

Таблица 2.2.4.3 – Атрибуты сущности «ItemType»

|  |  |  |
| --- | --- | --- |
| Атрибут | Тип данных | Назначение |
| Id | int | Уникальный ключ типа товара |
| Name | nvarchar(MAX) | Наименование типа товара |
| Unit | nvarchar(MAX) | Еденица измерения |

Сущность «OrderHeader» предназначена для хранения главной информации о заказах.

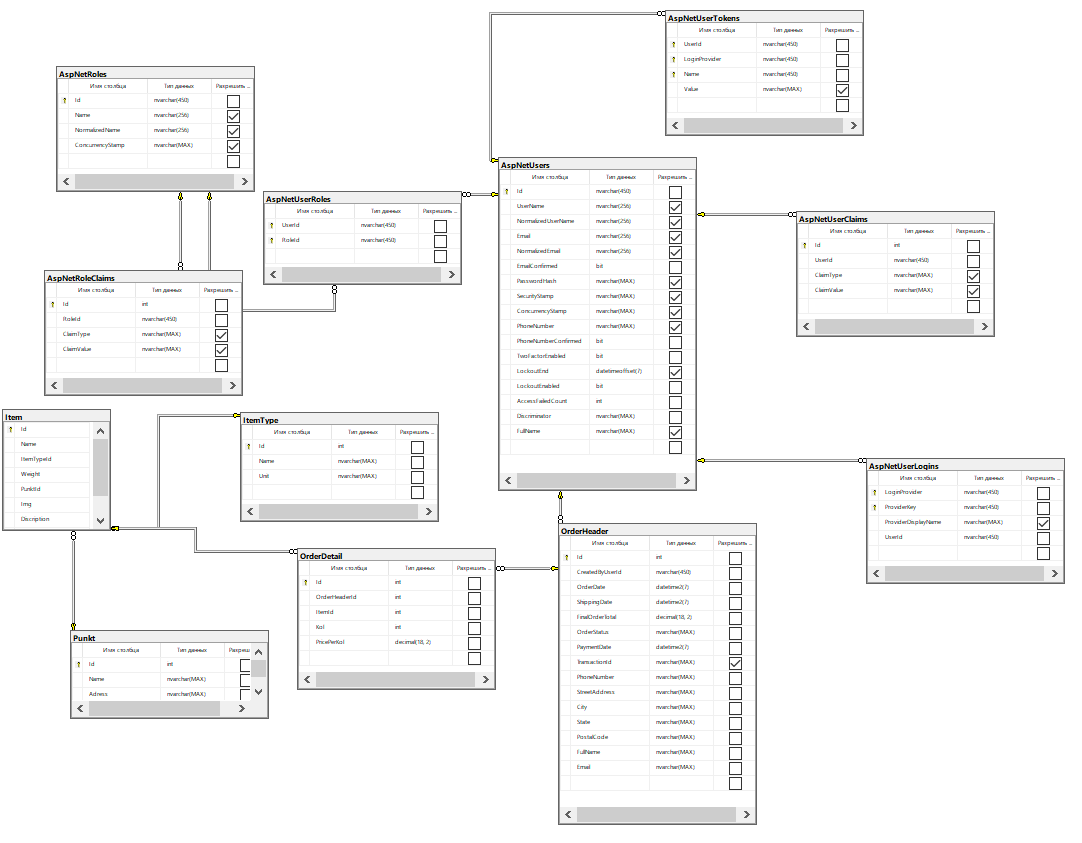
Таблица 2.2.4.4 – Атрибуты сущности «OrderHeader»

|  |  |  |
| --- | --- | --- |
| Атрибут | Тип данных | Назначение |
| Id | int | Уникальный ключ заказа |
| CreatedByUserld | nvarchar(450) | Содержит уникальный ключ пользователя оставившего заказ |
| OrderDate | datetime2(7) | Дата составления заказа |
| ShippingDate | datetime2(7) | Дата отправки заказа |
| FinalOrderTotaI | decimaI(18, 2) | Конечная цена заказа |
| OrderStatus | nvarchar(MAX) | Статус заказа |
| PaymentDate | datetime2(7) | Дата оплаты заказа |
| Transactionld | nvarchar(MAX) | Уникальный ключ транзакции |
| PhoneNumber | nvarchar(MAX) | Номер телефона заказчика |
| StreetAddress | nvarchar(MAX) | Улица доставки |
| City | nvarchar(MAX) | Город доставки |
| State | nvarchar(MAX) | Страна доставки |
| Posta1Code | nvarchar(MAX) | Почтовый индекс |
| FullName | nvarchar(MAX) | Имя заказчика |
| EmaiI | nvarchar(MAX) | Электронная почта заказчика |

Сущность «OrederDetail» предназначена для хранения дополнительной информации о заказах.

Таблица 2.2.4.5 – Атрибуты сущности «OrderDetail»

|  |  |  |
| --- | --- | --- |
| Атрибут | Тип данных | Назначение |
| Id | int | Уникальный ключ |
| OrderHeaderId | int | Уникальный ключ заказа |
| ItemId | int | Уникальный ключ товара |
| Kol | int | Количество заказанного товара |
| PricePerKol | decimal(18, 2) | Цена за количество товара |

****

**Рисунок 2.2.4.2 – Физическая модель базы данных**

В результате было произведено физическое моделирование базы данных. Выделены объекты базы данных: «Item», «Punkt», «ItemType», «OrderHeader», «OrderDetail», выделены атрибуты таблиц, определены связи между таблицами. Определена совокупность методов и средств размещения данных во внешней памяти.

### 2.2.5 Построение и описание диаграмм программного средства

При создании программного средства были созданы 3 диаграммы:

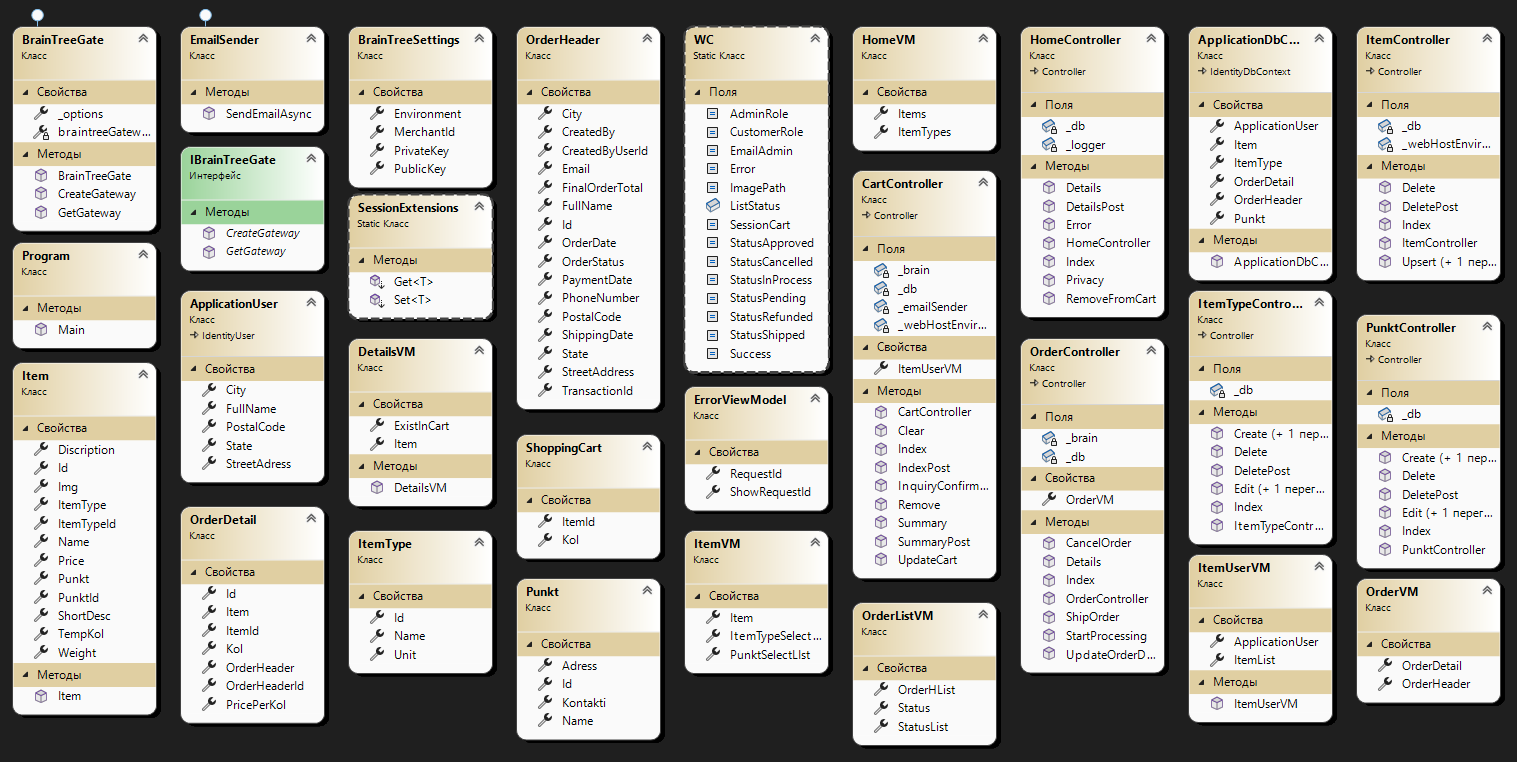
* диаграмма классов;
* диаграмма вариантов использования;
* диаграмма действий.

