**Р Е Ф Е Р А Т**

Пояснительная записка 122с., 46 рис., 3 табл., 14 источников

ИНТЕРНЕТ-МАГАЗИН, ЭЛЕКТРОННАЯ КОММЕРЦИЯ, ОНЛАЙН-КАТАЛОГ, ЗАКАЗ

Объектом исследования данного дипломного проекта является автоматизация процессов покупки товаров определенной организации.

Цель работы – проектирование и разработка программного средства, решающего задачи автоматизации процессов покупки товара, распространения информации о продукции путем просмотра пользователем онлайн-каталога, реализацией возможности совершения заказа, а также сбора и анализа результатов посещения интернет-магазина.

Разработка и использование описанного программного средства позволяет упростить процесс ведения коммерческой деятельности, а также сократить временные и экономические затраты, связанные с этим процессом.

В процессе работы над проектом были изучены подходы к формализации электронной коммерции, основные тенденции и направления развития в этой области, были изучены подходы к реализации программных средств по теме дипломного проектирования, разработана и реализована программная архитектура с возможностью дальнейшего масштабирования.

В разделе технико-экономического обоснования были произведены расчеты затрат, связанных с построением проекта, а также рентабельности разработки проекта. Проведенные расчеты показали экономическую целесообразность проекта.

**СОДЕРЖАНИЕ**

[Введение 7](#_Toc60571449)

[1 Аналитический обзор программных продуктов, методов и подходов по теме дипломного проектирования 8](#_Toc60571450)

[1.1 Основные понятия и определения в области электронной коммерции 8](#_Toc60571451)

[1.2 Обзор тенденций в области Интернет-торговли 10](#_Toc60571452)

[1.3 Анализ существующих программных решений по теме дипломного проектирования 12](#_Toc60571453)

[1.4 Постановка целей и задач дипломного проектирования 24](#_Toc60571454)

[2 Моделирование предметной области, разработка функциональных требований и составление их спецификации 25](#_Toc60571455)

[2.1 Общие сведения и требования к работе программного продукта 25](#_Toc60571456)

[2.2 Описание функциональности программного продукта 25](#_Toc60571457)

[2.3 Разработка информационной модели 30](#_Toc60571458)

[2.4 Разработка модели взаимодействия пользователя с интерфейсом 32](#_Toc60571459)

[2.5 Разработка спецификации функциональных требований 33](#_Toc60571460)

[2.6 Разработка технических требований к программному продукту 34](#_Toc60571461)

[3 Проектирование архитектуры программного продукта 35](#_Toc60571462)

[3.1 Разработка архитектуры программного продукта 35](#_Toc60571463)

[3.2 Проектирование архитектуры базы данных 39](#_Toc60571464)

[3.3 Проектирование алгоритмов ПС 39](#_Toc60571465)

[4 Разработка программного продукта 43](#_Toc60571466)

[4.1 Выбор и обоснование языков программирования 43](#_Toc60571467)

[4.2 Выбор среды разработки 45](#_Toc60571468)

[4.3 Диаграмма классов программного продукта 46](#_Toc60571469)

[4.4 Описание компонентов клиентской части программного продукта 49](#_Toc60571470)

[5 Тестирование программного продукта 50](#_Toc60571471)

[6 Руководство пользователя 52](#_Toc60571472)

[6.1 Развертывание сервера приложения 52](#_Toc60571473)

[6.2 Использование программного продукта 52](#_Toc60571474)

[7 Технико-экономическое обоснование разработки интернет-магазина для ОАО «Свiтанак» 60](#_Toc60571475)

[7.1 Описание функций, назначения и потенциальных пользователей программного продукта 60](#_Toc60571476)

[7.2 Расчет затрат на разработку программного средства 60](#_Toc60571477)

[7.3 Оценка экономического эффекта от использования ПС 64](#_Toc60571478)

[Заключение 66](#_Toc60571479)

[Список использованной литературы 67](#_Toc60571480)

[Приложение А 68](#_Toc60571481)

**ОПРЕДЕЛЕНИЯ И СОКРАЩЕНИЯ**

В настоящей пояснительной записке применяются следующие определения и сокращения.

