|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ \_ ИУ-КФ «Информатики и управления»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА \_\_\_ ИУ5-КФ «Системы обработки информации»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ***

***НА ТЕМУ:***

***\_\_Разработка web-сервиса помощи самозанятым\_\_***

***\_ для реализации товаров и услуг \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

Студент \_\_САПР.Б-81\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_**Халмедов В.М.\_\_\_

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель ВКР **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** \_\_\_\_Кириллов В.Ю.\_\_\_

(Подпись, дата) (И.О.Фамилия)

Нормоконтролер **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

*2020 г.*

# АННОТАЦИЯ

В данной выпускной квалификационной работе бакалавра поставлена задача разработки web-сервиса для самозанятых “HandyMady”.

Процесс разработки состоит из трех частей: научно-исследовательской, проектно-конструкторской и проектно-технологической. Объем работы составляет 64 страниц.

В научно-исследовательской части сформулировано техническое задание на разработку программной системы, описана и проанализирована предметная область разрабатываемого продукта, представлена концептуальная схема данных предметной области, сформулированы основные сценарии использования и обоснован выбор инструментов и платформ для разработки.

В проектно-конструкторской части была разработана структура программной системы, логическая схема данных, разработан пользовательский интерфейс и выбрана архитектура приложения.

В проектно-технологической части содержится порядок развертывания системы, технологические решения, поддерживающие эксплуатационный цикл программы и кратко описан интерфейс сервера.

# СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc42012355)

[1. Научно-исследовательская часть 5](#_Toc42012356)

[1.1. Техническое задание 5](#_Toc42012357)

[1.1.1. Введение 5](#_Toc42012358)

[1.1.2. Назначение разработки 5](#_Toc42012359)

[1.1.3. Требования к программному изделию 6](#_Toc42012360)

[1.1.4. Стадии и этапы разработки 8](#_Toc42012361)

[1.2. Описание предметной области 11](#_Toc42012362)

[1.3. Концептуальная схема данных 12](#_Toc42012363)

[1.4. Пользовательские сценарии использования 13](#_Toc42012364)

[1.5. Перечень задач, подлежащих решению в процессе разработки 23](#_Toc42012365)

[1.6. Обоснование выбора инструментов и платформы для разработки 24](#_Toc42012366)

[2. Проектно-конструкторская часть 26](#_Toc42012367)

[2.1. Разработка структуры программной системы 26](#_Toc42012368)

[2.2. Логическая схема данных 31](#_Toc42012369)

[2.3. Разработка пользовательского интерфейса 37](#_Toc42012370)

[2.4. Архитектура системы 41](#_Toc42012371)

[3. Проектно-технологическая часть 43](#_Toc42012372)

[3.1. Порядок развертывания системы 43](#_Toc42012373)

[3.2. Технологические решения, поддерживающие эксплуатационный цикл 46](#_Toc42012374)

[3.3. Краткое описание интерфейса сервера 47](#_Toc42012375)

[ЗАКЛЮЧЕНИЕ 54](#_Toc42012376)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 55](#_Toc42012377)

[ПРИЛОЖЕНИЕ А 58](#_Toc42012378)

# ВВЕДЕНИЕ

Объектом разработки данной выпускной квалификационной работы является web-сервис для самозанятых “HandyMady”, для облегчения рабочего процесса самозанятых лиц, в реализации товаров и услуг.

Актуальность данной темы обусловлена увеличением количества самозанятых лиц, не имеющих стабильную платформу для торговли товарами и услугами. Также из-за отсутствия специализированной платформы у покупателей имеется проблема поисках тех или иных продавцов или товаров. Для удовлетворения двух сторон требуется единая система, которая позволяет повысить эффективность работы с реализацией товаров и услуг и облегчить рабочий процесс самозанятым, а также обеспечить удобную торговую площадку с возможностью поиска для покупателей.

Для достижения поставленной цели были поставлены следующие задачи:

1. Проанализировать предметную область разрабатываемой системы;
2. Сформировать концептуальную и логическую модели хранимых данных;
3. Сформулировать основные сценарии использования;
4. Оформить описание процессов, протекающих в системе;
5. Определить и обосновать инструменты и платформу разработки;
6. Разработать структуру программной системы и выбрать архитектуру приложения;
7. Разработать и развернуть систему.

# Научно-исследовательская часть

# Техническое задание

# Введение

**Наименование программы**

Web-сервис для самозанятых “HandyMady”.

**Краткая характеристика области применения**

Данное программное обеспечение может быть использовано самозанятыми лицами с целью создавать свои магазины, где они могут выставлять свои работы на продажу для заработка, а также физическими лицами для покупки различных товаров.

**Основания для разработки**

Разработка ведется на основании данного технического задания. Этим документом подтверждается процесс разработки программного обеспечения на тему “Разработка web-сервиса помощи самозанятым для реализации товаров и услуг”.

# Назначение разработки

**Наименование темы разработки**

“Разработка web-сервиса помощи самозанятым для реализации товаров и услуг”.

**Функциональное назначение программы**

Функциональным назначением разработки указанного программного обеспечения является повышение эффективности работы, облегчения рабочего процесса, помощь в реализации товаров и услуг для самозанятых лиц, а также удобная торговая площадка для покупателей с возможностью поиска.

Критериями повышения эффективности будут служить:

* уменьшение затрат времени на поиск продавцов и товаров;
* удобный интерфейс для выставления товаров и для аукциона;
* возможность общения покупателей с самозанятыми;
* отслеживание доставки;

**Эксплуатационное назначение**

Подразумевается эксплуатация системы пользователями и самозанятыми, получающие доход от своей личной трудовой деятельности.

# Требования к программному изделию

**Требования к функциональным характеристикам**

Программная система должна представлять собой работоспособный сайт, реализующее следующие функции:

* функция регистрации для пользователей и продавцов;
* функция создания магазина;
* функция выставления товаров;
* функция ведения акций в магазине;
* поиск магазинов и товаров.

**Требования к надежности**

Система должна работать стабильно, не допускается прерываний работы системы, вызванных ее внутренними ошибками. Интерфейс системы не должен содержать элементов, назначение которых неочевидно, которые вводят пользователя в заблуждение или не несут функционального значения.

**Условия эксплуатации**

Система предназначена для использования в нормальных условиях. Специальных требований по эксплуатации системе не предъявляется.

**Требования к составу и параметрам технических средств**

Специальные требования к составу и параметрам технических средств отсутствуют.

**Требования к информационной и программной совместимости**

Описанная и разрабатываемая система должна включать в себя серверный компонент и клиентский интерфейс для взаимодействия с системой. Система должна быть совместима со всеми актуальными браузерами.

В качестве системы управления базой данных будет использоваться PostgreSQL.

Исходные коды серверной части должны быть реализованы на языке C# посредством свободно-распространяемого кросс-платформенного фреймворка для создания веб-приложений с открытым исходным кодом ASP.NET Core.

Визуальное представление должно быть реализовано на языках HTML и CSS в виде макетов.

В качестве среды разработки программы должна быть использована среда Visual Studio.

**Требования к программной документации**

Техническое задание должно быть оформлено в соответствии с требованиями ГОСТ. В результате выполнения выпускной квалификационной работы должны быть разработаны следующие материалы:

* Расчетно-пояснительная записка, содержащая 45-80 страницы и состоящая из научно-исследовательской, проектно-конструкторской и проектно-технологической частей;
* Графические материалы, которые состоят из 7 листов формата А1, содержащие релевантную информацию о разрабатываемой программном обеспечении.

# Стадии и этапы разработки

Содержимое разделов настоящего технического задания может быть изменено и дополнено по согласованию с руководителем.

**Стадии разработки**

Разработка должна быть проведена в пять стадии:

* Утверждение технического задания;
* Эскизный проект;
* Технический проект;
* Рабочий проект;
* Ввод в эксплуатацию.

**Этапы разработки**

Порядок разработки программной системы разбит на следующие стадии:

**Этап утверждения технического задания**

* Анализ и исследование предметной области;
* Разработка и утверждение технического задания на разработку программной системы.

Результатом работы на данном этапе является согласованное и утвержденное техническое задание на разработку системы и описание предметной области.

**Этап проектирования:**

* Оформление исследовательской части;
* Разработка концептуальной схемы данных,
* Формулировка и анализ основных сценариев использования программной системы;
* Обоснование выбора инструментария и платформ для разработки программной системы;

Результатом работы на данном этапе является оформленная исследовательская часть расчетно-пояснительной записки, разработанная концептуальная схема данных, сформулированные основные сценарии использования, а также обоснованный выбор инструментария и платформ для разработки программной системы.

**Этап технического проекта:**

* Оформление проектно-конструкторской части;
* Разработана структура программной системы;
* Разработка логической схемы данных;
* Разработка представления пользовательского интерфейса;
* Выбор архитектуры программной системы;
* Разработка технической версии проекта;

Результатом работы на данном этапе будет являться оформленная проектно-конструкторская часть, разработанная структура системы, логическая схема, представление пользовательского интерфейса, выбрана архитектура системы и разработана рабочая техническая версия программной системы.

**Этап рабочего проекта:**

* Разработка оставшегося функционала приложения;
* Отладка и тестирование рабочей версии проекта;
* Выбор технологических решений, поддерживающие эксплуатационный цикл программной системы.

Результатом работы на данном этапе является оформленная проектно-технологическая часть, рабочая версия программной системы и выбор технологических решений, поддерживающие эксплуатационный цикл программной системы.

**Этап ввода в эксплуатацию:**

* Формулировка развертывания системы;
* Первоначальная настройка системы.

Результатом работы на данном этапе является формулировка развертывания системы и спроектированный базовый функционал разрабатываемой системы.

# Описание предметной области

Самозанятость — это форма вознаграждения за труд или услугу напрямую от заказчиков. Самозанятыми являются лица, не имеющие работодателя, а также не имеющие наемных работников. То есть это люди, которые получают прибыль напрямую от своей трудовой деятельности.

Осуществлять свою рабочую деятельность самозанятые лица могут с помощью следующих интернет площадок:

* Социальные сети — это платформа онлайн, предназначенная для общения. На базе социальных сетей самозанятое лицо создает сообщество с целью продвижения товара и его дальнейшего сбыта. Самозанятое лицо, продавец, ведет каталоги, добавляя новый товар в наличие, ведет переписки с потенциальными покупателями, работает над продвижением своего товара. Оплата в социальных сетях чаще всего производится посредством онлайн переводов при помощи сторонних приложений банков. Это существенно затрудняет ведение финансовой отчетности и порождает проблему мошенничества.
* Интернет-магазин — это сайт, для торговли различными товарами в интернете. Магазины позволяют пользователям сформировать заказ на покупку онлайн. В интернет-магазинах оплата осуществляется при помощи посредников.

Необходимо разработать интернет-магазин, реализующий товары и услуги самозанятых. Система должна обеспечить удобный интерфейс продавцам для создания своего магазина с возможностью добавления товара в каталог и ведение акций, в то время как пользователи получат удобный интерфейс поиска товаров по категориям.

# Концептуальная схема данных

На основании представленной информации в предметной области можно выделить несколько сущностей (рисунок 1.1): Пользователь, магазин, товар, акция, заказ и позиция заказа.

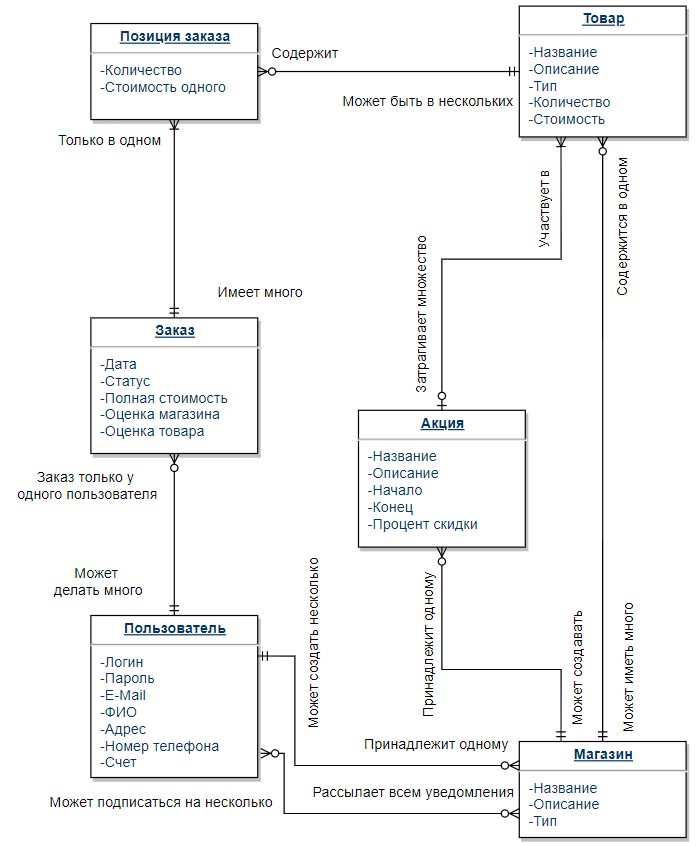


Рисунок 1.1 – Концептуальная модель данных

На рисунке 1.1 представлена концептуальная схема данных предметной области. Из данной схемы можно выделить следующие связи: существуют пользователи, которые могут делать много заказов, подписываться на чужие магазины и создавать множество своих. Владельцы магазинов могут создавать акции и добавлять в ассортимент новые товары. В одном заказе может быть множество позиций заказов, содержащие в себе товар и его количество.

# Пользовательские сценарии использования

После регистрации на сайте пользователь может как делать заказы в магазинах других продавцов, так и создать свои магазины с различными товарами и продавать их. У пользователя есть возможность просмотра каталога товаров по различным категориям. Пользователь как в роли покупателя, так и в роли продавца может создать несколько магазинов с различными категориями товаров. У каждого товара есть свое описание, кратко рассказывающее о том, что представляет из себя продукт, и фотографии, при помощи которых можно построить себе визуальное представление о товаре. Чтобы сделать заказ покупатель должен добавить товары в корзину, указать номер телефона, если он не указан в профиле пользователя и оплатить заказ. При этом количество единиц товара в магазине у продавца уменьшается в зависимости от того, сколько пользователь добавил. После того как товар был получен покупатель обязан оценить по пятибалльной шкале товар и магазин. Также пользователь может воспользоваться поиском товаров с различными фильтрами для точности. Для поиска товаров реализована поисковая строка. Поиск может проводиться как по названию, так и по различным категориям товаров. Покупатель может зайти на страницу магазина других пользователей, где может подписаться на рассылку новостей о появлении новый товаров и акций. UML диаграмма пользовательских сценариев, изображенная на рисунке 1.2, показывает возможные варианты действий покупателя.