Диаграмма классов – это структурная диаграмма языка моделирования UML, которая демонстрирует общую структуру иерархии классов в системе. Она показывает связи между классами, их атрибуты (поля), методы и кооперации. Диаграмма классов широко используется для документирования, визуализации и конструирования систем.

Вот некоторые ключевые элементы диаграммы классов:

* Имя класса: В верхней части диаграммы указывается имя класса, начинающееся с заглавной буквы;
* Атрибуты и методы: Классы содержат атрибуты (поля) и методы. Атрибуты представляют данные, а методы – операции, которые класс может выполнять;
* Взаимосвязи: Диаграмма классов показывает отношения между классами, такие как зависимость, ассоциация, агрегация, композиция, обобщение и реализация.

Цель создания диаграммы классов – графическое представление статической структуры декларативных элементов системы. Она помогает разработчикам понять взаимосвязи между классами, атрибутами и методами, а также упрощает проектирование и реализацию системы.

****

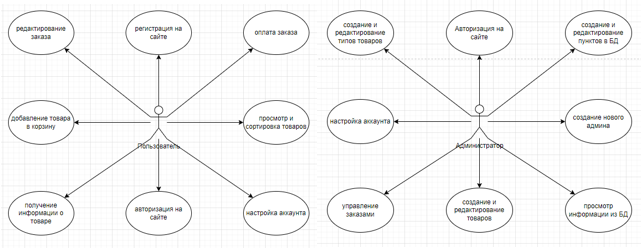
**Рисунок 2.2.5.1 – Диаграмма классов**

Диаграмма вариантов использования – это структурная диаграмма языка моделирования UML, которая фиксирует функциональность и требования системы с помощью субъектов и вариантов использования. Варианты использования моделируют сервисы, задачи и функции, которые должна выполнять система. Они представляют собой функциональные возможности высокого уровня и то, как пользователь будет обращаться с системой.

Вот некоторые ключевые элементы диаграммы вариантов использования:

* Имя класса: В верхней части диаграммы указывается имя класса, начинающееся с заглавной буквы;
* Атрибуты и методы: Классы содержат атрибуты (поля) и методы. Атрибуты представляют данные, а методы – операции, которые класс может выполнять;
* Взаимосвязи: Диаграмма вариантов использования показывает отношения между классами, такие как зависимость, ассоциация, агрегация, композиция, обобщение и реализация1.

Цель создания диаграммы вариантов использования – графическое представление статической структуры декларативных элементов системы. Она помогает разработчикам понять взаимосвязи между классами, атрибутами и методами, а также упрощает проектирование и реализацию системы.

****

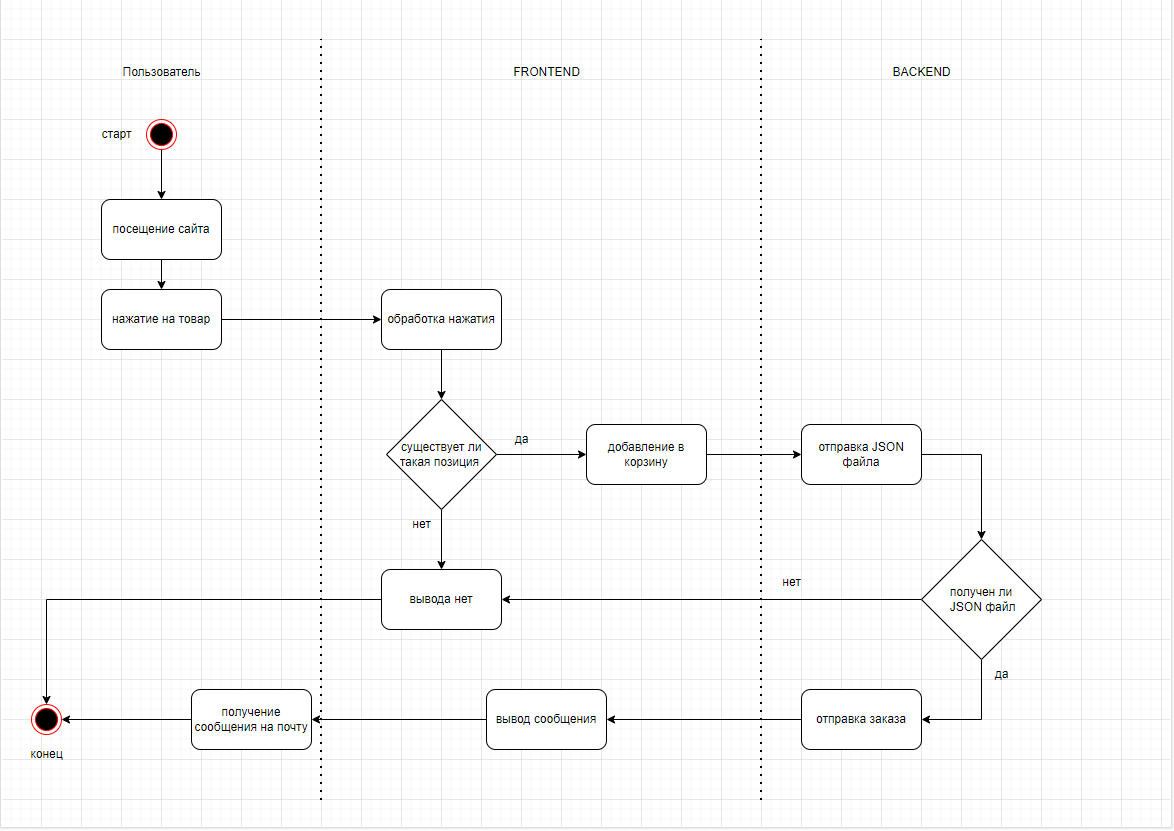
**Рисунок 2.2.5.2 – Диаграмма вариантов использования**

Диаграмма действий – это блок-схема действий, которая показывает, как одно действие ведет к другому. Она представляет рабочий процесс между различными действиями системы. Диаграммы действий похожи на блок-схемы, но они не являются блок-схемами. Вместо этого диаграмма действий – это усовершенствование потоковой диаграммы, которая содержит некоторые уникальные возможности.

Вот некоторые ключевые элементы диаграммы действий:

* Активности: Это поведение, которое разделено на одно или несколько действий. Действия представляют собой сеть узлов, соединенных ребрами. Это могут быть узлы действий, узлы управления или узлы объектов. Узлы действий представляют собой конкретные действия. Узлы управления отражают поток управления действиями. Узлы объектов используются для описания объектов, используемых внутри действия.
* Раздел активности/дорожка: Это высокоуровневая группировка набора связанных действий. Раздел может относиться к классам, вариантам использования, компонентам или интерфейсам. Если раздел невозможно отобразить четко, его имя пишется поверх названия действия.
* Узлы разветвления и соединения: С помощью узлов разветвления и соединения можно создавать параллельные потоки внутри действия. Узел-вилка имеет одно входящее ребро и множество исходящих ребер, а узел соединения выполняет логическую операцию И на всех входящих фронтах.
* Пальцы: Булавки используются для упрощения сложных диаграмм действий. Они помогают управлять потоком выполнения действий, сортируя потоки и убирая ненужные детали.
* Узлы объектов: Эти узлы представляют входы или выходы действия. И входные, и выходные контакты имеют ровно одно ребро.

Диаграмма действий позволяет создать событие как действие, содержащее набор узлов, соединенных ребрами. Она помогает описать поток управления и последовательность действий в системе. Это мощный инструмент для моделирования и анализа динамического поведения системы.