*Авторизация* – предоставление определённому лицу или группе лиц прав на выполнение определённых действий; а также процесс проверки (подтверждения) данных прав при попытке выполнения этих действий.

*Аутентификация* – проверка подлинности предъявленного пользователем идентификатора.

*Инициализация* – приведение областей памяти в состояние, исходное для последующей обработки или размещения данных.

*Интерпретатор* – программа или техническое средство, выполняющие интерпретацию.

*Программа* – данные, предназначенные для управления конкретными компонентами системы обработки информации в целях реализации определенного алгоритма.

*Программное* *обеспечение* – совокупность программ системы обработки информации и программных документов, необходимых для эксплуатации этих программ.

*Программирование* – научная и практическая деятельность по созданию программ.

*Программный* *модуль* – программа или функционально завершенный фрагмент программы, предназначенный для хранения, трансляции, объединения с другими программными модулями и загрузки в оперативную память.

*Спецификация* *программы* – формализованное представление требований, предъявляемых к программе, которые должны быть удовлетворены при ее разработке, а также описание задачи, условия и эффекта действия без указания способа его достижения.

*Фреймворк* – программное обеспечение, облегчающее разработку и объединение различных компонентов большого программного проекта.

ООП – объектно-ориентированное программирование.

СУБД – система управления базами данных.

БД – база данных.

ПС – программное средство.

ПП – программный продукт.

ОС – операционная система.

API – application programming interface (Интерфейс программирования приложений).

JSON – JavaScript object notation.

URL – uniform resource locator (единообразный локатор ресурса).

# Введение

Интернет-магазин – это прикладная система, построенная с использованием технологии электронной торговли. Подобно привычному нам магазину, электронный магазин реализует следующие функции: представление товаров и их характеристик покупателю, обработку заказов, продажу и предоставление информации для дальнейшей доставки товаров покупателю.

Интернет-магазин объединяет элементы прямого маркетинга с образом посещения традиционного магазина. Отличительной чертой интернет-магазинов по сравнению с обычной формой торговли является то, что интерактивный магазин может предложить значительно большее количество товаров и услуг, и обеспечить потребителей значительно большим объемом информации, необходимым для принятия решения о покупке.

Основные проблемы реализации интернет-магазина лежат на стыке технологий Интернета и традиционной коммерческой деятельности. В обычной торговле покупатель привык к тому, что есть возможность оценить товар визуально, определить его качество и характеристики. В электронной торговле он такой возможности лишен. Иногда визуальной информации достаточно, но здесь имеет место быть эмоциональным и психологическим факторам.

По последним данным аудитория в интернете стремительно растет, а продажи через интернет в крупных городах достигают до 25%, при этом специалисты подчеркивают тенденцию к росту продаж именно через интернет. Ежегодно количество интернет-магазинов увеличивается, так как это действительно прибыльно и удобно для покупателя, не говоря о экономии бюджета и времени. Интернет-магазин работает круглые сутки и может продавать определенные товары в автоматическом режиме без участия продавца. К преимуществам так же можно отнести то, что не надо закупать товар заранее, а это существенная экономия, на складских помещениях. По сравнению с обычным магазином, территория продаж которого ограничивается территорией города или района, охвата интернет-магазина увеличивается на всю территорию Республики Беларусь и русскоязычную аудиторию в других странах, ведь товар можно доставлять не только курьерской службой, но и почтой.

Основной целью данной дипломной работы является разработка интернет-магазина по продаже трикотажных изделий из хлопка и смесей хлопка, выпускаемой ОАО «Свiтанак». Данная система будет работать в рамках определенного предприятия и выполнять функции дополнительной площадки для ведения коммерческой деятельности.