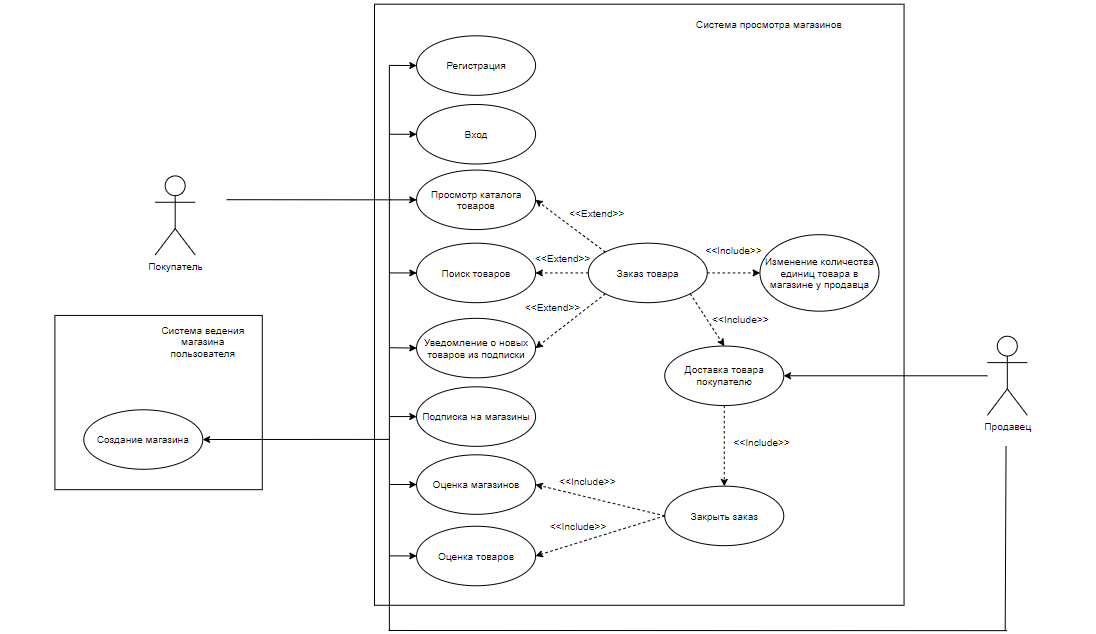


Рисунок 1.2 – UML диаграмма пользовательских сценариев для покупателя

Продавец имеет такие же пользовательские сценарии что и покупатель, но при этом он может заниматься менеджментом своих магазинов. Он может проводить различные акции индивидуально для каждого своего магазина с настройкой описания, даты начала и окончания. В профиле магазина владелец может видоизменить его вид, добавить товары на продажу и купить рекламу для продвижения товаров. UML диаграмма пользовательских сценариев (рисунок 1.3) показывает возможные варианты действий продавца.

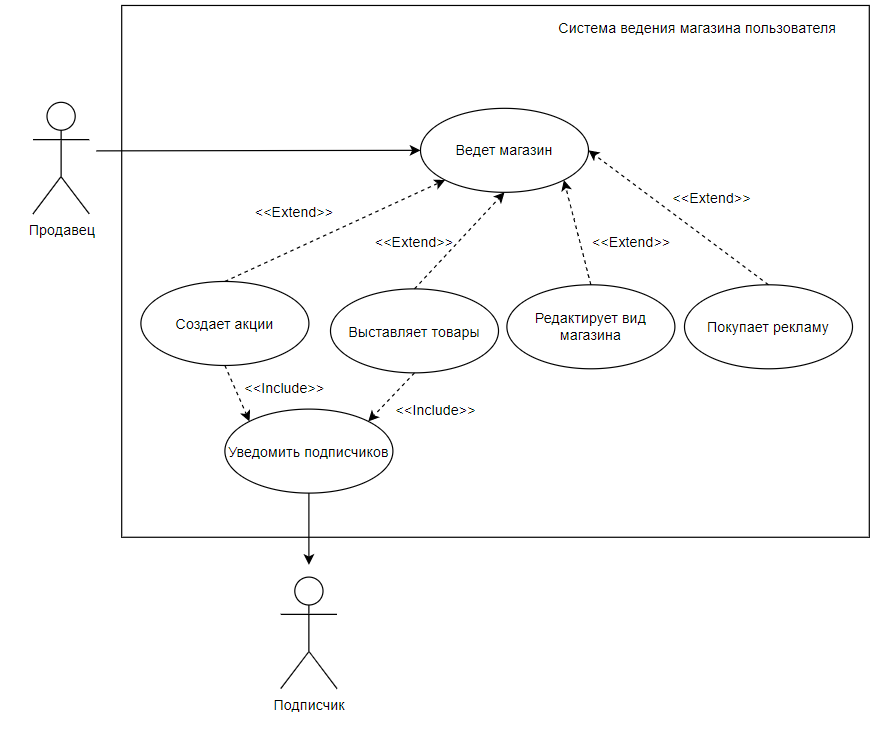


Рисунок 1.3 – UML диаграмма пользовательских сценариев для продавца

В таблицах 1.1 – 1.10 представлены основные пользовательские сценарии поведения системы.

Таблица 1.1 – Сценарий регистрации пользователя

|  |  |
| --- | --- |
| **Название** | Регистрация |
| **Краткое описание** | Пользователь регистрируется в системе для дальнейшего взаимодействия с ней. |
| **Участники** | Пользователь, система |
| **Базовый сценарий:**   1. Пользователь открывает сайт. 2. Пользователь нажимает кнопку «Регистрация» 3. Система отправляет пользователю форму регистрации. 4. Пользователь вводит данные во все поля формы. 5. Система отправляет данные на сервер для занесения пользователя в базу данных. | |
| **Постусловие** | Пользователь зарегистрирован в системе. |
| **Расширения:** | |
| **1** | Пользователь заполнил не все обязательные поля.  Система сообщит об ошибке и попросит их заполнить. |
| **2** | Имя пользователя занято  Система сообщит об ошибке и попросит выбрать другое. |
| **3** | Электронная почта заблокирована за нарушение правил.  Система уведомит пользователя о том, что его электронная почта была заблокирован из-за нарушения правил. |

Таблица 1.2 – Сценарий входа в систему

|  |  |
| --- | --- |
| **Название** | Вход в систему |
| **Краткое описание** | Пользователь входит в систему под своим логином и паролем для дальнейшего взаимодействия с ней |
| **Участники** | Пользователь, система |
| **Базовый сценарий:**   1. Пользователь открывает сайт и нажимает на кнопку «Войти в систему». 2. Система отправляет пользователю форму авторизации. 3. Пользователь вводит имя пользователя и пароль. 4. Система проверяет корректность ввода логина и пароля. 5. Пользователь авторизован в системе. | |
| **Постусловие** | Пользователь авторизовался в системе |
| **Расширения:** | |
| **1** | Неверный логин  Система уведомит об ошибке и попросит ввести корректный логин. |
| **2** | Неверный пароль  Система уведомит об ошибке и попросит ввести корректный пароль. |
| **3** | Пользователь заблокирован.  Система уведомит пользователя о том, что он был заблокирован из-за нарушения правил. |
| **4** | Пользователь заполнил не все обязательные поля.  Система сообщит об ошибке и попросит их заполнить. |

Таблица 1.3 – Сценарий добавления товара в корзину

|  |  |
| --- | --- |
| **Название** | Добавление товара в корзину |
| **Краткое описание** | Пользователь добавляет товар, который он хочет купить в корзину. |
| **Участники** | Пользователь, система |
| **Базовый сценарий:**   1. Пользователь пишет в строку поиска название товара и при желании изменяет настройки поиска. 2. Система выдает пользователю результат поиска с выбранными настройками 3. Пользователь выбирает в результатах поиска товар, который ему нужен. 4. Ознакомившись с описанием товара, информацией о доставке и о цене пользователь нажимает на кнопку «добавить в корзину». | |
| **Постусловие** | Пользователь добавил товар в корзину |
| **Расширения:** | |
| **1** | Отсутствие товара в наличии  Система уведомляет об этом пользователя об отсутствии товара. Также система уведомит о следующем поступлении если продавец указал информацию. |
| **2** | Пользователь был заблокирован создателем магазина  Система уведомит пользователя о том, что его он был заблокирован создателем магазина. Также система покажет сообщение содержащую причину блокировки пользователя. |

Таблица 1.4 – Сценарий оформления пользовательского заказа

|  |  |
| --- | --- |
| **Название** | Оформление пользовательского заказа |
| **Краткое описание** | Пользователь оформляет заказ и оплачивает его. |
| **Участники** | Пользователь, продавец, система |
| **Базовый сценарий:**   1. В меню корзины пользователь выбирает кнопку «оформить заказ» 2. Система списывает деньги со счета в профиле сервиса. 3. Система уведомляет продавца об отправке товара. 4. Продавец отправляет товар выбранным способом доставки | |
| **Постусловие** | Пользователь оформил заказ |

Таблица 1.5 – Сценарий закрытия заказа

|  |  |
| --- | --- |
| **Название** | Закрытие заказа |
| **Краткое описание** | После получения товара пользователь закрывает заказ и ставит оценку. |
| **Участники** | Пользователь, система |
| **Базовый сценарий:**   1. Пользователь выбирает заказ из списка 2. Пользователь нажимает кнопку «Закрыть заказ» 3. Система помечает заказ в базе данных как завершенный. 4. Система предлагает пользователю оценить товар и услуги магазина. | |
| **Постусловие** | Пользователь закрыл заказ |

Таблица 1.6 – Сценарий создание магазина

|  |  |
| --- | --- |
| **Название** | Создание нового магазина |
| **Краткое описание** | Пользователь создает новый магазин. |
| **Участники** | Пользователь, система |
| **Базовый сценарий:**   1. Пользователь нажимает на кнопку «создать магазин». 2. Пользователь заполняет все необходимые поля и выбирает тип магазина. 3. Система добавляет магазин в систему | |
| **Постусловие** | Пользователь создал новый магазин |
| **Расширения:** | |
| **1** | Неверное название магазина  Система уведомит пользователя о том, что название уже используется и попросит ввести новое название. |
| **2** | У пользователя была заблокирована возможность создавать таблицы за легкое нарушения правил.  Система уведомит пользователя о том, что у пользователя была заблокирована возможность создавать таблицы за легкое нарушения правил. Также система покажет сообщение содержащую причину блокировки пользователя. |

Таблица 1.7 – Сценарий поиска товаров

|  |  |
| --- | --- |
| **Название** | Поиск товаров |
| **Краткое описание** | Пользователь ищет товары для покупки. |
| **Участники** | Пользователь, система |
| **Базовый сценарий:**   1. Пользователь вводит название товара в строку поиска. 2. Пользователь при желании меняет категорию поиска. 3. Система предоставляет пользователю результаты поиска. | |
| **Постусловие** | Пользователь получил результаты поиска. |
| **Расширения:** | |
| **1** | Отсутствие товаров в системе, которые ищет пользователь  Система сообщит о том, что в системе отсутствует товар, который ищет пользователь и выдаст пустой результат поиска. |
| **2** | Отсутствие товаров, которые пользователь ищет по названию и в какой-то категории.  Система сообщит о том, что в системе отсутствует товар, который ищет пользователь и выдаст прочие товары из выбранной пользователем категории. |

Таблица 1.8 – Сценарий оценки магазина

|  |  |
| --- | --- |
| **Название** | Оценка магазина |
| **Краткое описание** | Пользователь оценивает магазин после закрытия заказа. |
| **Участники** | Пользователь, система |
| **Базовый сценарий:**   1. После получения уведомления от системы о закрытии заказа пользователь также получит сообщение, что ему нужно оценить магазин если он еще не заказывал из него товары. 2. Пользователь ставит любую оценку магазину по пятибалльной шкале. 3. Система меняет общий пятибалльный рейтинг у магазина. | |
| **Постусловие** | У магазина сменился рейтинг. |

Таблица 1.9 – Сценарий оценки товара

|  |  |
| --- | --- |
| **Название** | Оценка товара |
| **Краткое описание** | Пользователь оценивает товар после закрытия заказа. |
| **Участники** | Пользователь, система |
| **Базовый сценарий:**   1. После получения уведомления от системы о закрытии заказа пользователь также получит сообщение, что ему нужно оценить товар. 2. Пользователь ставит любую оценку товару по пятибалльной шкале. 3. Система меняет общий пятибалльный рейтинг у товара. | |
| **Постусловие** | У товара сменился рейтинг. |

Таблица 1.10 – Сценарий добавления товара в магазин

|  |  |
| --- | --- |
| **Название** | Добавления товара в магазин |
| **Краткое описание** | Пользователь добавляет товар в магазин. |
| **Участники** | Пользователь, система |
| **Базовый сценарий:**   1. Пользователь нажимает на кнопку «Добавить товар» в своем магазине. 2. Пользователь заполняет все необходимые поля и выбирает тип товара. 3. Система добавляет товар в магазин. | |
| **Постусловие** | Пользователь добавил товар в магазин |

# Перечень задач, подлежащих решению в процессе разработки

Задача проектирования такой системы может быть решена при помощи следующих средств разработки:

* Средством реализации пользовательских интерфейсов должно быть реализовано на языках HTML и CSS;
* Хранение информации должно быть реализовано при помощи системы управления базами данных PostgreSQL;
* Исходные коды должны быть реализованы на языке C# посредством свободно-распространяемого кросс-платформенного фреймворка ASP.NET Core.

Структурное проектирование системы осуществляется с помощью составления концептуальной и логической моделей хранимых данных. Описание этапов проектирования архитектуры приложения сопровождается описанием на естественном языке.

Перечень реализуемых функций:

* Возможность создавать свои магазины
* Функция ведения товаров
* Реализация поиска товаров по названию.
* Система рейтинга, базирующаяся на оценках пользователей.
* Предоставление пользователю информации товарах.
* Функция подписок на магазины с возможностью уведомления о поступлении новых товарах и акциях.
* Возможность уведомления о скидках на товары из списка желаемого.

# Обоснование выбора инструментов и платформы для разработки

Платформой для разработки серверной части был выбран ASP.NET Core, который предназначен для создания веб-приложений любой сложности и размеров.

ASP.NET Core является фреймворком с открытым исходным кодом благодаря чему предоставляет достаточно гибкий для контроль над проектом. Также с помощью данного фреймворка возможно создавать кроссплатформенные приложения благодаря тому, что он может работать поверх кроссплатформенной среды .NET Core и может быть развернута на таких популярных системах как Windows, Mac OS и Linux. Таким образом для развертывания веб-приложения можно использовать либо службы IIS, либо кроссплатформенный веб-сервер Kestrel. [8]

ASP.NET Core построен из относительно независимых компонентов благодаря чему возможно либо использование реализации этих компонентов, либо расширение их с помощью наследования или вовсе создать и применять свой функционал.

Благодаря тому, что фреймворк поддерживает модули, все необходимые дополнительные компоненты веб-приложения могут быть загружены через Nuget, который является пакетным менеджером.

Также, одним из главных особенностей платформы ASP.NET Core является применение паттерна MVC, который значительно упрощает разработку приложения.

В качестве среды разработки была выбрана Visual Studio 2019 благодаря полной поддержке проектов ASP.NET Core и наличию удобного инструментария.

Для хранения информации в системе был выбран PostgreSQL, которая является объектно-реляционной системой управления базами данных. Она является одной из наиболее развитых и является свободно распространяемой, а также славится своей надежностью и высокой производительностью. [2]

Данная система управления базами данных поддерживает декларативный язык программирования SQL. PostgreSQL поддерживает большой набор функций для работы с данными при этом сохраняя простоту эксплуатации.