****

**Рисунок 2.2.5.3 – Диаграмма действий**

### 2.2.6 Описание логической струтуры

В ходе работы на индивидуальным заданием было разработано 6 контроллеров, 14 моделей, 7 представлений и 3 класса:

* контроллер CartController.cs, который обрабатывает работу карзины товаров, создание транзакций и отправку сообщений на электронную почту;
* контроллер HomeController.cs, который обрабатывает работу главной страницы товаров;
* контроллер ItemController.cs, который обрабатывает работу страницы товаров, позволяет администртору добавлять, изменять и удалять товары;
* контроллер ItemTypeController.cs, который обрабатывает работу страницы типов предметов, позволяет администртору добавлять, изменять и удалять типы;
* контроллер PunktController.cs, который обрабатывает работу страницы пунктов выдачи, позволяет администртору добавлять, изменять и удалять пункты;
* контроллер OrderController.cs, который обрабатывает работу администратора с заказами;
* модель ApplicationUser.cs, которая хранит модель значений для данных пользователя;
* модель Item.cs, которая хранит модель значений для товаров на складе;
* модель ItemType.cs, которая хранит модель значений для типов товаров;
* модель OrderDetail.cs, которая хранит модель значений для деталей заказа;
* модель OrderHeader.cs, которая хранит модель значений для данных о заказе;
* модель Punkt.cs, которая хранит модель значений для пунктов выдачи;
* модель ShoppingCart.cs, которая хранит модель значений для корзины товаров;
* модель DetailsVM.cs, которая хранит экземпляр модели Item и значений типа bool для проверки нахождения товара в корзине;
* модель HomeVM.cs, которая хранит список значений IEnumerable<Item> и IEnumerable<ItemType>;
* модель ItemUserVM.cs, которая хранит экземпляр класса ApplicationUser и список значений IList<Item>;
* модель ItemVM.cs, которая хранит экземпляр класса Item, список значений IEnumerable<SelectListItem>;
* модель OrderListVM.cs, которая хранит список значений IEumerable<OrderHeader>, IEnumerable<SelectListItem> и поле для статуса заказа;
* модель OrderVM.cs, которая храниь экземпляр класса OrderHeader и список значений IEnumerable<OrderDetail>;
* класс WC.cs, который нужен для хранения веб-констант;
* класс ApplicationDbContext.cs, который нужен для обращения к БД через перечисленные выше модели используя EntityFramework Core;
* класс Program.cs, который нужен для подключения сервисов, запуска и настройки проекта.

Также в ходе работы были разработаны 3 диаграммы:

* диаграмма классов (приложение Г);
* диаграмма вариантов использования (приложение Д);
* диаграмма скруктуры базы данных (приложение Е).

### 2.2.7 Тестирование

Таблица 2.2.7.1 – Тестирование программного средства

|  |  |  |  |
| --- | --- | --- | --- |
| Модуль | Описание теста | Ожидаемый результат | Полученный результат |
| Поисковая панель | Поиск концерта по названию | Отображение подходящих концертов | Отображение подходящих концертов |
| Панель фильтров | Поиск концертов по дате | Отображение подходящих концертов | Отображение подходящих концертов |
| Панель фильтров | Поиск концертов по типу | Отображение подходящих концертов | Отображение подходящих концертов |
| Окно создания админа | Попытка зарегестрировать админа | Успешная регистрация | Успешная регистрация |
| Страница оформления заказа | Заполнение форм необходимых для создания заказа | Успешное создание заказа | Успешное создание заказа |
| Корзина товаров | Добавление товаров в корзину, изменение их количества и подсчёт итоговой цены | Учпешное добавление товара в корзину, изменение количества и корректное отображение цены | Учпешное добавление товара в корзину, изменение количества и корректное отображение цены |
| Страница с заказми | Переход на страницу и попытка провести манипуляции с заказом | Успешное отображение всех заказов и сохранение произведенных манипуляций | Успешное отображение всех заказов и сохранение произведенных манипуляций |
| Страница «ItemType» | Добавление, удаление и редактирование записей в базе данных | Успешное добавление, удаление и редактирование | Успешное добавление, удаление и редактирование |
| Страница «Item» | Добавление, удаление и редактирование записей в базе данных | Успешное добавление, удаление и редактирование | Успешное добавление, удаление и редактирование |

Окончание таблицы 2.2.7.1

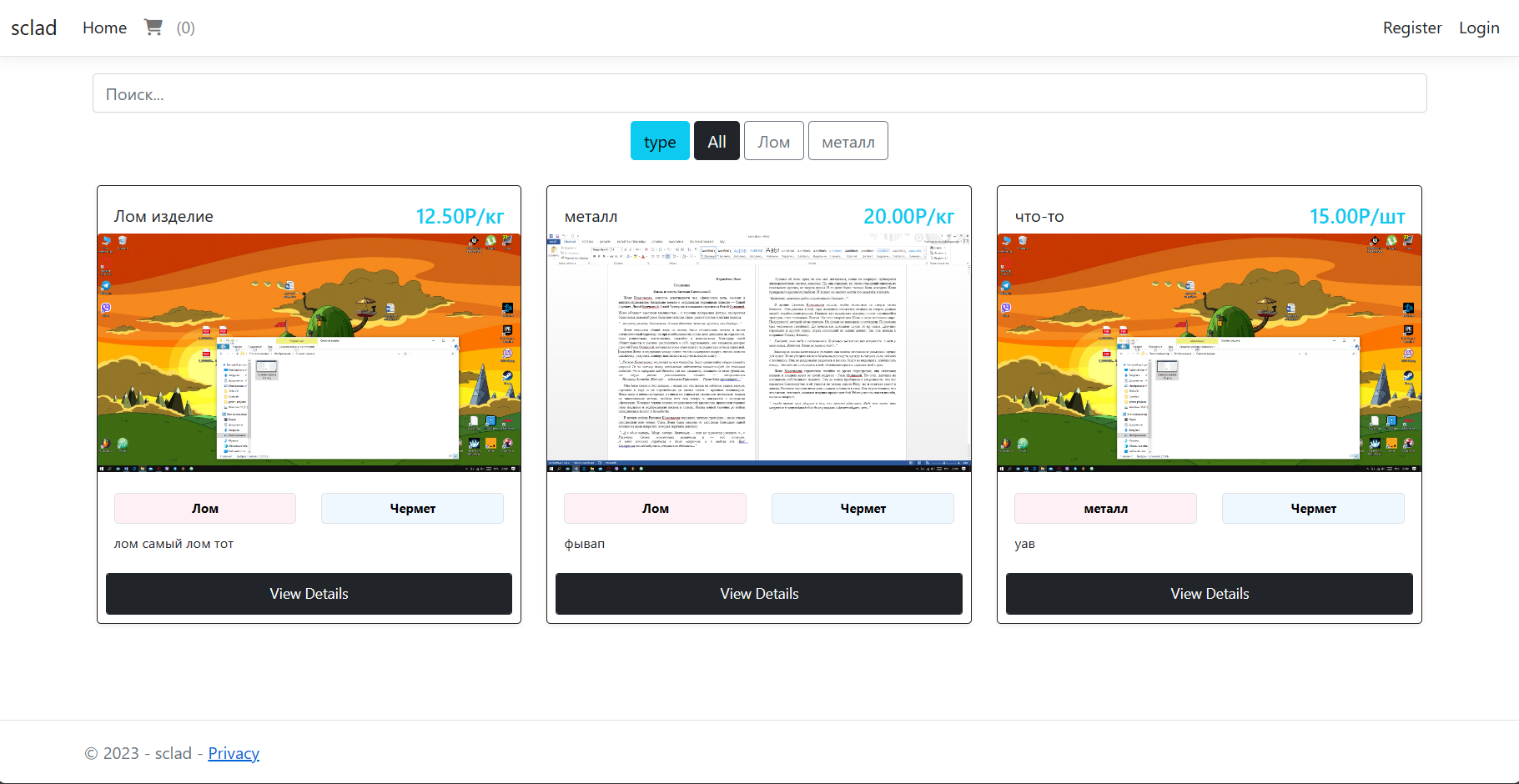
|  |  |  |  |
| --- | --- | --- | --- |
| Страница «Punkt» | Добавление, удаление и редактирование записей в базе данных | Успешное добавление, удаление и редактирование | Успешное добавление, удаление и редактирование |
| Регистрация | Попытка зарегистрироваться | Успешная регистрация с последующей авторизацией | Успешная регистрация с последующей авторизацие |
| Авторизация | Попытка авторизоваться | Успешная авторизация | Успешная авторизация |

Исходя из данных таблицы 2.2.7.1, можно сделать вывод, что программное средство работает корректно.