# 1 Аналитический обзор программных продуктов, методов и подходов по теме дипломного проектирования

## Основные понятия и определения в области электронной коммерции

Электронная коммерция – это форма поставки продукции, при которой выбор и заказ товаров осуществляется при помощи персонального компьютера или подобного ему устройства, а расчеты между покупателем и поставщиком осуществляются с использованием электронных документов и/или специальных средств платежа, при этом в качестве покупателей товаров или услуг могут выступать как частные лица, так и организации. Электронная коммерция представляет собой огромный комплект различных бизнес-операций.

Термин «электронная коммерция» объединяет в себе много различных технологий:

* электронный обмен информацией (Electroniс Data Interchange, EDI),
* электронное движение капитала (Electronic Funds Transfer, EFT),
* электронную торговлю (англ. E-trade),
* электронные деньги (e-cash),
* электронный маркетинг (e-marketing),
* электронный банкинг (e-banking),
* электронные страховые услуги (e-insurance).

Электронная торговля – осуществление торгово-закупочной деятельности через Интернет. Электронная коммерция – это общая концепция, содержащая в себе любые формы деловых операций, исполняемых электронным способом, а также использующая различные телекоммуникационные технологии. Деловые операции могут осуществляться напрямую между фирмами, фирмой и заказчиком, а также между фирмой и государственным учреждением.

Электронная коммерция реализуется в рамках интернет-экономики, которую нередко называют сетевой экономикой.

Сетевая экономика – среда, в которой каждая компания или индивид, находящиеся в любой точке экономической системы, могут контактировать свободно и с минимальными затратами с любой другой компанией либо индивидом по поводу совместной работы, для торговли либо для обмена идеями.

Современные информационные технологии, используемые в системе электронной коммерции, содержат в себе специальную инфраструктуру программного и аппаратного обеспечения, общие службы, специальные приложения, а также правовую структуру и соответствующие стандарты.

Основу функционирования системы электронной коммерции составляют электронные магазины.

Электронные магазины представляют собой реализованное предпринимателем представительство в сети Интернет на базе создания web-сервера. Основная цель создания такого предприятия состоит в обеспечении продажи товаров и оказании услуг другим пользователям сети Интернет.

Сферы применения системы электронной коммерции весьма разнообразны. Они охватывают широкий спектр коммерческих сделок (коммерческих сделок) и сделок, в частности:

* установить контакт между потенциальным покупателем и поставщиком;
* электронный обмен необходимой информацией;
* пред и послепродажная помощь покупателю, совершившему покупку в интернет магазине (предоставление подробной информации о продукте или услуге, предоставление инструкций по использованию продукта, оперативные ответы на вопросы клиентов
* осуществление прямого акта продажи товара или услуги;
* электронная оплата покупок (электронным переводом, кредитными картами, электронными деньгами, электронными чеками);
* предоставить покупателю товар, включая как управление отгрузкой, так и отслеживание физических товаров, а также прямую доставку товаров, которая может распространяться в электронном виде;
* создание виртуального предприятия, то есть группы независимых компаний, которые объединяют свои различные ресурсы, чтобы получить возможность предоставлять продукты и услуги, недоступные для независимых компаний;
* реализация независимых бизнес процессов, осуществляемых совместно производителем и его деловыми партнерами.

Электронная коммерция может осуществляться на двух уровнях: национальном и международном. Основанием для разницы в ведении бизнес операций на этих уровнях является не техническая и технологическая составляющие, а законодательная. На международном уровне (по сравнению с национальным) внедрение системы электронной коммерции значительно усложняется. Это связано с такими факторами, как использование разных систем налогообложения, таможенных пошлин, принятие индивидуальных и в то же время неравноправных соглашений между разными странами, существенные различия в правилах проведения банковских операций. Функционирование систем электронной коммерции на национальном уровне в основном связано с представлением компании в сети, рекламой, а также до и послепродажной поддержкой.