Также, неотъемлемой частью PostgreSQL является pgAdmin, предоставляющий интуитивный интерфейс для работы с базами данных. Данный интерфейс является свободным кроссплатформенным программным обеспечением. [12]

Для создания макетов пользовательских интерфейсов были выбраны языки HTML и CSS. HTML является языком гипертекстовой разметки. Этот язык интерпретируется браузерами и состоит из элементов, которые могут быть применены к различным частям текста, для придачи им различных значений. [5]

CSS является языком таблиц стилей, позволяющий применять различные стили такие как шрифты и цвет к структурированным документам, например HTML. Обычно стили используются для создания и изменения элементов веб-страниц и пользовательских интерфейсов на языках HTML. Благодаря разделению стиля представления документов от их содержимого, CSS делает проще создание веб-страниц, а также обслуживание. [1][9]

# Проектно-конструкторская часть

# Разработка структуры программной системы

Приложение представляет собой клиент-серверное приложение структура которого изображена на рисунке 2.1.

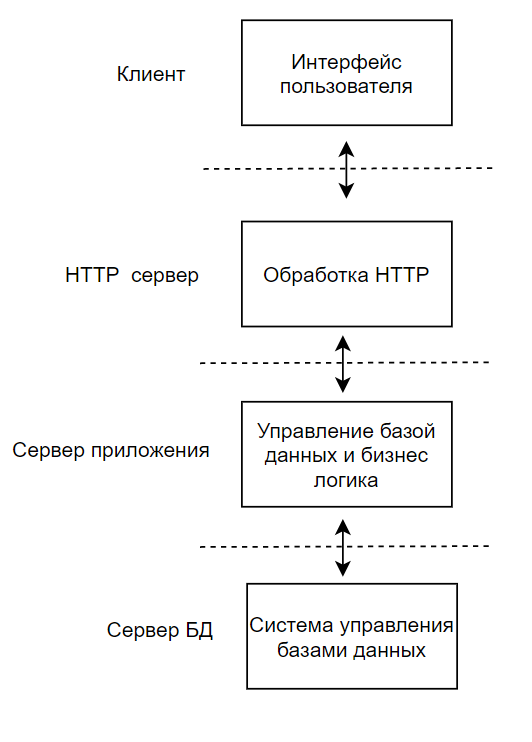


Рисунок 2.1 – Четырёхуровневое клиент-серверное приложение

Четырёхуровневое клиент-серверное приложение содержит в себе:

* Клиент, который представляет пользовательский интерфейс для взаимодействия с остальными уровнями приложения.
* HTTP сервер является сервером, который принимает различные запросы, а также отправляет ответы, вместе с HTML-страницей, изображением или прочими данными,
* Сервер приложения, на котором осуществляется работа с серверными компонентами разрабатываемой системы. На данном уровне осуществляется вся бизнес-логика и управление базой данных.
* Сервер БД обслуживает и управляет базами данных, а также отвечает за сохранность данных.

Последовательность разработки системы:

1. Проектирование базы данных.
2. Создание логики серверной части.
3. Создание интерфейса.
4. Заполнение базы данных.
5. Тестирование.

Основным бизнес-процессом в разрабатываемой системе является обработки заказа. Для формализации бизнес-процессов и действий проектируемой системы была использована нотация IDEF0. [3]

Контекстной диаграммой является модель, представляющая из себя набор различных иерархических действий. Каждое действие делает различные преобразования над некоторым объектом или набором объектов. Самом высшим действие иерархии называется действием контекста, которое является уровнем, описывающий непосредственно саму систему. Уровни ниже действия контекста являются декомпозициями и представляют из себя подпроцессы родительского действия. [4]

На рисунке 2.2 изображена контекстная диаграмма, описывающая процесс обработки заказа. Входным параметром являются позиции товаров. Информация о доставке и база данных системы отвечают за управление в данном процессе. Механизмы и действующие лица для выполнения данной работы – система, продавец и пользователь. На выход данного процесса поступают доставленный товар и оценка покупателя.

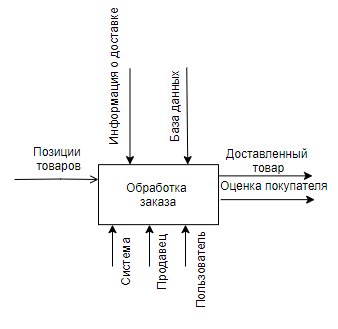


Рисунок 2.2 – IDEF0 контекстная диаграмма бизнес-процесса «обработки заказа»

На рисунке 2.3 представлена декомпозиция функции «обработка заказа» для описания каждой подсистемы и их взаимодействие. Процесс обработки заказа можно разделить на пять последовательных этапов:

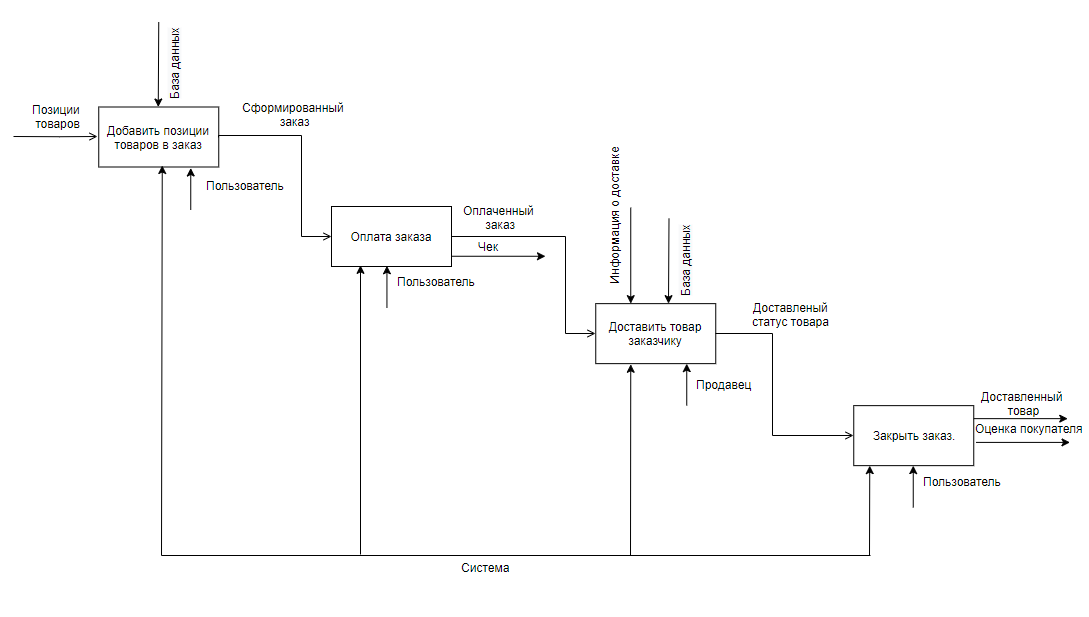


Рисунок 2.3 – IDEF0 диаграмма декомпозиции бизнес-процесса «обработки заказа»

1. Создание заказа.

На данном этапе система добавляет все созданные позиции с товарами в один заказ. Входным параметром являются позиции заказа. За управление в данном процессе отвечает база данных. Механизмы и действующие лица для выполнения данной работы – система и пользователь. В результате выполнения данного процесса на выход поступает сформированный заказ.

1. Оплата товара.

После того как пользователь закончить составлять свой заказ система предоставляет пользователю счет, который тот должен оплатить. Входным параметром является сформированный заказ. Механизмы и действующие лица для выполнения данной работы – это пользователь и система. В результате выполнения работы данного процесса поступают оплаченный заказ, а также чек.

1. Доставка товара.

После получения информации о том, что заказ был оплачен система уведомляет продавца об отправке товара выбранным способом доставки заказчику. За управление в данном процессе отвечает база данных и информация о доставке. Входным параметром является оплаченный заказ. Механизмы и действующие лица для выполнения данной работы - продавец и система. В результате выполнения работы данного процесса будет доставленный статус товара.

1. Закрытие заказа.

На последнем этапе пользователь получает товар, уведомляет систему о его получении и ставит оценку товару и магазину. Входным параметром является доставленный статус товара. Механизмы и действующие лица для выполнения данной работы – пользователь и система. Результатом выполнения работы данного процесса будет являться доставленный товар и оценка пользователя.

# Логическая схема данных

Модель «сущность-связь» представляет собой графическую нотацию, которая основывается на блоках и соединительных линиях. С их помощью можно описать объекты модели и отношения между ними. ER имеет несколько основных понятий, таких как сущности, атрибуты и связи. На рисунке 2.4 представлена логическая схема базы данных разрабатываемой системы. [6]

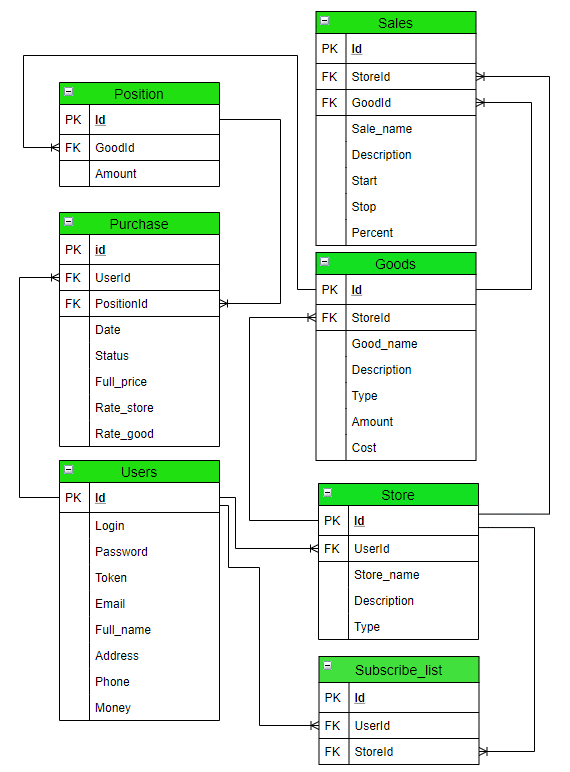


Рисунок 2.4 – Логическая схема базы данных

В результате анализа предметной области на данном этапе разработки системы были выделены следующие сущности:

Сущностью для хранения данных о покупателях и продавцах является Пользователь. Он содержит в себе такие атрибуты как фамилия, имя и отчество, адрес для доставки, контактный номер и денежный счет в онлайн-магазине. Будучи покупателем, пользователь может сделать множество заказов в различных магазинах, а также подписаться на новостную рассылку о новых товарах и акциях. Чтобы стать продавцом пользователь может создать несколько магазинов. Также есть поля для хранения данных о логине, пароле для входа на сайт, E-Mail для восстановления профиля и токен, наличие которого показывает зашел ли пользователь в систему. Наличие магазина у пользователя решает является ли пользователь продавцом

Следующей сущностью разрабатываемой системы является Магазин. Он содержит в себе атрибуты названия, описания и типа, означающий чем он занимается. В один магазин может быть добавлено множество товаров. Продавец–владелец от имени магазина может создавать акции на свои товары. Также магазина есть автоматическая рассылка уведомлений при создании новой акции и при добавлении нового товара.

Сущностью для наполнения магазина является Товар, реализующий поля для хранения данных о предметах, которые пользователи продают. Он содержит в себе название, описание, тип товара, стоимость одного экземпляра и количество на продажу. Если количество равно NULL, значит товар либо удален продавцом, либо скрыт от возможности покупки. Связь между товаром и магазином подразумевает, что у одного магазина может быть множество индивидуальных товаров, то есть один товар не может продаваться в двух магазинах.

Сущность Акция реализует поля для хранения данных описывающие скидки, которые делают продавцы в своих магазинах: название события, описание, дата начала и окончания акций и процент скидки на товары. Владелец может сделать множество акций на свои товары, при этом акция может принадлежать только одному магазину.

Для описания пользовательских заказов товаров есть сущность Заказ, хранящий в себе такие атрибуты как дата заказа, статус, который описывает состояние заказа, полная стоимость, оценка магазина и товара. Оценка будет играть ключевую роль при составлении рейтинга. Заказ может быть только у одного пользователь, при этом в одном заказе может быть множество позиций заказа.

Сущность Позиция заказа описывает возможность наличия в одном заказе нескольких товаров. В данной сущности имеется атрибут, который хранит данные о количестве товаров. В одном заказе может быть множество позиций, при этом в позиции заказа может быть только один товар.

На рисунке 2.5 изображена инициализация подключения к базе данных в настройках проекта ASP.NET Core. [10]

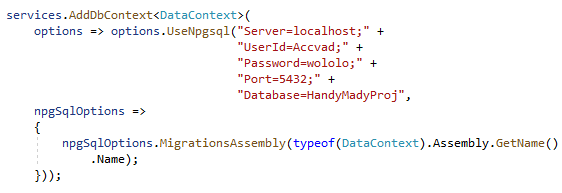


Рисунок 2.5 – Инициализация подключения к БД

Веб-приложения ASP.NET Core получают доступ и управляют данными через объекты C#, которые называются моделями. Модели определяют структуру хранимых данных, включая типы полей, их максимальный размер, значения по умолчанию и т. д. На рисунках 2.6-2.8 изображены структуры данных пользователя, магазина и товаров. На рисунках 2.9-2.10 изображены функции обработки запроса для регистрации и для входа в систему. На рисунке 2.11 изображен репозиторий получающий все товары из магазина, а на 2.12 изображен репозиторий, который возвращает товары, названия которых совпадают со строкой. [11]

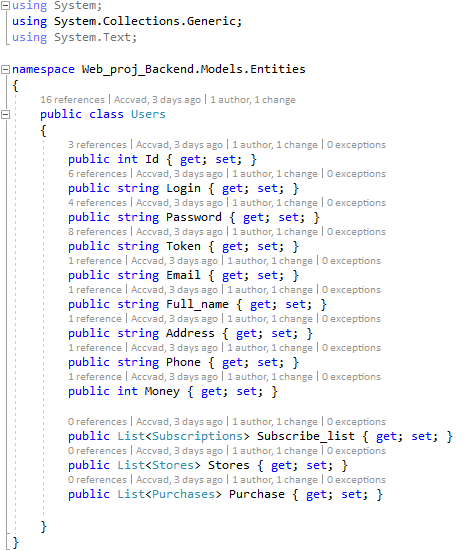


Рисунок 2.6 – Структура данных “Users”

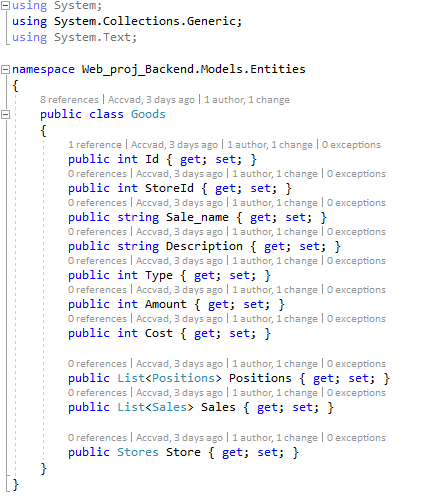


Рисунок 2.7 – Структура данных “Goods”