### 2.2.8 Руководство по использованию

Главная страница сайта (рисунок 2.2.8.1) на которой находится:

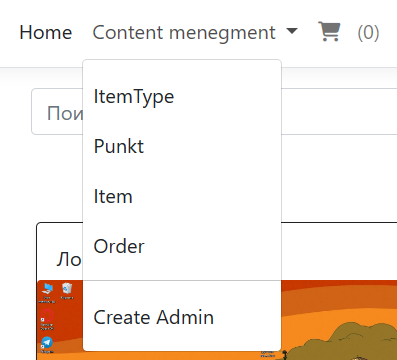
* панель навигации;
* строка поиска товаров;
* кнопки для сортировки товаров;
* карточки товаров.



**Рисунок 2.2.8.1 – Главная страница**

В навигационной панели находятся пункты меню (рисунок 2.2.8.2):

* пункт «Home» для перехода на главную стриницу;
* пункт «Корзина» для перехода в корзину товаров для текущей сессии;
* пункт «Content menegment», который доступен только админам и позволяет перейти в разделы с функциями администратора.

****

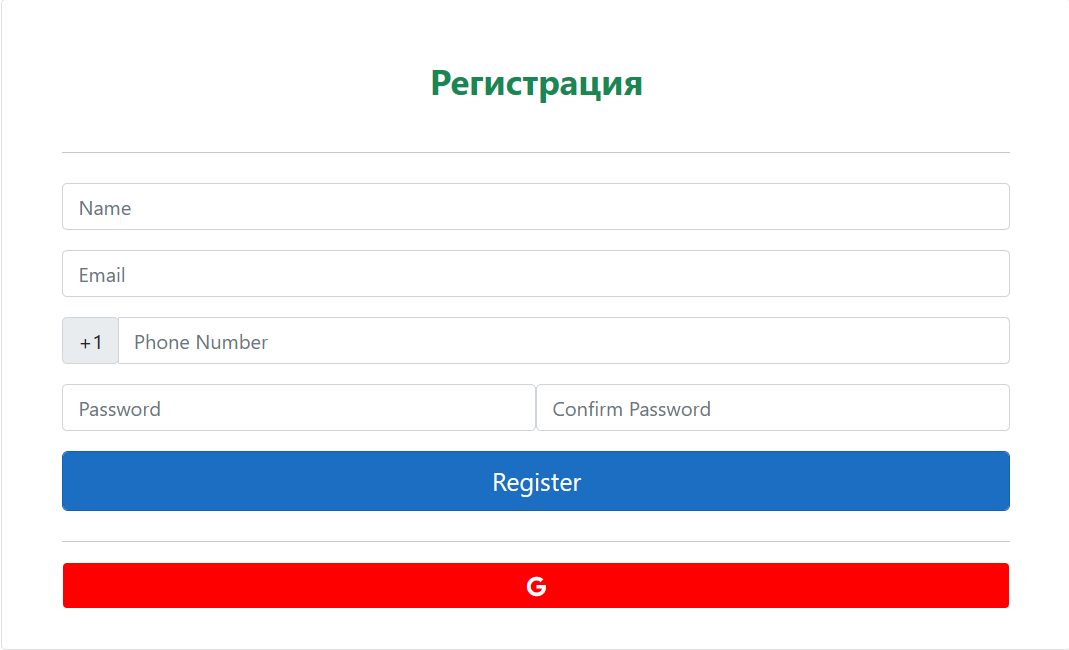
**Рисунок 2.2.8.2– Пункты навигационного меню**

В правой части навигационной панели находятся пункты для перехода к регистрации и авторизации на сайт (рисунок 2.2.8.3).

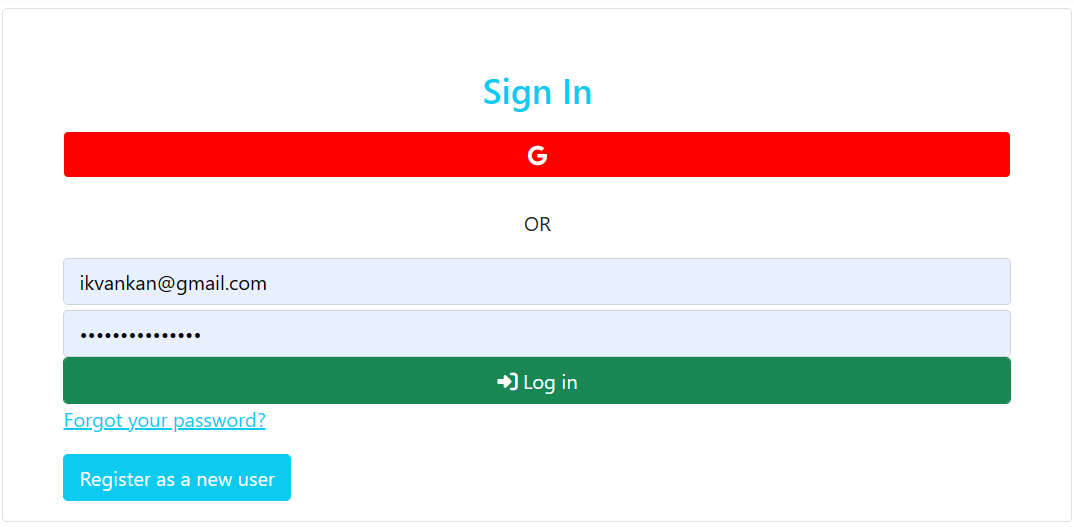
****

**Рисунок 2.2.8.3 – Пункты для авторизации и регистрации**

После перехода в разделы регистрации или авторизации пользователь переходят на соответсвующие страницы где он может зайти на сайт при помощи своего логина и пороля или при помощи Google аккаунта (рисунки 2.2.8.4 – 2.2.8.5).

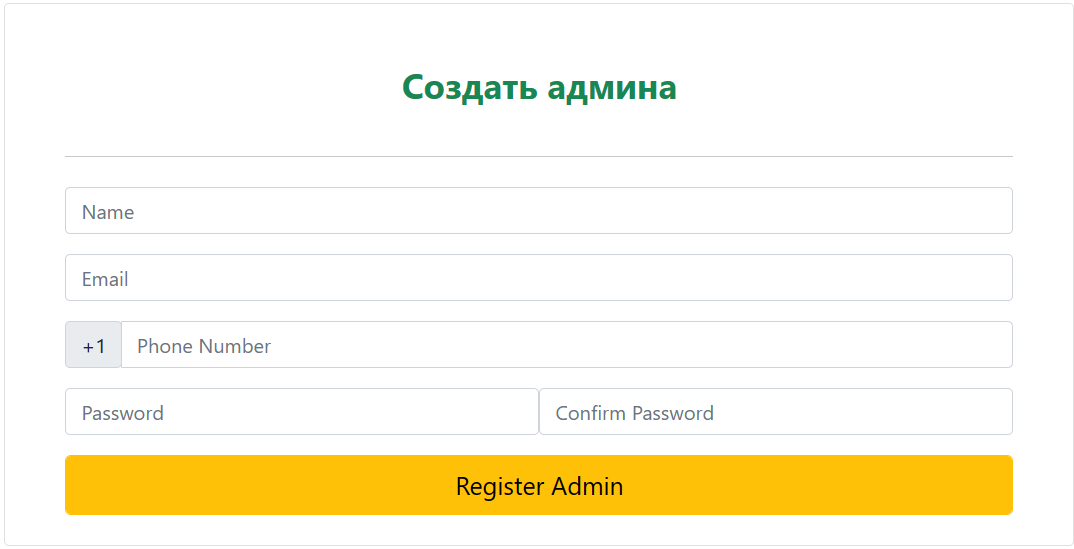
****

**Рисунок 2.2.8.4 – Страница регистрации**

****

**Рисунок2.2.8.5 – Страница авторизации**

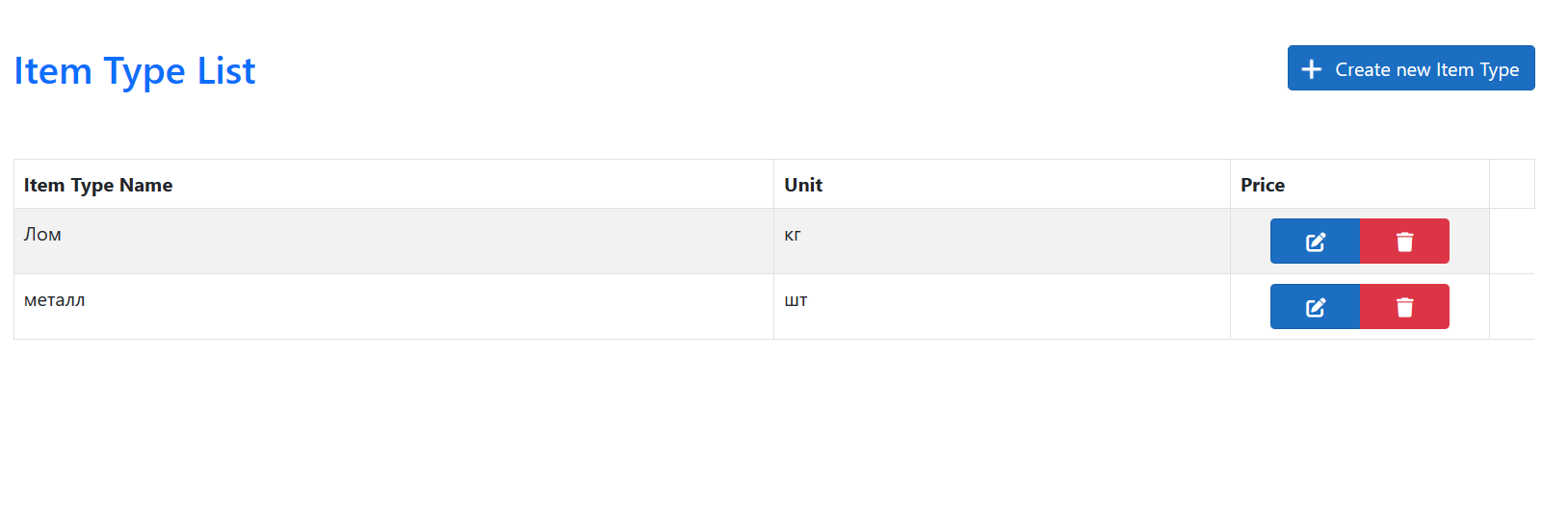
Также для администраторов в меню «Content menegment» доступна функция создания нового администратора (рисунок 2.2.8.6).

****

**Рисунок 2.2.8.6– Страница регистрации нового администратора**

Для админстратора также доступны пункты меню «Content menegment» которые позволяют настроить пунты, товары и их типы:

* cтраница для просмотра, создания, изменения и удаления типов товаров (рисунок 2.2.8.7);

****

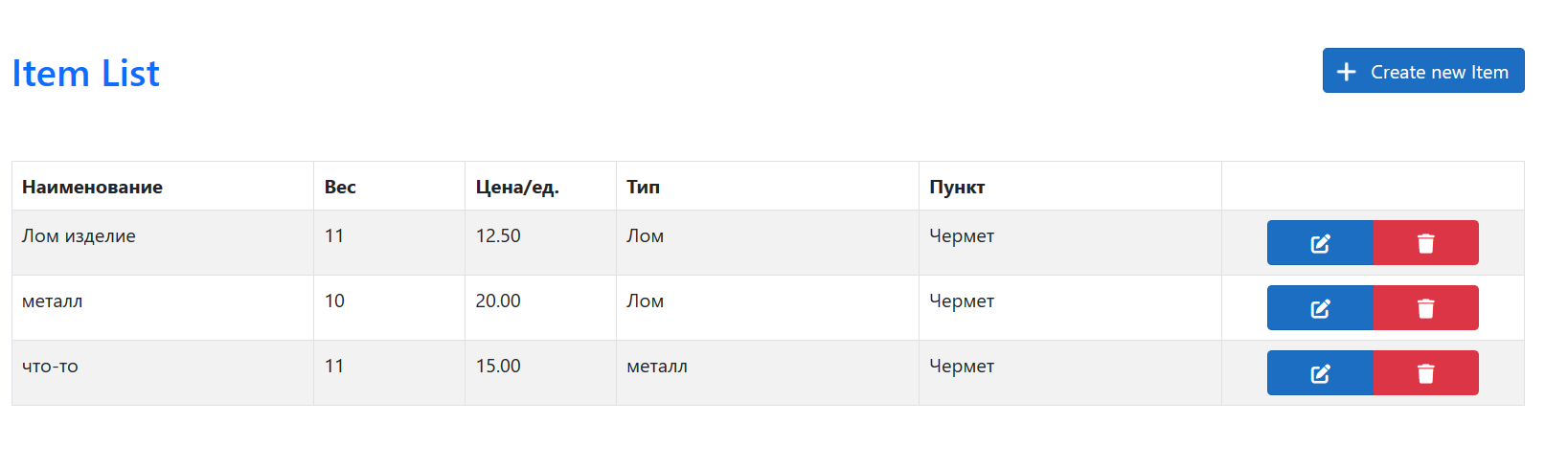
**Рисунок2.2.8.7 – Страница «Item Type List»**

* cтраница для просмотра, создания, изменения и удаления пунктов где хранится товар (рисунок 2.2.8.8);

****

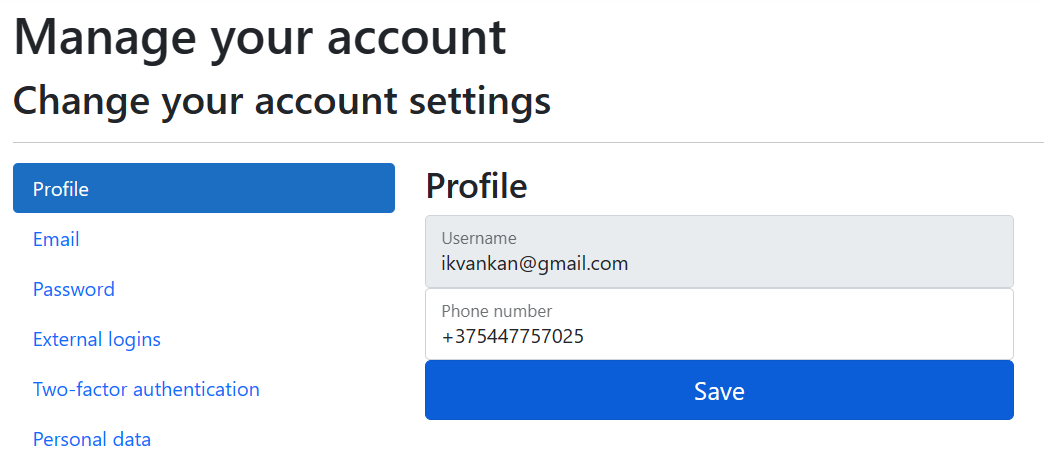
**Рисунок 2.2.8.8 – Страница «Punkt List»**

* cтраница для просмотра, создания, изменения и удаления товаров (рисунок 2.2.8.9).

****

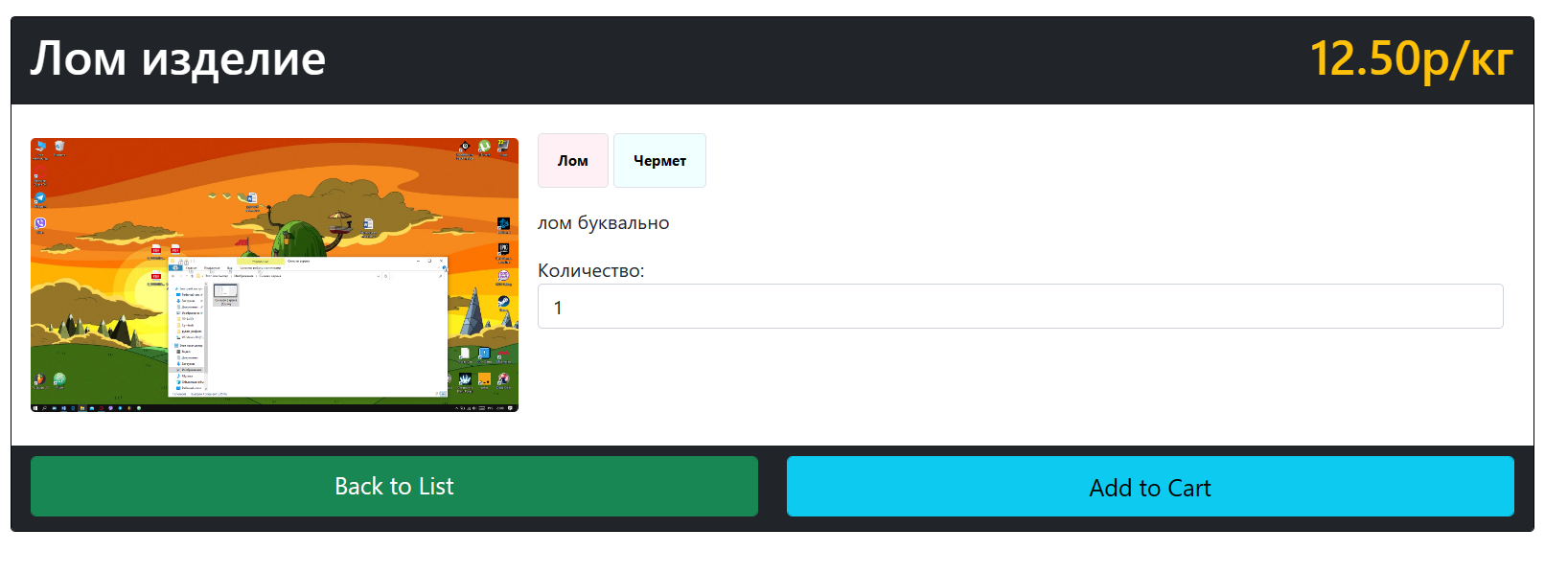
**Рисунок 2.2.8.9 – Страница «Item List»**

Пользователям при успешной авторизации становится доступна страница с настройками аккаунта (рисунок 2.2.8.10).