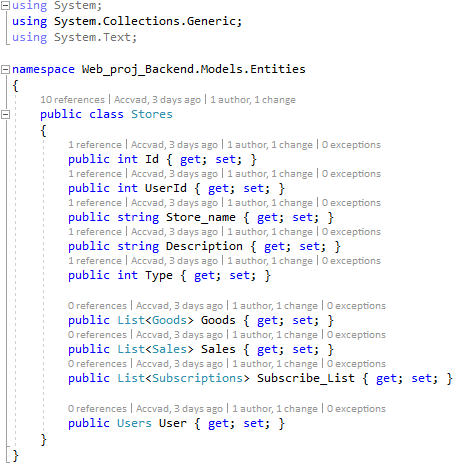


Рисунок 2.8 – Структура данных “Store”

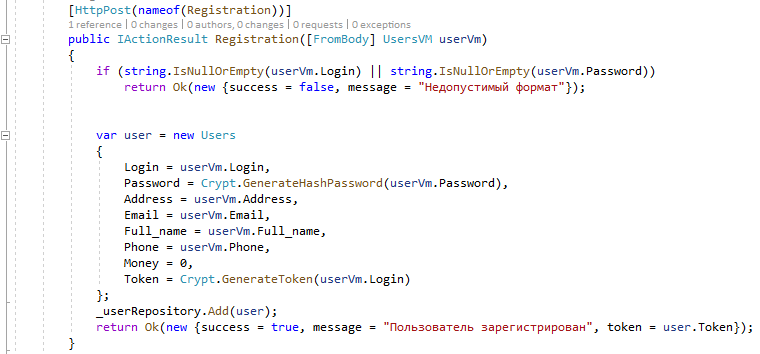


Рисунок 2.9 – Функция обработки запроса для регистрации

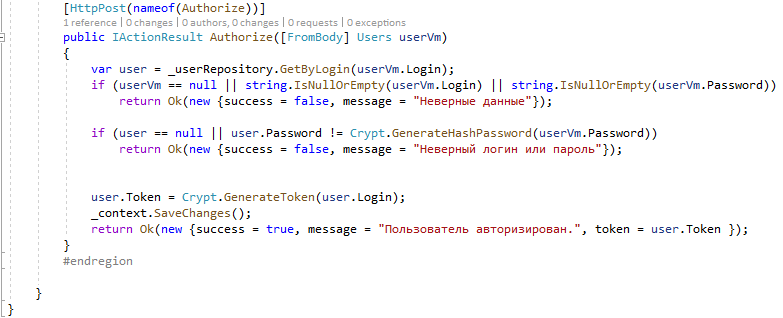


Рисунок 2.10 – Функция обработки запроса для входа в систему

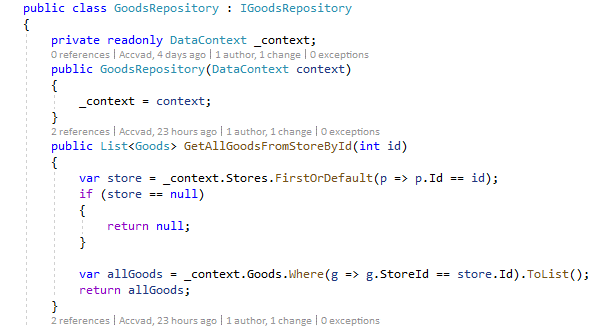


Рисунок 2.11 – Репозиторий, который возвращает все товары из магазина

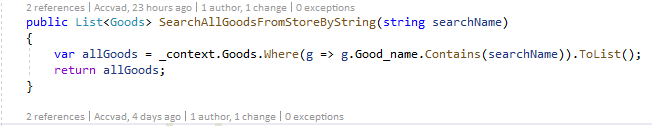


Рисунок 2.12 – Репозиторий, который возвращает товары, названия которых совпадают со строкой

# Разработка пользовательского интерфейса

Для разработки пользовательского интерфейса были созданы следующие макеты на языках HTML и CSS:

* Главная страница;
* Форма регистрации;
* Форма для входа в аккаунт;
* Страница с товаром;
* Интерфейс корзины.

На рисунке 2.13 изображен макет главной страницы магазина, на которую попадает пользователь при заходе на сайт.

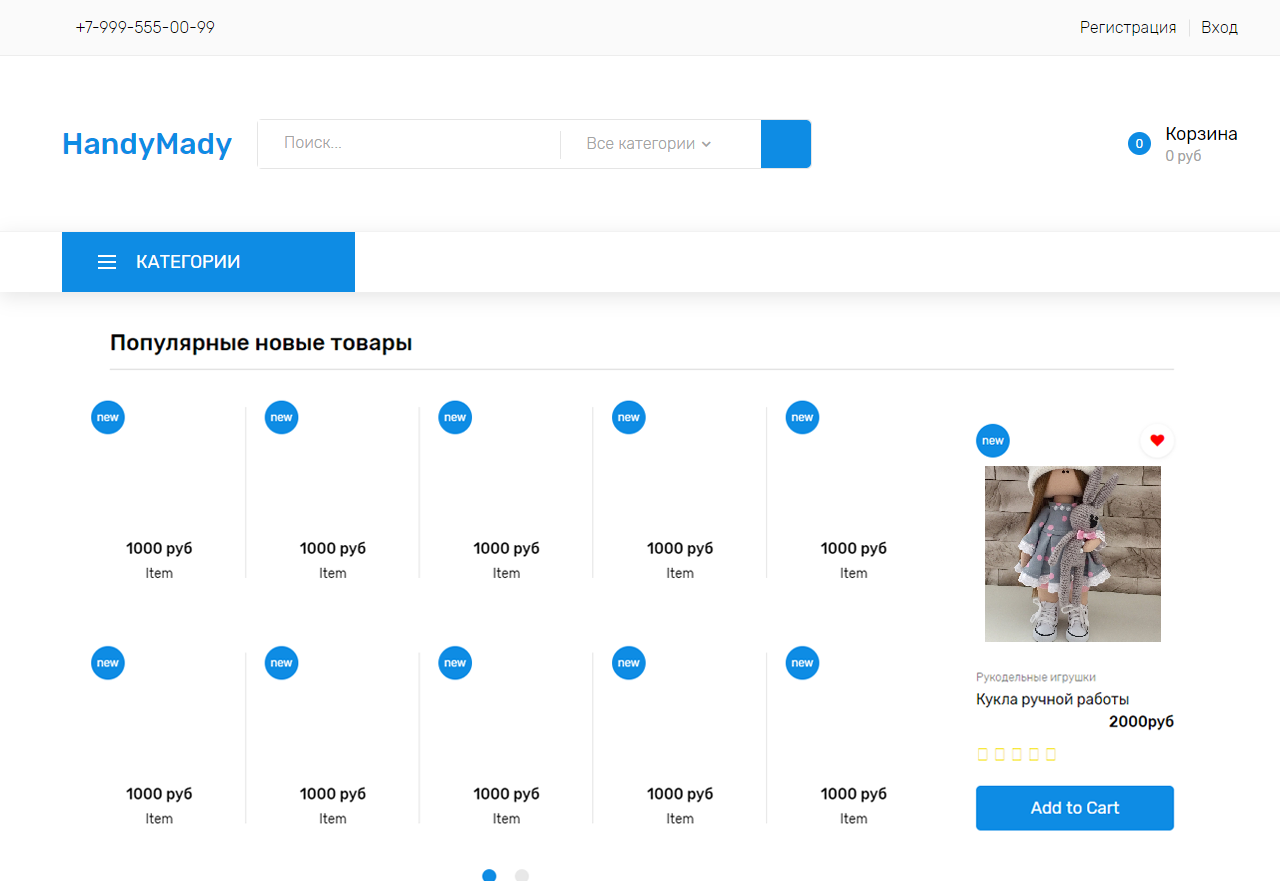


Рисунок 2.13 – Главная страница

На этой странице показаны популярные новые товары, которые можно добавить в корзину или открыть подробности. Также есть меню просмотра товаров по категориям и поиск. Для регистрации и входа предусмотрены соответствующие кнопки в правом верхнем углу.

При нажатии на кнопку «Регистрация» откроется форма для заполнения необходимых данных (рисунок 2.14).

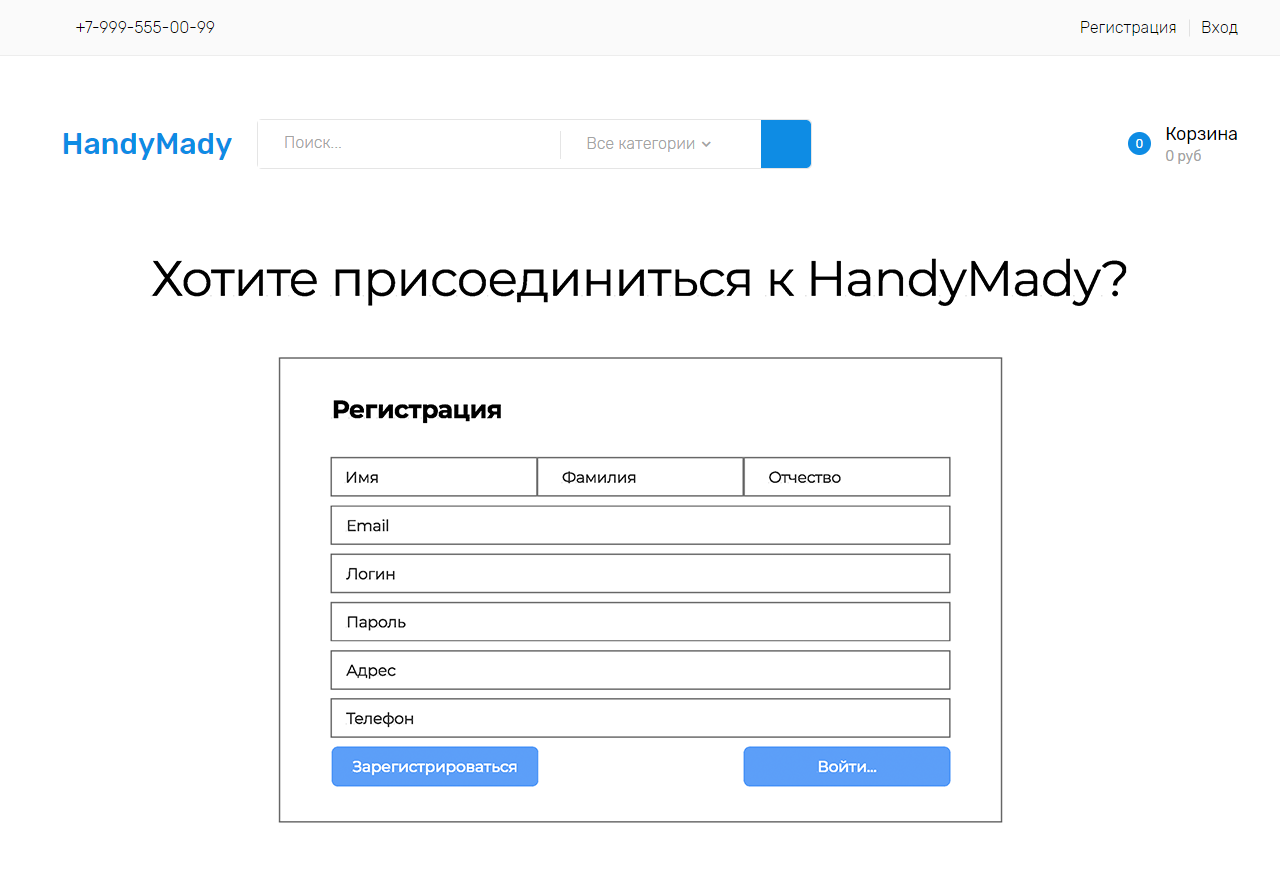


Рисунок 2.14 – Форма регистрации

Для вхождения в систему необходимо нажать на кнопку «Вход», после чего откроется форма для вхождения в систему. (Рисунок 2.15)

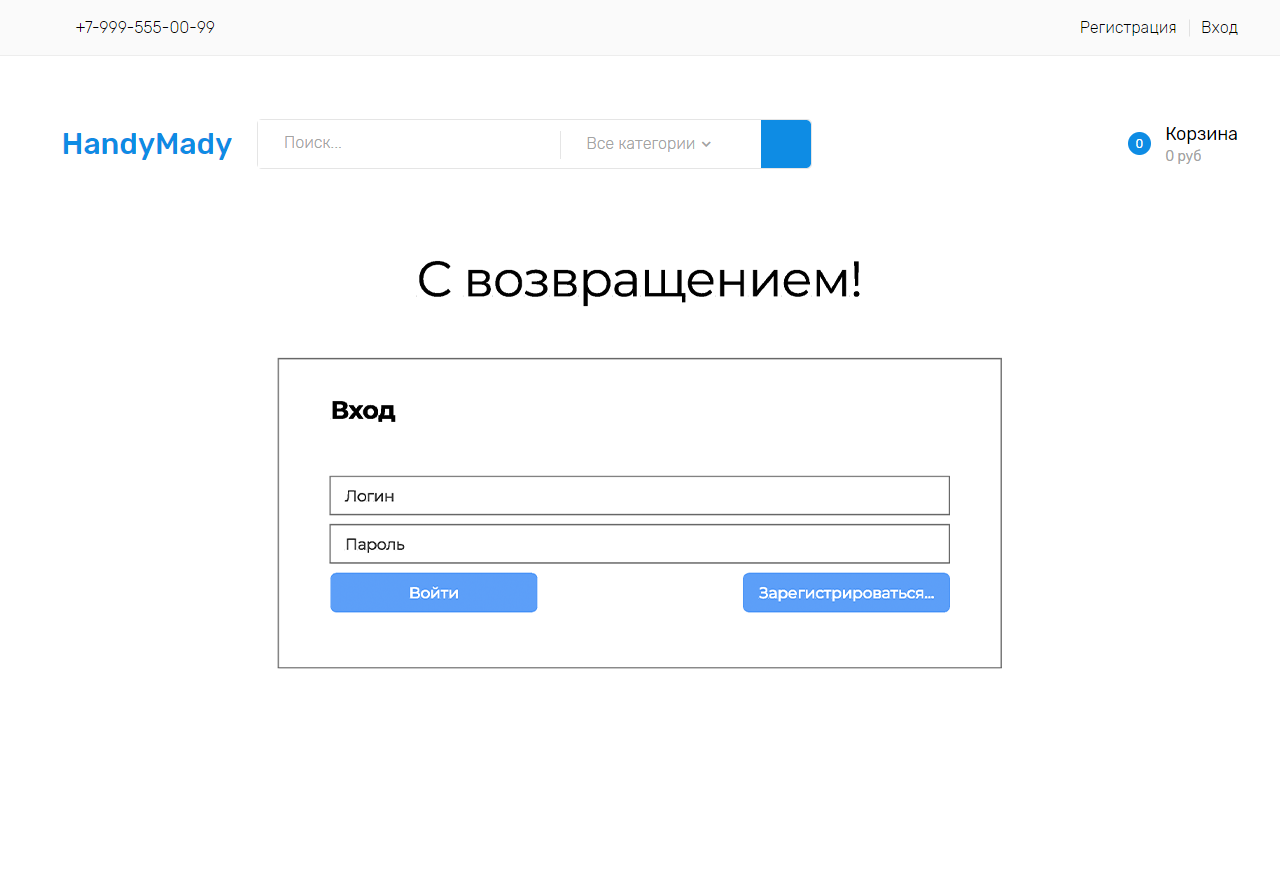


Рисунок 2.15 – Форма для входа

При нажатии на любой товар откроется его подробное описание и расширенная фотография (Рисунок 2.16).