****

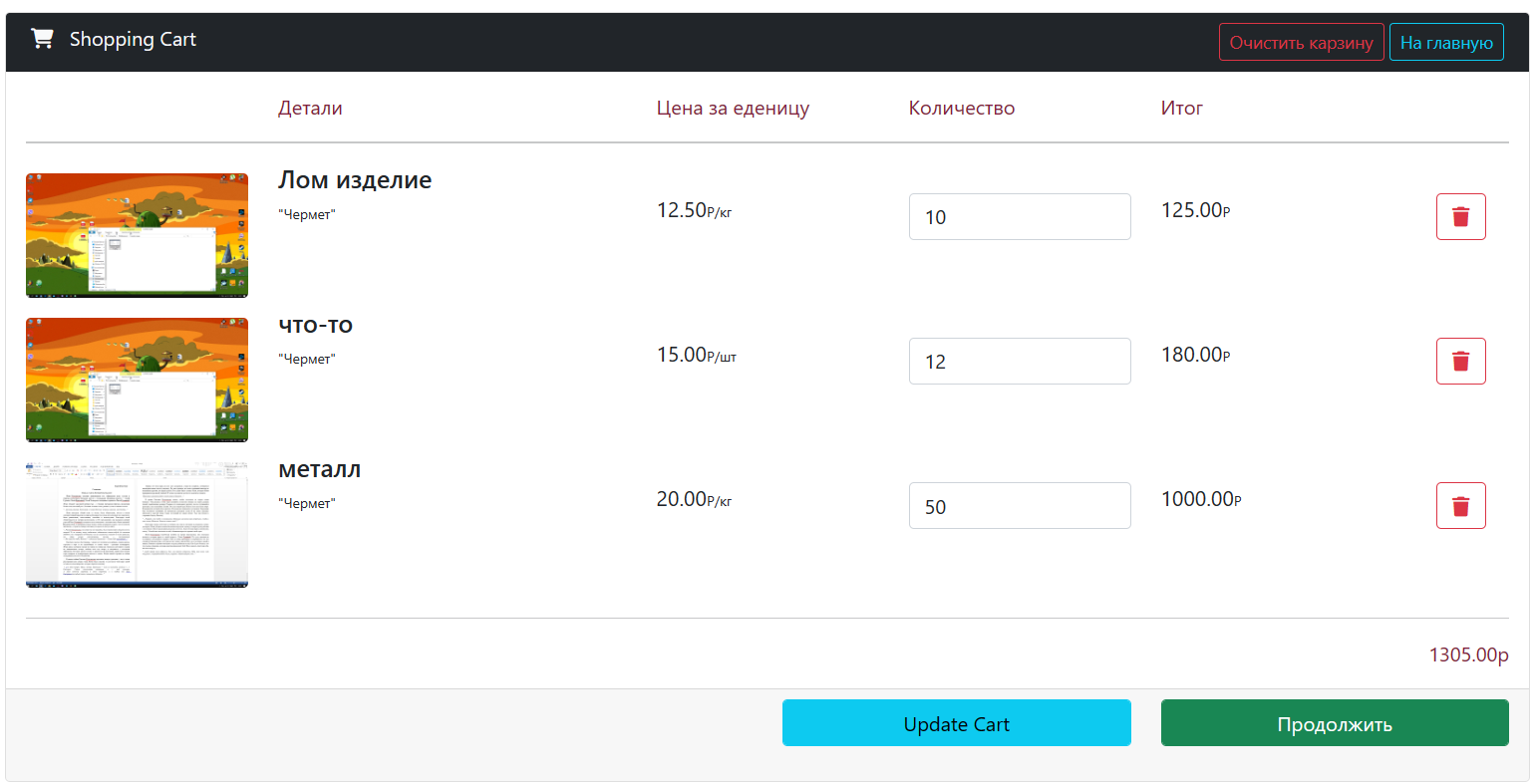
**Рисунок 2.2.8.10 – Страница настройки аккаунта**

При нажатии на кнопку «View Details» открывается окно в котором можно просмотерь информацию о товаре, выбрать количество желаемого товара и добавить его в корзину (рисунок 2.2.8.11).

****

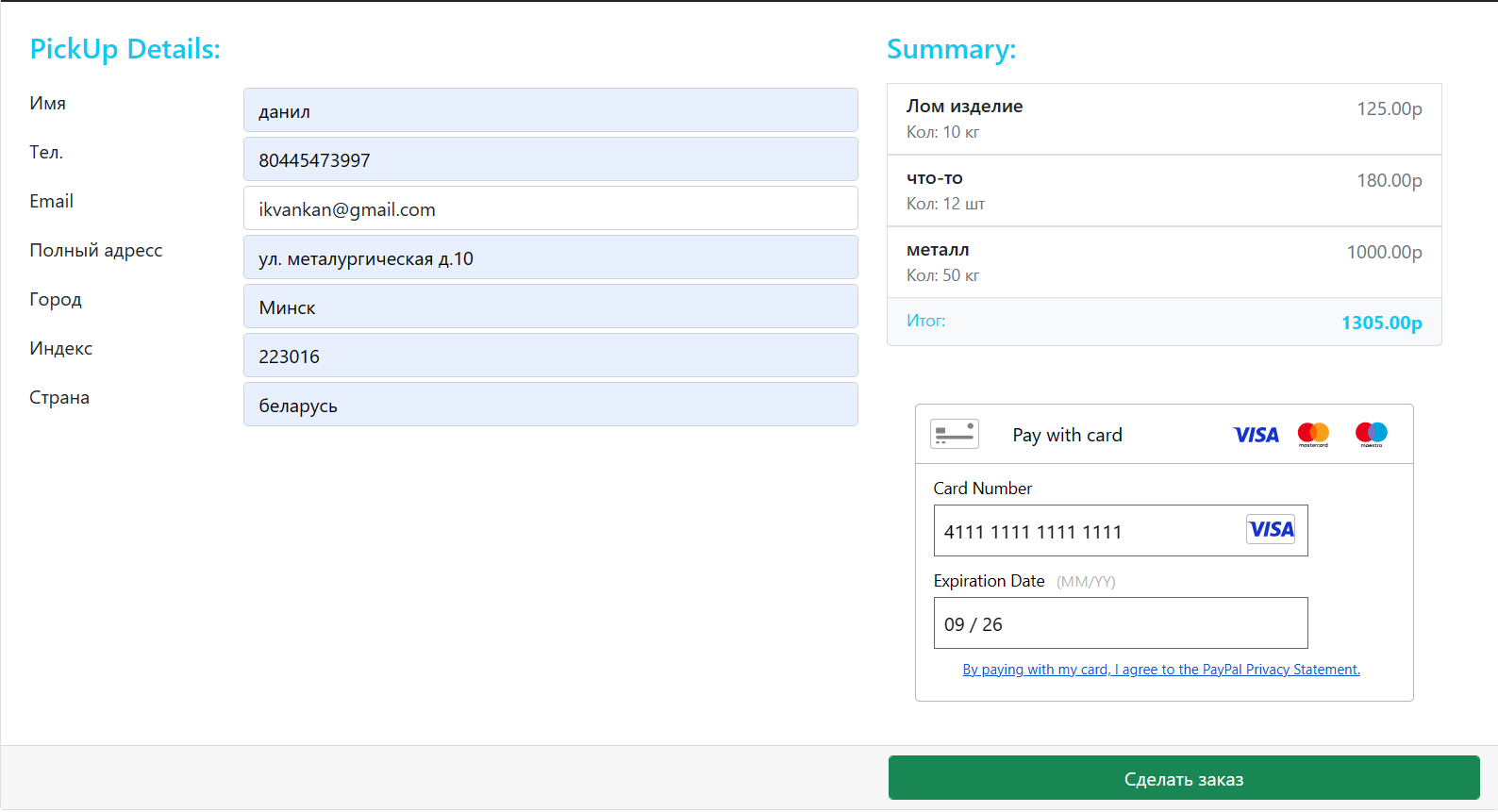
**Рисунок 2.2.8.11 – Детали о выбранном товаре**

После чего становится доступным интерфейс корзины в котором можно детальнее настроить желаемые товары и их количесвто, в корзине также подсчитвается расчёт стоимости товаров (рисунок 2.2.8.12).