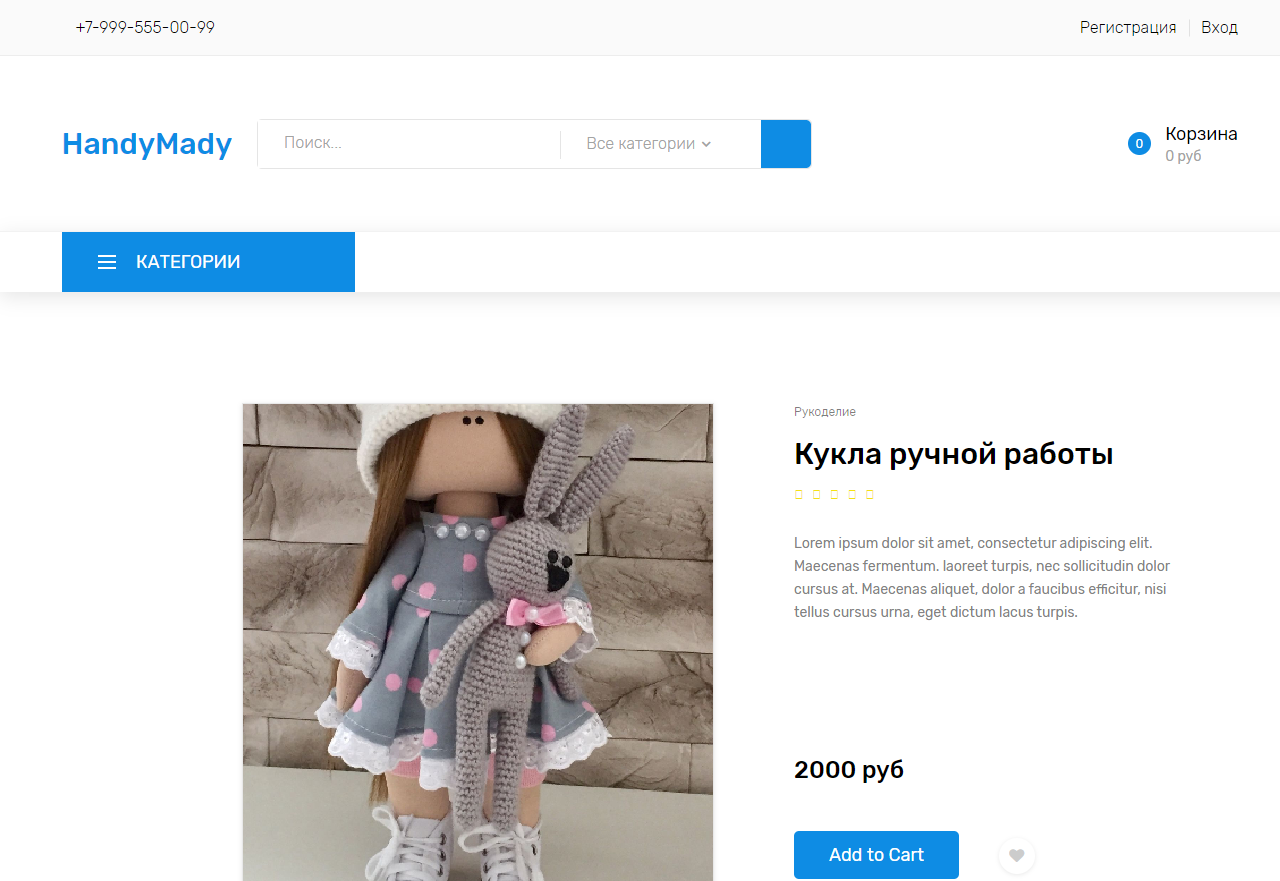


Рисунок 2.15 – Страница с товаром

На этой странице предусмотрена кнопка для добавления товара в корзину чтобы он отложился до дальнейшей покупкит, а также кнопка для добавления товара в список желаемого. Также в шапке сайте предусмотрен переход в корзину.

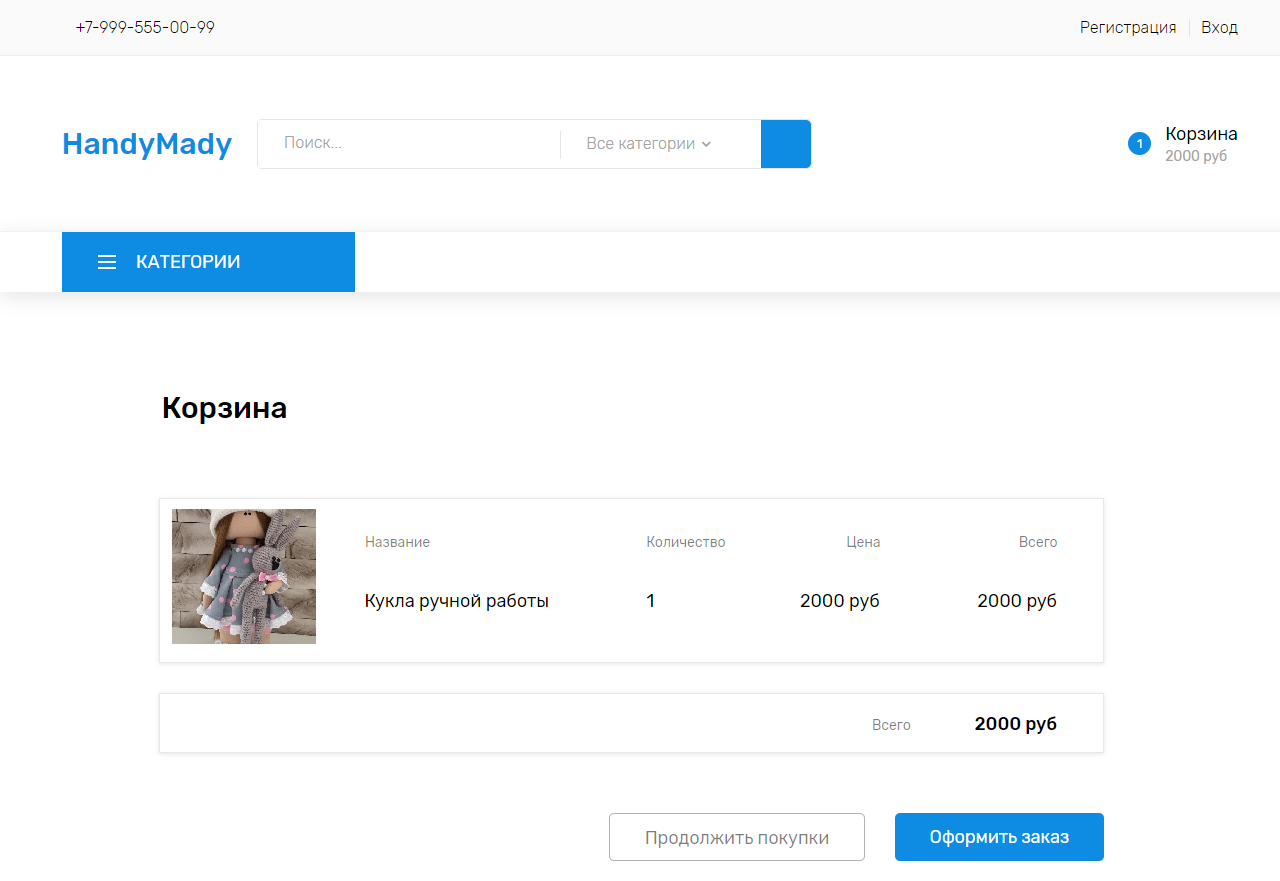


Рисунок 2.15 – Интерфейс корзины

В корзине представлена информация об отложенных товарах для покупки. Пользователь может ознакомиться с окончательной ценой всего заказа и перейти к оформлению заказа при помощи кнопки. Если покупатель хочет вернуться в магазин чтобы продолжить добавлять товары в корзину, он может нажать на кнопку «Продолжить покупки».

# Архитектура системы

Проектирование серверной части было осуществлено посредством свободно-распространяемого кросс-платформенного фреймворка для создания веб-приложений с открытым исходным кодом на языке C# – ASP.NET Core. Он реализует распространённый архитектурный паттерн в веб-программировании Model-View-Controller.

Архитектурой MVC является схемой разделения данных, интерфейса и логики на три отдельных компонента таким образом, что модификация каждого компонента не зависит от другого. Модель данного паттерна изображена на рисунке 2.17.

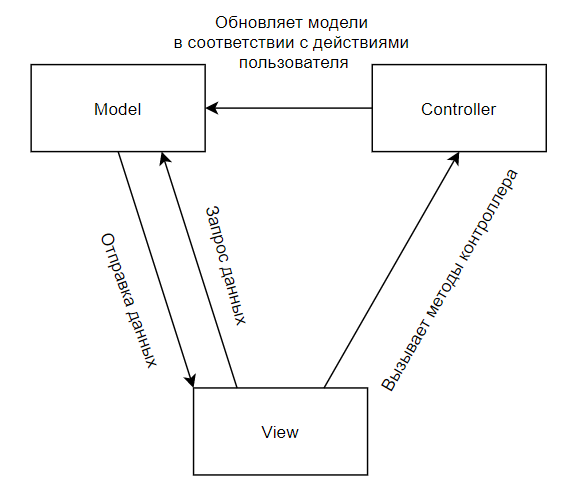


Рисунок 2.17 – Паттерн Model-View-Controller

Концепция MVC позволяет разделить модель, представление и контроллер на три отдельных компонента:

* Model предоставляет доступ к данным и обрабатывается слоем работы с базой данных. Она осуществляет запросы в БД и делает проверку на корректность.
* View отвечает за визуализацию информации.
* Controller осуществляет контроль и направление данных от пользователя к системе и наоборот.

Связь между View и Controller будет осуществлена при помощи архитектурного стиля REST. REST API является программным архитектурным стилем, определяющий набор различных ограничений, которые будут использоваться для создания веб-служб. Главная особенность REST заключается в том, что сервер не запоминает состояние пользователя между запросами. Все запросы передают информацию, которая идентифицирует пользователя и все параметры, необходимые для выполнения операции (рисунок 2.18). [13]

Всё взаимодействие с сервером сводится к 4 операциям:

1. Получение данных с сервера (GET запрос).
2. Добавление новых данных на сервер (POST запрос).
3. Модификация существующих данных на сервере (PUT запрос).
4. Удаление данных на сервере (DELETE запрос).



Рисунок 2.18 – архитектурный стиль REST API

# Проектно-технологическая часть

# Порядок развертывания системы

Развертывание разрабатываемой системы предполагает установку требуемого программного обеспечения и ее последующую настройку.

Для работы системы нам необходимо установить система управления базами данных PostgreSQL версии 12 или выше и фреймворк который необходим для работы приложения .NET Framework. Также для настройки приложения потребуется интегрированная среда разработки Microsoft Visual Studio.

Далее в проекте необходимо изменить данные в настройках для инициализации подключения к базе данных в соответствии с теми, что были введены при установке PostgreSQL. После этого необходимо осуществить миграцию данных для создания базы данных, таблиц и сущностей. Для этого не обходимо перейти в Package Manager Console и ввести команду для добавления миграции в проект Add-migration “MigrationName” (рисунок 3.1). При этом в проекте автоматически появится новый файл миграции, в котором содержатся все изменения данных (рисунок 3.2). Далее, для применения изменений нужно ввести команду для обновления базы данных при помощи Update-Database (рисунок 3.3). Результаты миграции показаны на рисунке 3.4.

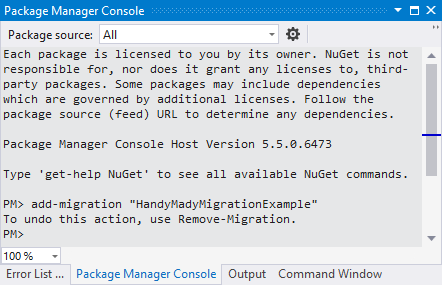


Рисунок 3.1 – Добавление миграции в Package Manager Console



Рисунок 3.2 – Файл миграции в проекте

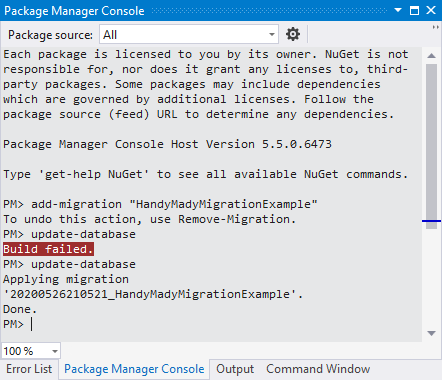


Рисунок 3.3 – Применение миграции в Package Manager Console

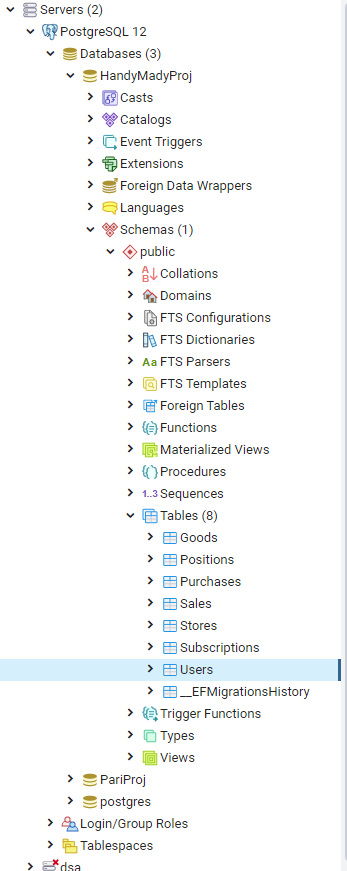


Рисунок 3.4 – Результат миграции

После проделанной миграции дальнейшая настройки СУБД не требуется. При наличии можно применить бекап для наполнения базы данных если она соответствует модели данных в проекте.

# Технологические решения, поддерживающие эксплуатационный цикл

Для поддержки эксплуатационного цикла система разрабатывается с использованием инкрементальной модели разработки (рисунок 3.5). Модель разделяет жизненный цикл на последовательность разных итераций, включая в себя все процессы разработки. Предполагается, что под конец каждой итерации будет выходить рабочая версия системы. [7]

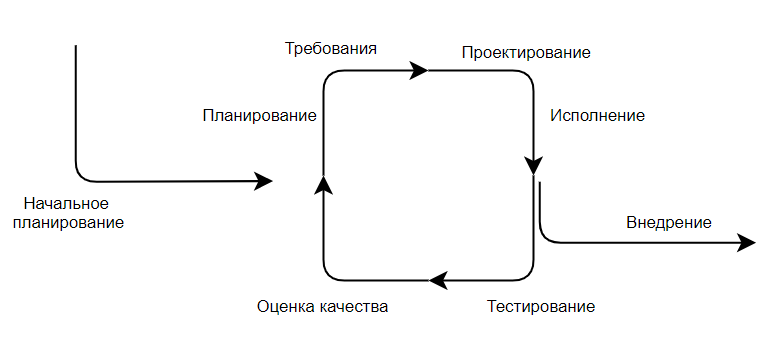


Рисунок 3.5 – Инкрементная модель

Достоинства данной модели:

* Уменьшение серьёзных рисков на ранних стадиях разработки, что может привести к минимизации затрат;
* Возможно увеличение усилий во время разработки определенного функционала проекта;
* Непрерывное тестирование, которое позволяет оценить успешность проекта;
* Возможность обнаружения конфликтов на ранней стадии разработки проекта;
* Равномерная нагрузка разработчиков проекта;

# Краткое описание интерфейса сервера

Для удобства описания интерфейса сервера в данном руководстве будет использоваться специальный инструмент Postman для отправки запросов.

Для регистрации пользователя необходимо отправить POST запрос с JSON со всеми данными для регистрации по адресу http://localhost:5100/api/authorize/registration (рисунок 3.6).

JSON содержит следующие атрибуты для заполнения:

* Логин
* Пароль
* Имя учетной записи
* Адрес для доставки
* Электронная почта
* Фамилия, имя и отчество
* Телефон

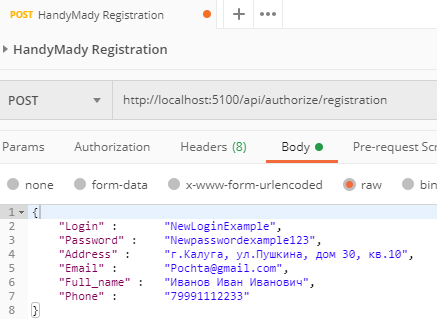


Рисунок 3.6 – Отправка POST запроса для регистрации пользователя

Если все данные введены правильно, то сервер вернет статус 200, а также JSON данные с токеном, который является уникальным для каждого пользователя идентификатором (рисунок 3.7). После регистрации система добавляет пользователя в базу данных и добавит ему токен означающий что пользователь зашел в систему.

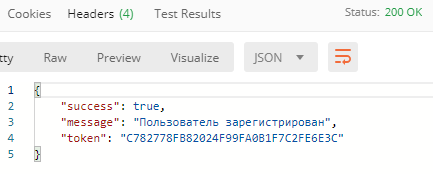


Рисунок 3.7 – Ответ сервера на успешный запрос для регистрации пользователя

Для того чтобы выйти из системы предусмотрен GET запрос http://localhost:5100/api/authorize/userexitbyid в котором параметром передается ID пользователя. При статусе 200 сервер возвращает сообщение об успешном выходе и удаляет токен у пользователя из базы данных (рисунок 3.8).

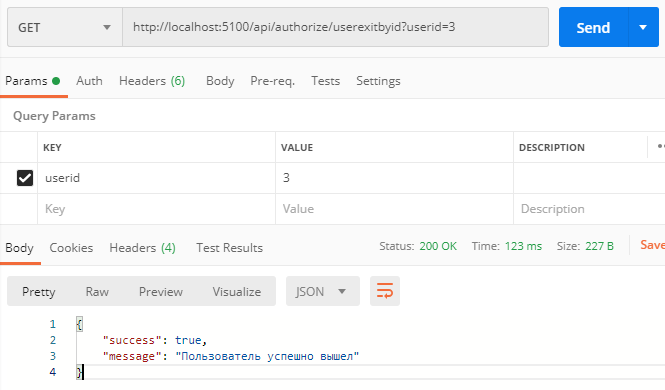


Рисунок 3.8 – Отправка GET запроса для выхода из системы

Для входа в систему необходимо отправить POST запрос http://localhost:5100/api/authorize/authorize, который содержит логин и пароль (рисунок 3.9). Если данные введены верно сервер возвратит сообщение что пользователь авторизирован и привяжет к нему токен (рисунок 3.10).

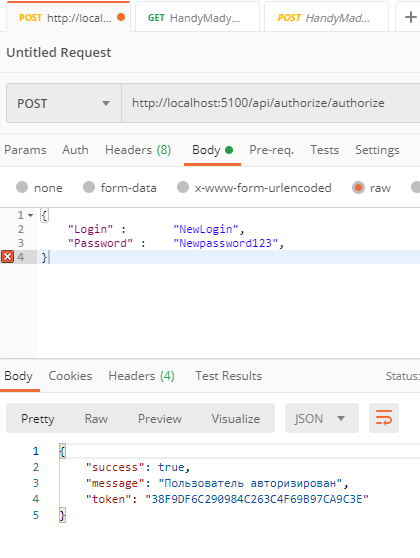


Рисунок 3.9 – Отправка POST запроса для входа в систему

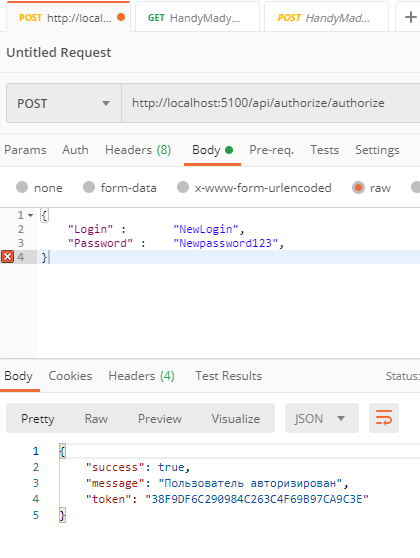


Рисунок 3.10 – Ответ сервера на успешный запрос для входа в систему

Чтобы создать магазин нужно отправить POST запрос http://localhost:5100/api/Store/createstore, который содержит наименование, описание и тип магазина, а также токен по которому магазин будет присвоен пользователю, который его создал (рисунок 3.11).

Данный запрос имеет следующие атрибуты для заполнения:

* Токен пользователя
* Наименование магазина
* Описание магазина
* Тип магазина

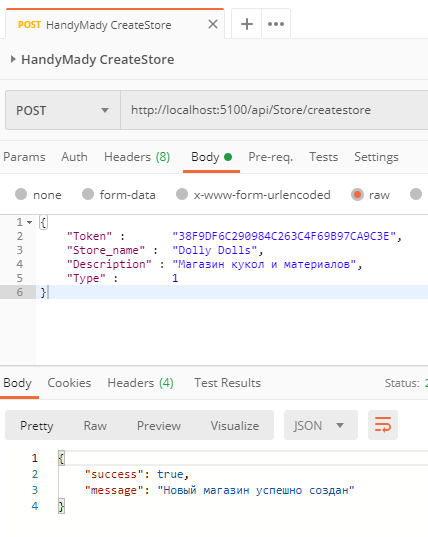


Рисунок 3.11 – Отправка POST запроса для создания магазина

При правильно введенных данных сервер возвратит сообщение о том, что магазин успешно создан. (рисунок 3.12)

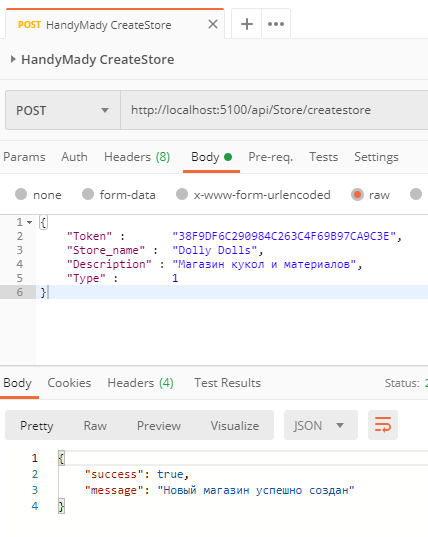


Рисунок 3.12 – Ответ сервера на успешный запрос для создания магазина

POST запрос http://localhost:5100/api/Store/addgoodintostoreid отвечает за создание добавление товара в магазин (рисунок 3.13). При правильно введенных данных запрос вернет сообщение, что товар успешно добавлен в магазин.

В запрос передаются следующие атрибуты для заполнения:

* Id магазина
* Наименование товара
* Описание товара
* Тип товара
* Количество
* Цена

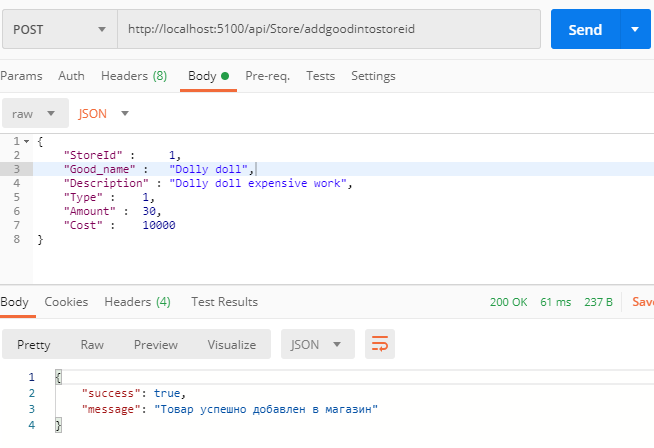


Рисунок 3.13 – Отправка POST запроса для добавления товара в магазин

Для получения всех товаров из магазина следует использовать GET запрос http://localhost:5100/api/Store/getgoodsfromstorebyid у которого параметром нужно передавать id магазина. В ответ мы получим JSON со списком товаров (рисунок 3.14).

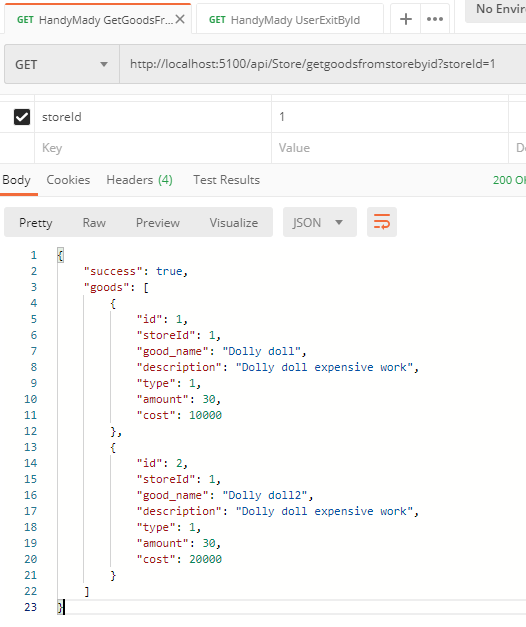


Рисунок 3.14 – Отправка GET запроса для получения списка товаров

GET запрос http://localhost:5100/api/store/searchgoods позволяет произвести поиск товаров по названию. Для этого параметром нужно передавать строку с названием предполагаемого товара. В ответ мы получим JSON со списком товаров (рисунок 3.15).

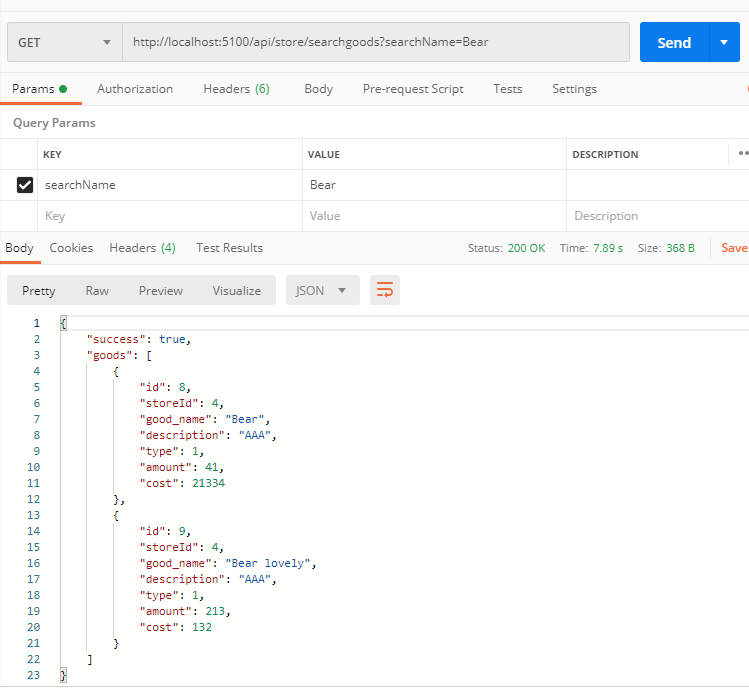


Рисунок 3.15 – Отправка GET запроса для поиска товаров по названию

# ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы было сформулировано техническое задание на разработку программной системы, описана и проанализирована предметная область разрабатываемого продукта, представлена концептуальная схема данных предметной области, сформулированы основные сценарии использования и обоснован выбор инструментов и платформ для разработки.

В проектно-конструкторской части была разработана структура программной системы, логическая схема данных и разработан пользовательский интерфейс и выбрана архитектура приложения.

В проектно-технологической части содержится порядок развертывания системы, технологические решения, поддерживающие эксплуатационный цикл программы и кратко описан интерфейс сервера.

В ходе реализации веб-приложения были получены практические навыки работы с ASP.NET Core.