****

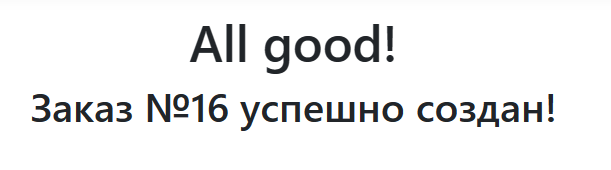
**Рисунок 2.2.8.12 – Корзина товаров**

После того как пользователь определился с желаемыми товарами он переходит на страницу, в которой необходимо заполнить поля для дальнейшего создания заказа, также здесь создается транзакция при помощи банковской карты клиента. После нажатия на кнопку «Сделать заказ» происходит отправка заказа на рассмотрение администратору и пользователю приходит уведомление на электронную почту (рисунок 2.2.8.13).

****

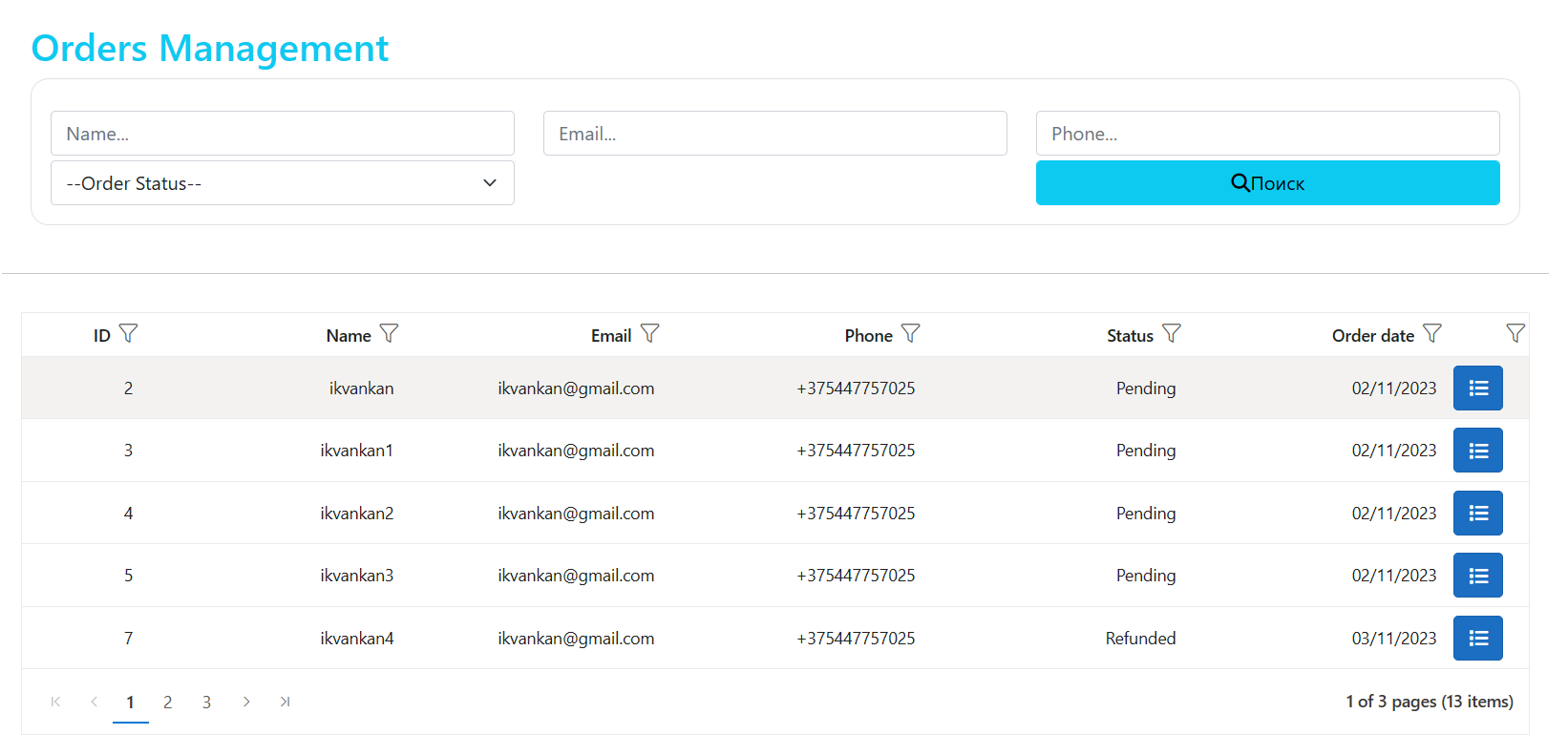
**Рисунок 2.2.8.13 – Заполнение деталей заказа**

После нажатия на кнопку «Сделать заказ» происходит отправка заказа на рассмотрение администратору и пользователю приходит уведомление на электронную почту (рисунок 2.2.8.14).

****

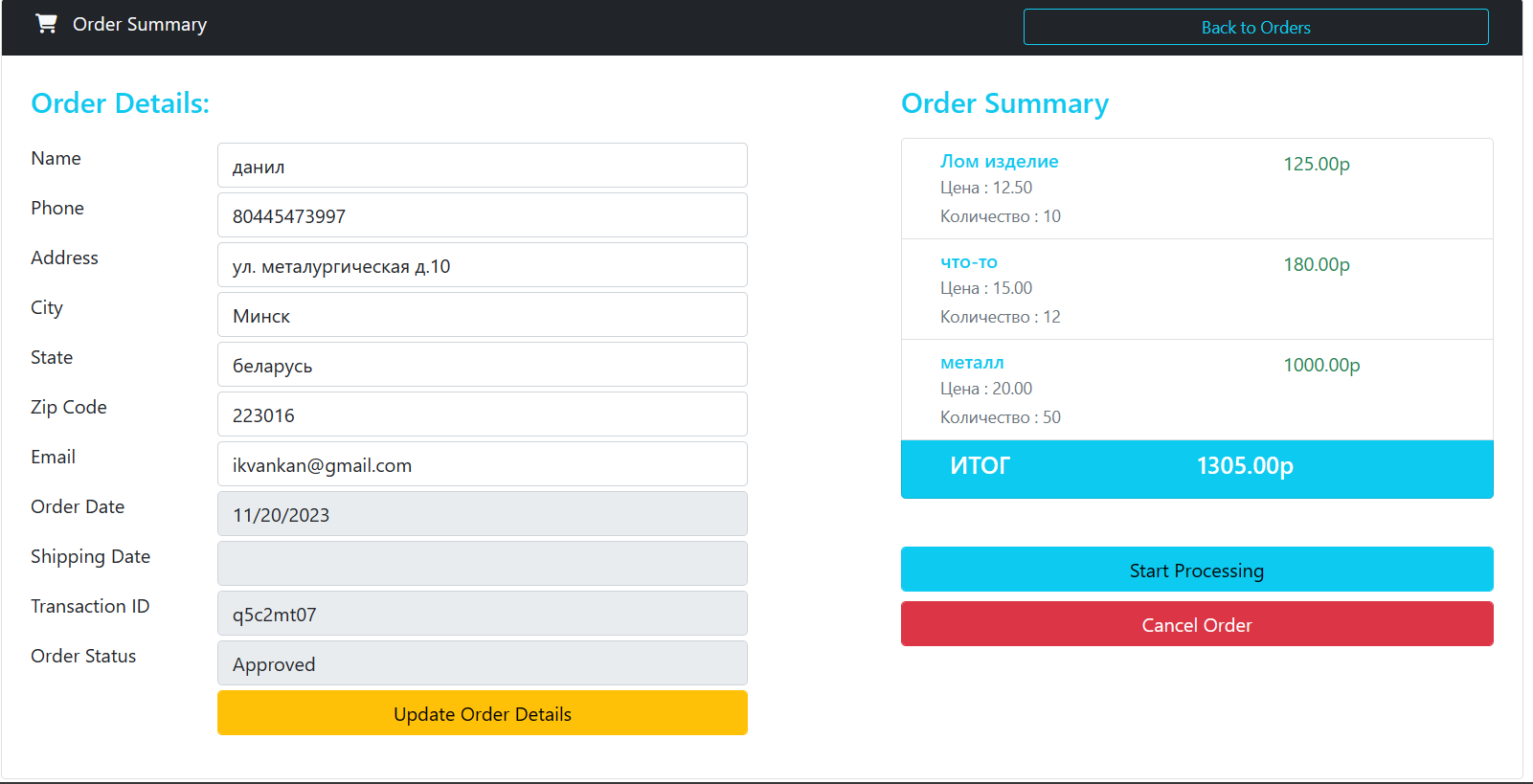
**Рисунок 2.2.8.14 – Успешно созданный заказ**

Далее администратор может просмотреть заказ в пункте меню «Content menegment» под названием «Order» (рисунок 2.2.8.15).