Таким образом, цель ВКР достигнута, а все поставленные задачи выполнены.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Белугина, С. В. Разработка программных модулей программного обеспечения для компьютерных систем. Прикладное программирование : учебное пособие / С. В. Белугина. — Санкт-Петербург : Лань, 2020. — 312 с. — ISBN 978-5-8114-4496-0. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/133920 (дата обращения: 23.04.2020). — Режим доступа: для авториз. пользователей.
2. Гутман, Г. Н. Объектно-реляционная СУБД PostgreSQL : учебное пособие / Г. Н. Гутман. — Самара : Самарский государственный технический университет, ЭБС АСВ, 2016. — 125 c. — ISBN 2227-8397. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/90660.html (дата обращения: 25.04.2020). — Режим доступа: для авторизир. Пользователей
3. Ипатова, Э. Р. Методологии и технологии системного проектирования информационных систем : учебное пособие / Э. Р. Ипатова. — 2-е изд., стер. — Москва : ФЛИНТА, 2016. — 256 с. — ISBN 978-5-89349-978-0. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/84356 (дата обращения: 01.05.2020). — Режим доступа: для авториз. пользователей.
4. Миндалёв, И. В. Моделирование бизнес-процессов с помощью IDEF0, DFD, BPMN за 7 дней : учебное пособие / И. В. Миндалёв. — Красноярск : КрасГАУ, 2016. — 123 с. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/103833 (дата обращения: 23.04.2020). — Режим доступа: для авториз. пользователей.
5. Основы работы с HTML / . — 2-е изд. — Москва : Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. — 208 c. — ISBN 2227-8397. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/73698.html (дата обращения: 23.04.2020). — Режим доступа: для авторизир. Пользователей
6. Ревунков, Г. И. Проектирование баз данных : учебное пособие / Г. И. Ревунков, Н. А. Ковалева, Е. Ю. Силантьева. — Москва : МГТУ им. Н.Э. Баумана, 2018. — 48 с. — ISBN 978-5-7038-4718-3. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/103499 (дата обращения: 23.04.2020). — Режим доступа: для авториз. пользователей.
7. Рочев, К. В. Информационные технологии. Анализ и проектирование информационных систем : учебное пособие / К. В. Рочев. — 2-е изд., испр. — Санкт-Петербург : Лань, 2019. — 128 с. — ISBN 978-5-8114-3801-3. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/122181 (дата обращения: 09.05.2020). — Режим доступа: для авториз. пользователей.
8. Столбовский, Д. Н. Разработка Web-приложений ASP.NET с использованием Visual Studio .NET : учебное пособие / Д. Н. Столбовский. — 3-е изд. — Москва, Саратов : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2020. — 375 c. — ISBN 978-5-4497-0370-5. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/89469.html (дата обращения: 23.04.2020). — Режим доступа: для авторизир. Пользователей
9. Сычев, А. В. Перспективные технологии и языки веб-разработки : практикум / А. В. Сычев. — 3-е изд. — Москва : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Эр Медиа, 2019. — 493 c. — ISBN 978-5-4486-0507-9. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/79730.html (дата обращения: 02.05.2020). — Режим доступа: для авторизир. пользователей
10. Тюкачев, Н. А. C#. Алгоритмы и структуры данных : учебное пособие / Н. А. Тюкачев, В. Г. Хлебостроев. — 3-е изд., стер. — Санкт-Петербург : Лань, 2018. — 232 с. — ISBN 978-5-8114-2566-2. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/104961 (дата обращения: 23.04.2020). — Режим доступа: для авториз. пользователей.
11. Тюкачев, Н. А. C#. Основы программирования : учебное пособие / Н. А. Тюкачев, В. Г. Хлебостроев. — 3-е изд., стер. — Санкт-Петербург : Лань, 2018. — 272 с. — ISBN 978-5-8114-2567-9. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/104962 (дата обращения: 23.04.2020). — Режим доступа: для авториз. пользователей.
12. Шёниг, Г. -. PostgreSQL 11. Мастерство разработки / Г. -. Шёниг ; перевод с английского А. А. Слинкина. — Москва : ДМК Пресс, 2020. — 352 с. — ISBN 978-5-97060-671-1. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/131714 (дата обращения: 23.04.2020). — Режим доступа: для авториз. пользователей.
13. Эделман, Д. Автоматизация программируемых сетей : руководство / Д. Эделман, С. С. Лоу, М. Осуолт ; перевод с английского А. В. Снастина. — Москва : ДМК Пресс, 2019. — 616 с. — ISBN 978-5-97060-699-5. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/123708 (дата обращения: 23.04.2020). — Режим доступа: для авториз. пользователей.

# ПРИЛОЖЕНИЕ А

**Графический материал**

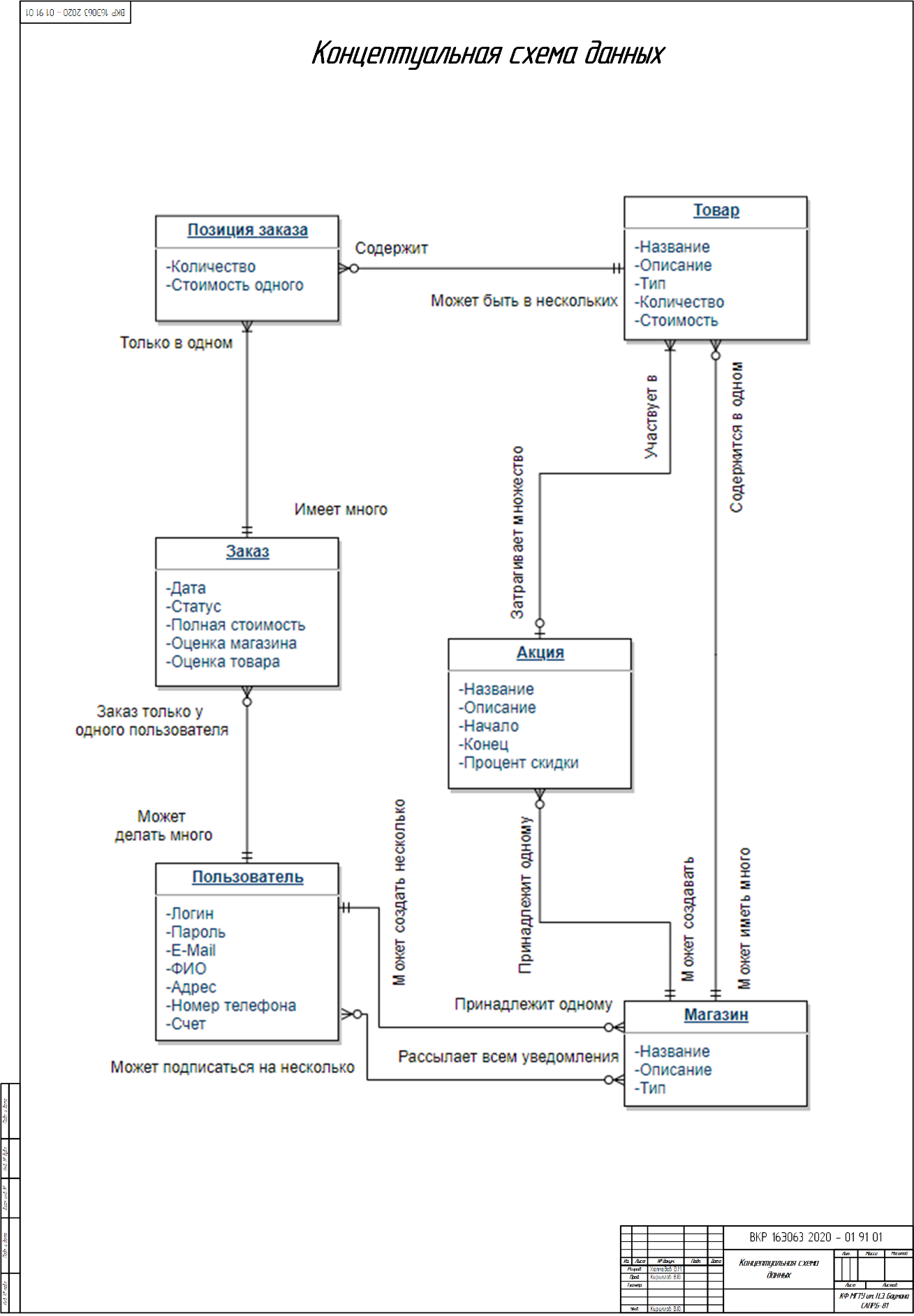
****

Рисунок А.1 – Концептуальная схема данных

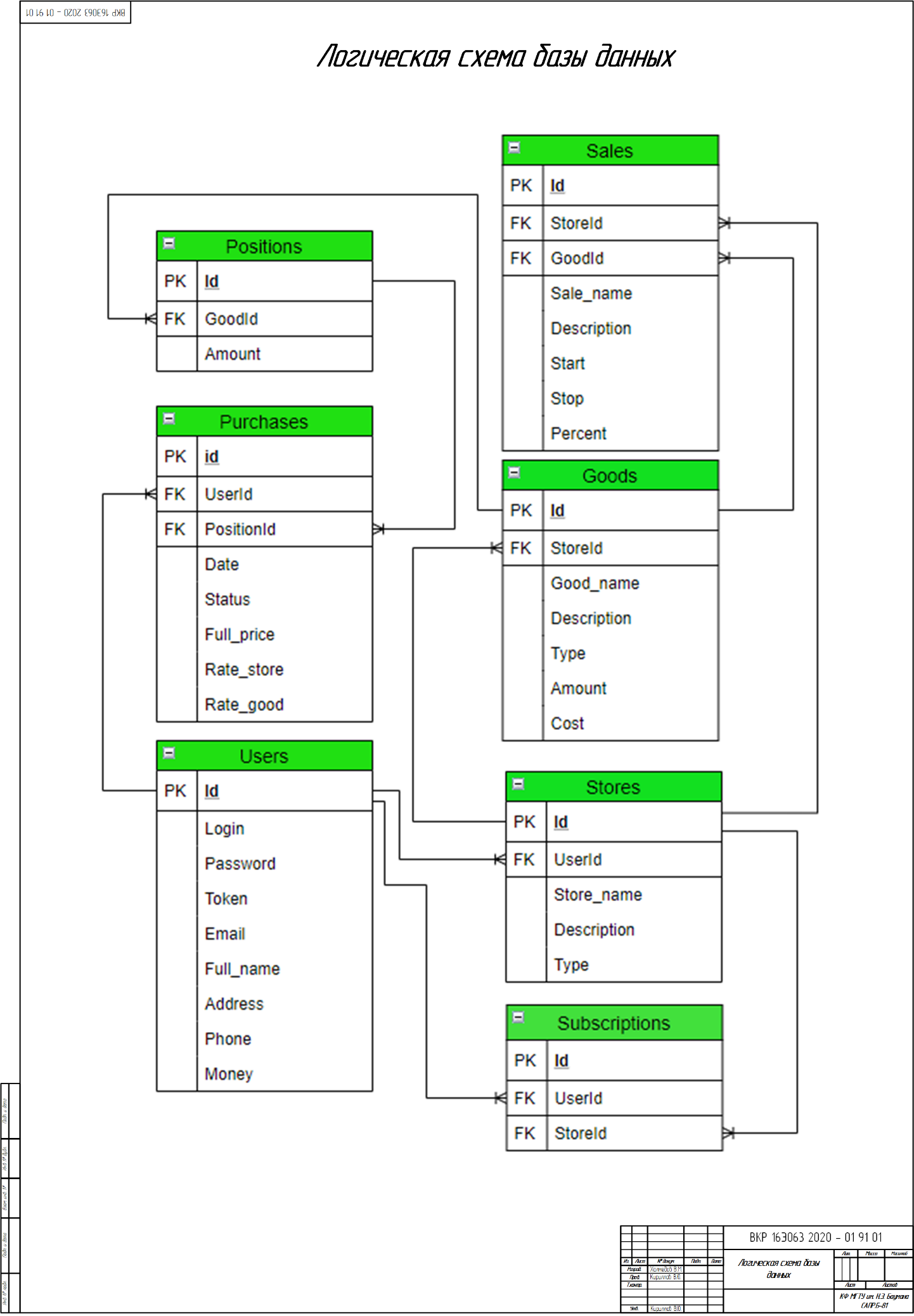


Рисунок А.2 – Логическая схема базы данных

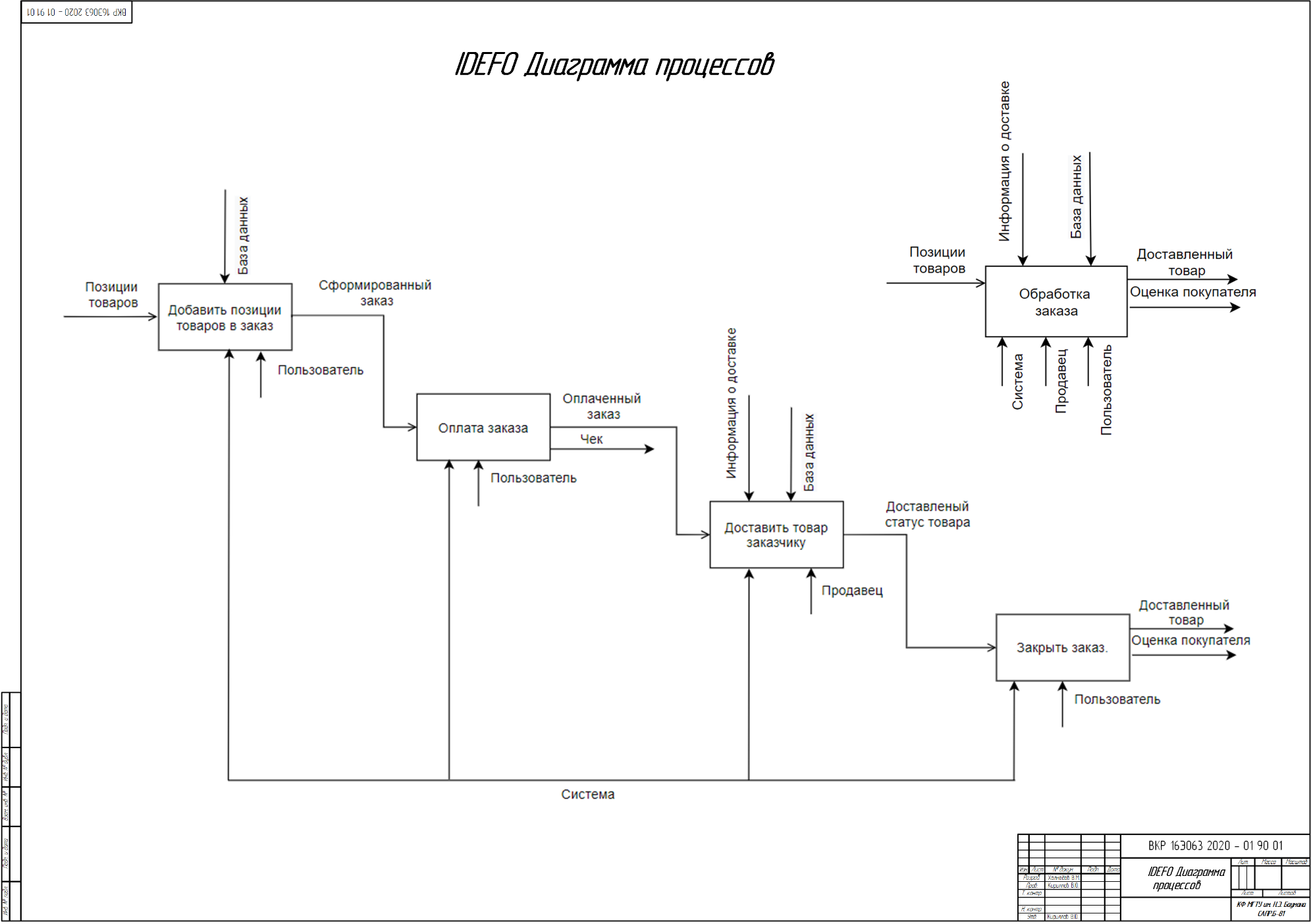


Рисунок А.3 – IDEF0 диаграмма процессов

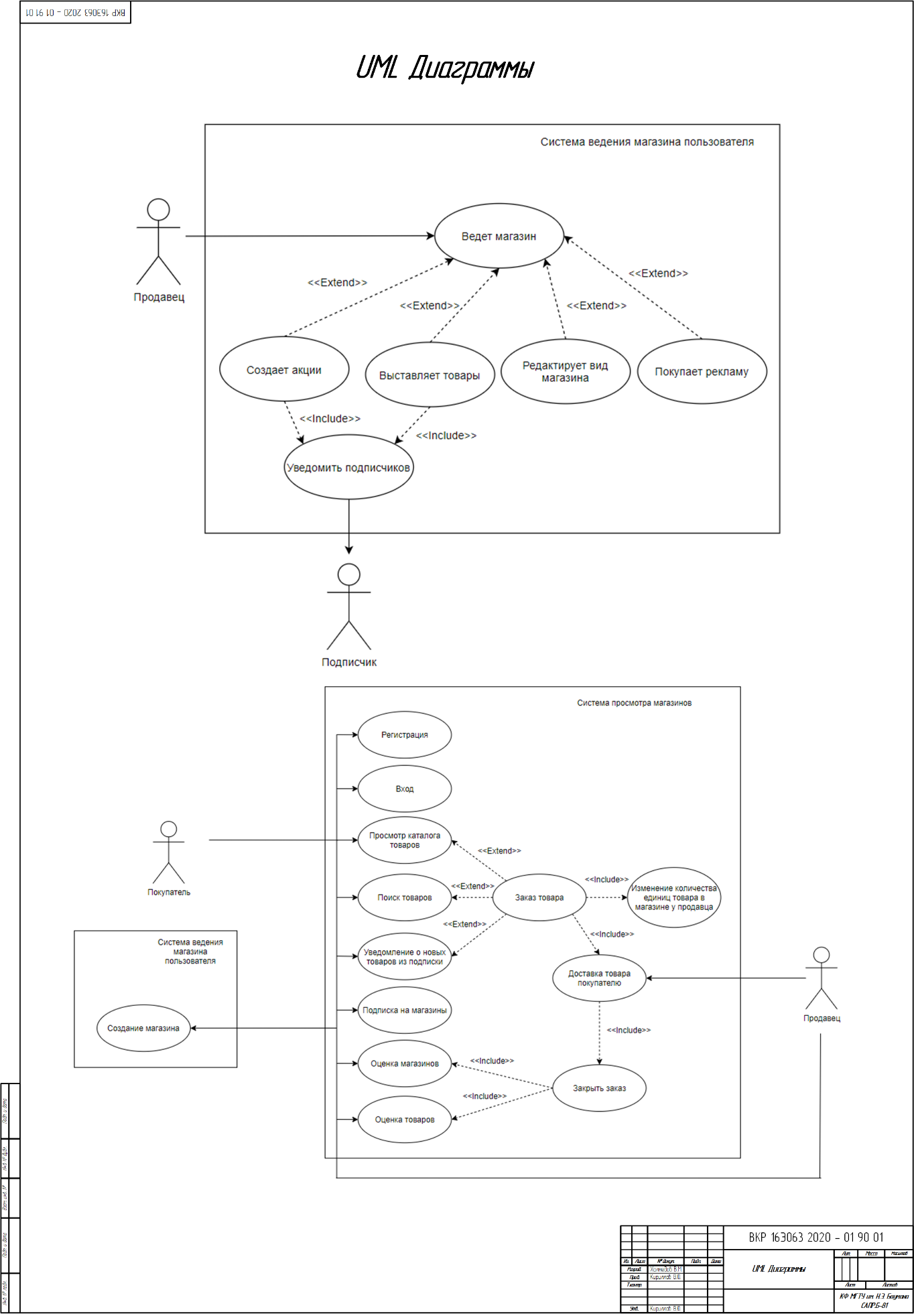


Рисунок А.4 – UML диаграммы

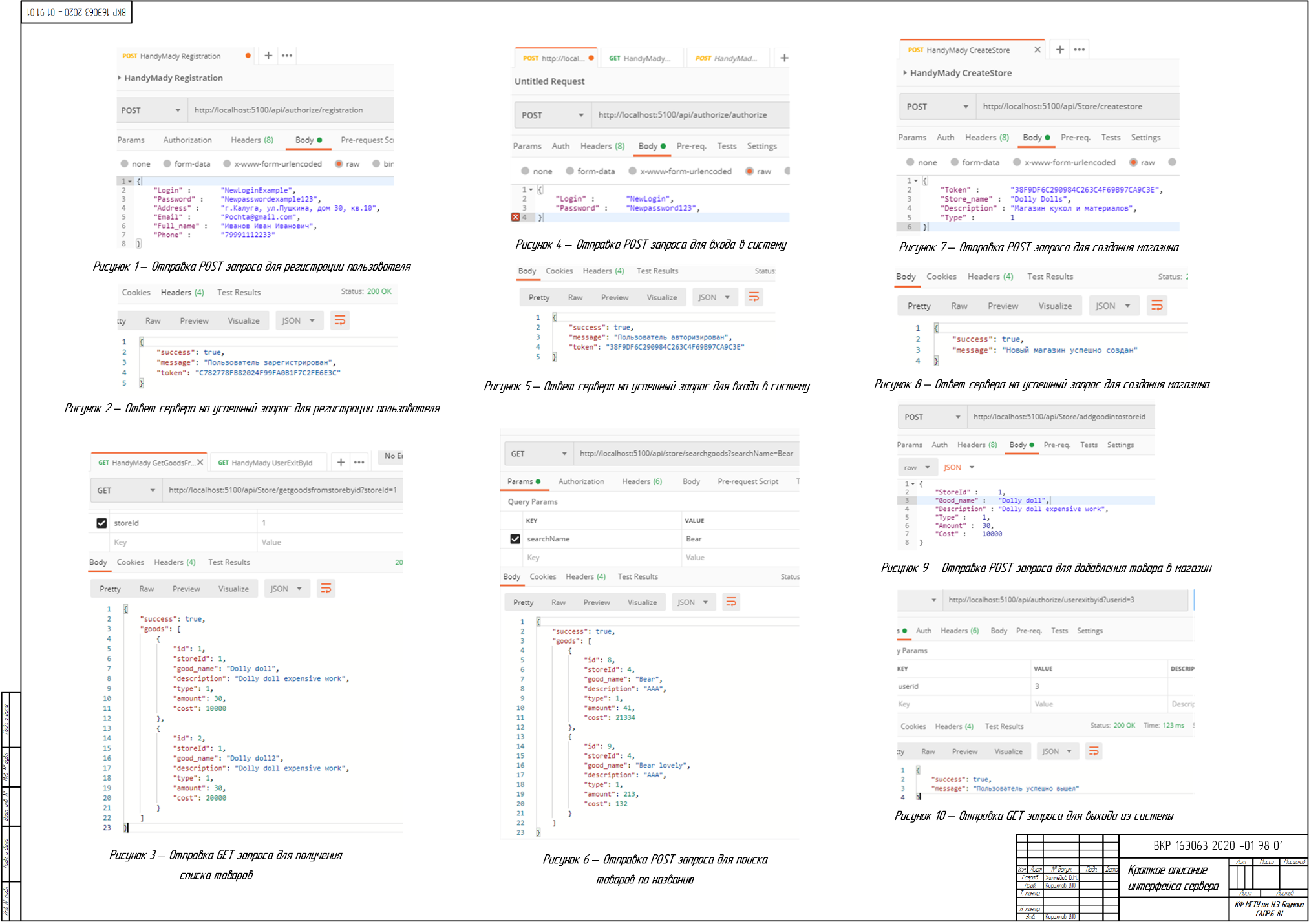


Рисунок А.5 – Краткое описание интерфейса сервера

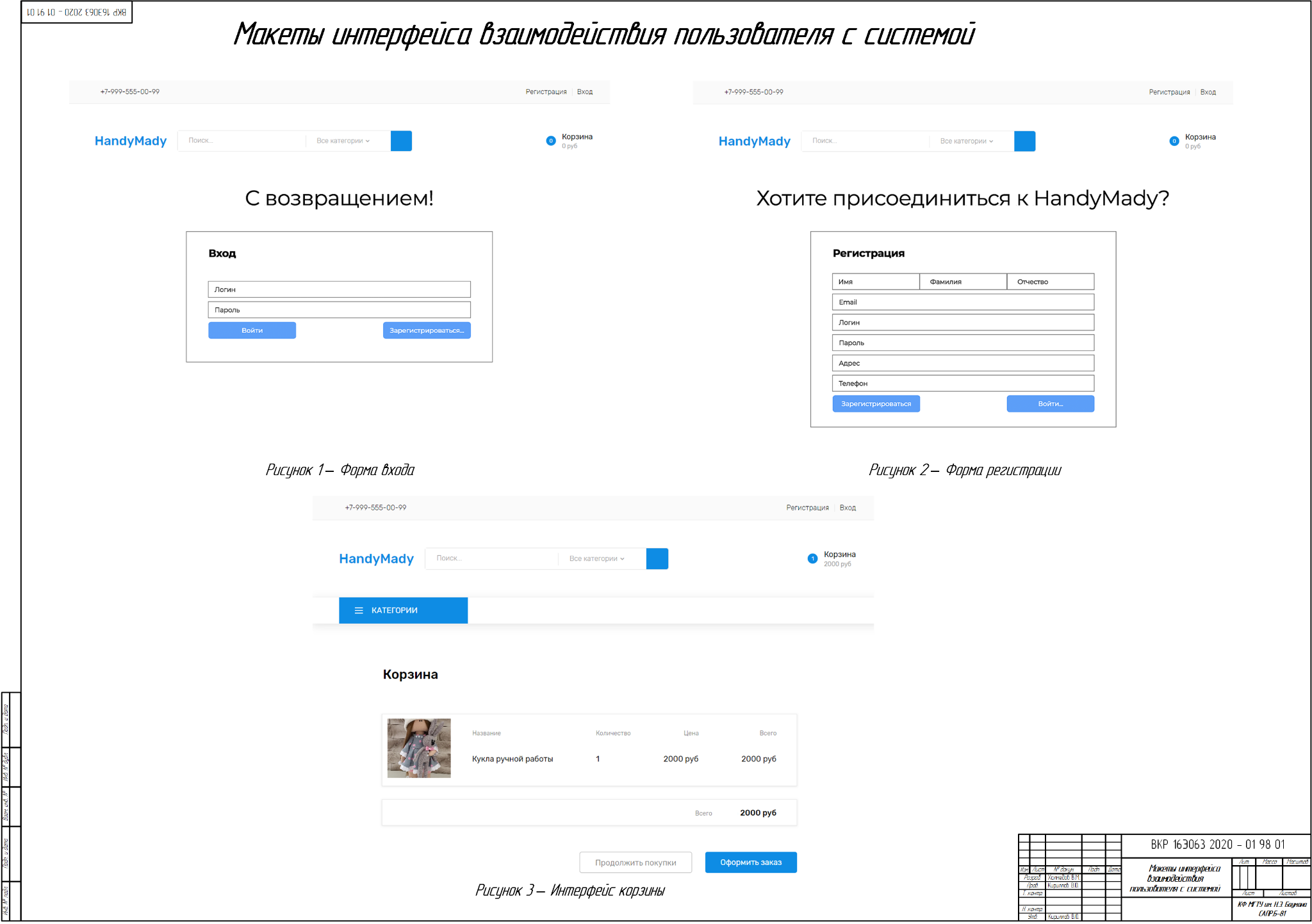


Рисунок А.6 – Макеты интерфейса взаимодействия пользователя с системой

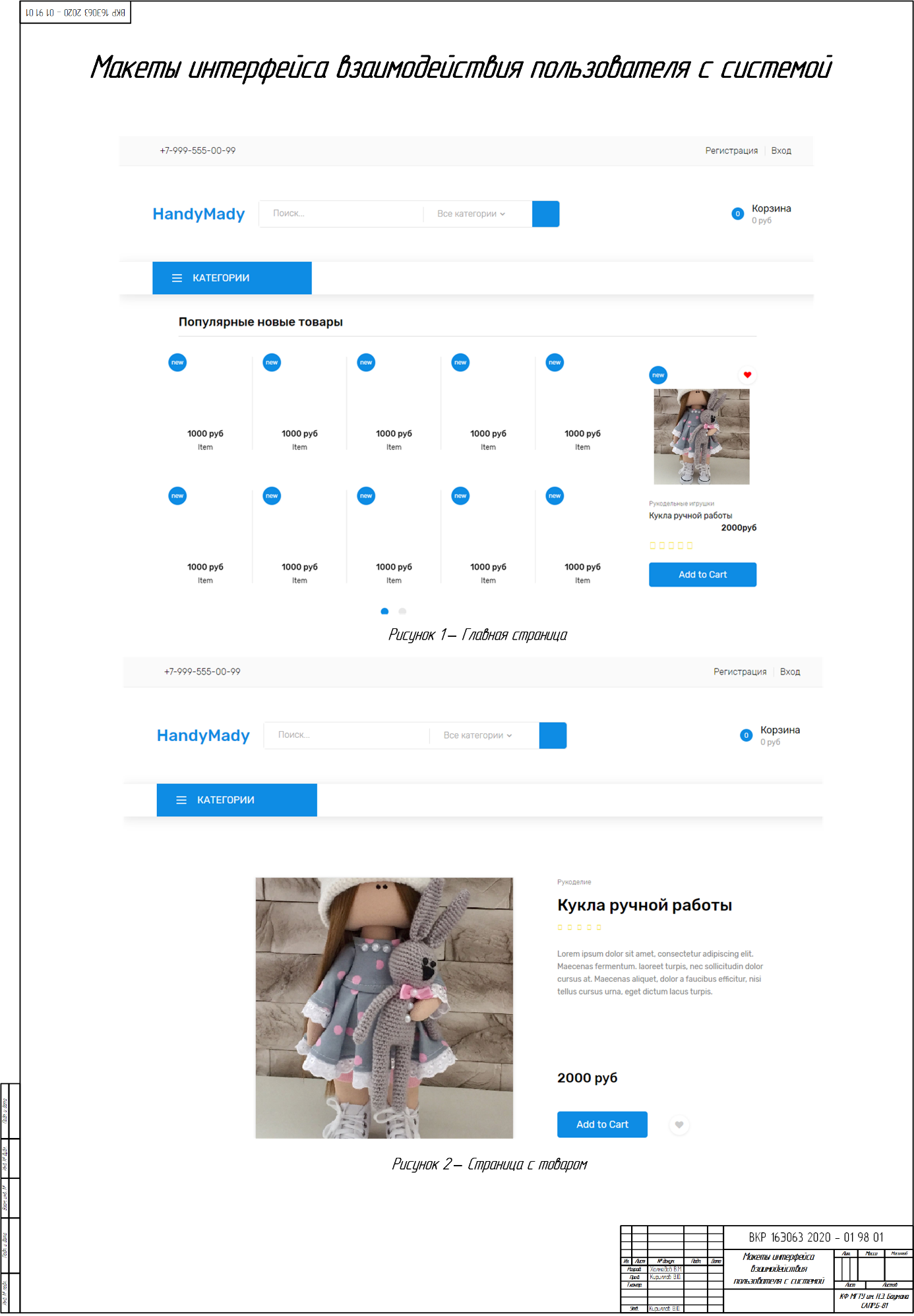


Рисунок А.7 – Макеты интерфейса взаимодействия пользователя с системой