****

**Рисунок 2.2.8.15 – Страница для просмотра заказов**

При нажатии на детали заказа открывается страница, в которой можно управлять заказом (рисунок 2.2.8.16).

****

**Рисунок 2.2.8.16 – Детали заказа**

### ЗАКЛЮЧЕНИЕ

Базы данных являются важнейшей составной частью информационных систем, которые предназначены для хранения и обработки информации. Изначально такие системы существовали в письменном виде. Для этого использовались различные картотеки, папки, журналы, библиотечные каталоги.

Развитие средств вычислительной техники обеспечило возможность для создания и широкого использования автоматизированных информационных систем. Разрабатываются информационные системы для обслуживания различных систем деятельности, системы управления хозяйственными и техническими объектами, модельные комплексы для научных исследований, системы автоматизации проектирования и производства, всевозможные тренажеры и обучающие системы.

Современные информационные системы основаны на концепции интеграции данных, характеризующих большими объектами хранимых данных, сложной организацией, необходимостью удовлетворять разнообразные требования многочисленных пользователей.

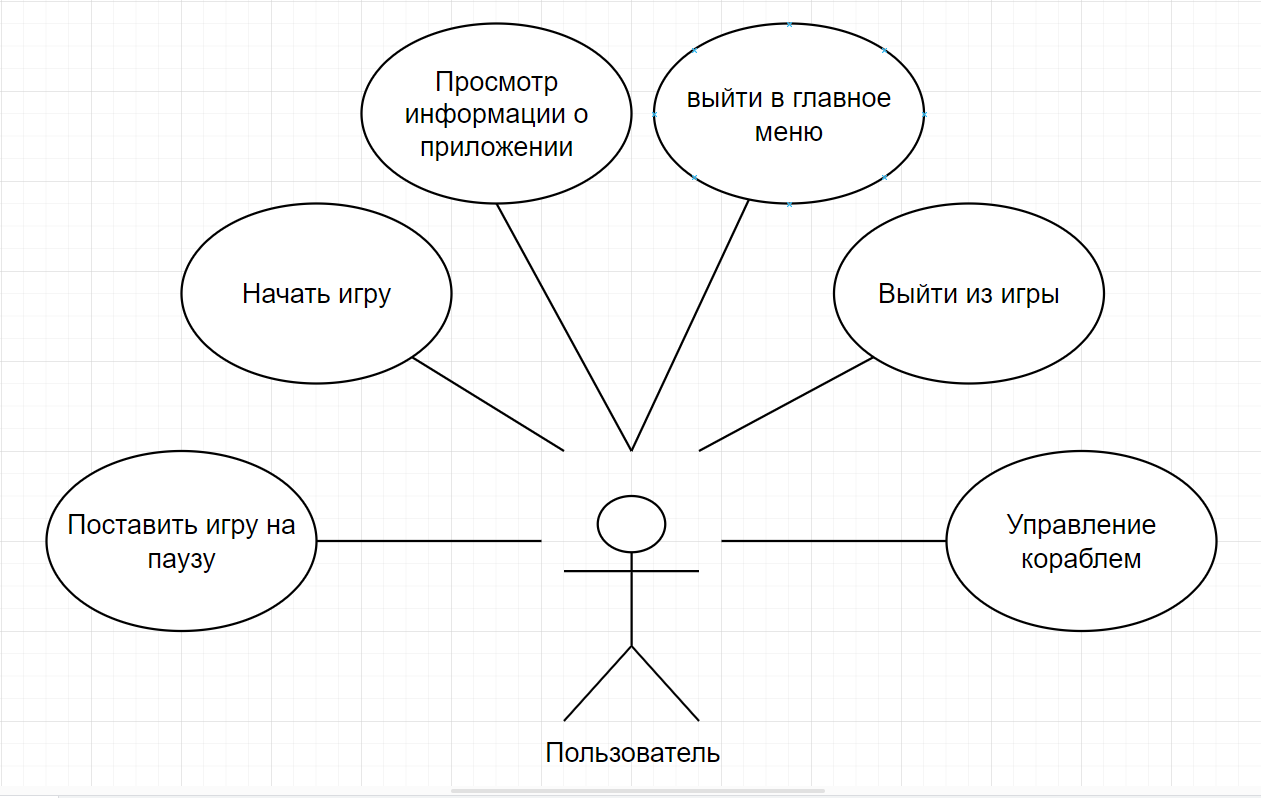
Для управления этими данными и обеспечения эффективности доступа к ним были созданы системы управления данными. В свою очередь нами была разработана автоматизированная информационная система киноцентра «Пирамида» и внешнее приложение к ней.

### СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

### ПРИЛОЖЕНИЯ

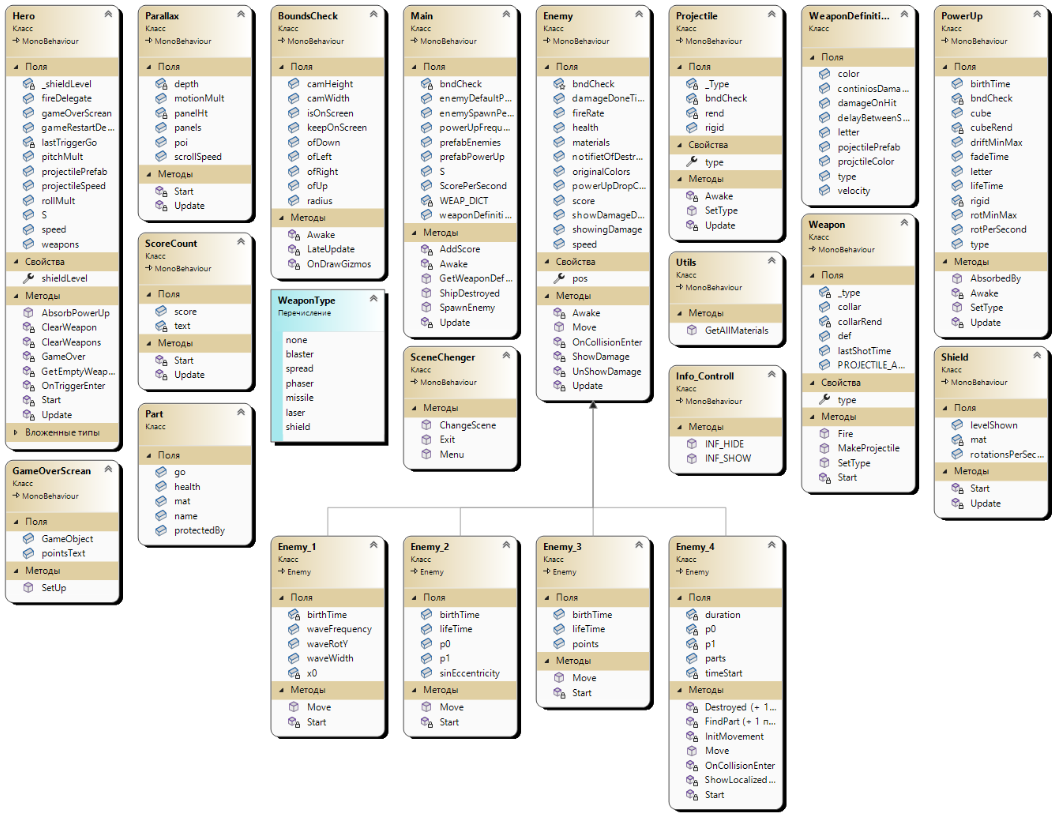
**ПРИЛОЖЕНИЕ А**

**ДИАГРАММА ВАРИАНТОВ ИСПОЛЬЗОВАНИЯ**

****

**ПРИЛОЖЕНИЕ Б**

**ДИАГРАММА КЛАССОВ**

****

**ПРИЛОЖЕНИЕ В**

**ЛИСТИНГ ПРИЛОЖЕНИЯ**

**ПРИЛОЖЕНИЕ Г**

**ПРЕЗЕНТАЦИЯ**