В системе электронной коммерции выделяют следующие категории организации коммерческой деятельности:

* коммерческие организации:
  + B2B (Business to Business) – «взаимоотношения между коммерческими организациями»;
  + B2C (Business to Consumer) – «взаимоотношения между коммерческой организацией и потребителями»;
  + B2E (Business to Employee) – «взаимоотношения между коммерческими организациями и сотрудниками (наёмными рабочими)»;
  + B2G (Business to Government) – «взаимоотношения между организацией и правительством»;
* потребители:
  + C2A (Consumer to Administration) – «взаимоотношения между потребителями и администраторами»;
  + C2B (Consumer to business) – «взаимоотношения между потребителями и коммерческими организациями»;
  + C2C (Consumer to Consumer) – «взаимоотношения между потребителями»;
* администрация:
  + A2A (Administration to Administration) – «взаимоотношения между администрациями»;
  + A2B (Administration to Business) – «взаимоотношения между администрацией и коммерческими организациями»
  + A2C (Administration to Consumer) – «взаимоотношения между администрацией и потребителями».

## Обзор тенденций в области Интернет-торговли

Объем продаж в области электронной торговли по данным статистической компании «Statista» за 2019 год достиг 3,5 триллиона долларов, что является почти 4% мирового ВВП. Однако недавний кризис оказал в некотором смысле положительное влияние на легкую промышленность. Объем продаж, осуществляемый посредством сети Интернет показал безумной быстрый темп роста и в 2020 году достиг 4,2 триллиона долларов. По некоторым прогнозам, придерживаясь такого темпа роста к 2023 году они составят порядка 6,5 трлн долларов [8]. Диаграмма роста объема продаж отображена на рисунке 1.1.

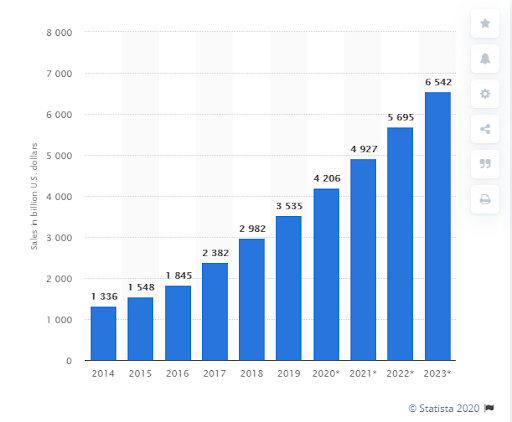


Рисунок 1.1 – Диаграмма роста объема продаж в области электронной коммерции от статистической компании «Statista»

Попадая на веб-сайт пользователи в течение некоторого времени формируют мнение о данной платформе, и многое тут зависит от дизайна. Как раз-таки за счет следования тенденциям можно овладеть доверием покупателя. Люди все чаще совершают покупки с помощью своих мобильных устройств, поэтому крайне важно иметь адаптивный дизайн.

Порядка 50% всех покупок в интернет-магазинах и чуть менее половины транзакций происходит с мобильных устройств. И доля подобных покупок и транзакций продолжает расти. Поэтому обязательными функциями для удобства пользователей являются лёгкая навигация, быстрая и простая процедура оформления заказа вкупе с несколькими вариантами оплаты заказа.

Преимущество адаптивного дизайна заключается в том, что вам не нужно создавать совершенно другую мобильную версию для вашего сайта. Адаптивный дизайн адаптируется к соответствующему устройству, и единственное, что вам нужно проверить, - это то, что изображения и размеры отображаются правильно.

Следующим пунктом можно рассмотреть такой тип дизайна, как минималистичный, либо же плоский веб-дизайн. Большим преимуществом такого дизайна является то, что он чистый и простой. Основное внимание в этих проектах уделяется удобству восприятия информации и простоте навигации. Минимализм – это одна из основных тенденций в электронной коммерции. Обесцвечивая интерфейс, делая его нейтральным, дизайнеры отдают главную роль фотографиям товара, высококлассным рендерам, исчерпывающему описанию и понятному уникальному торговому предложению.

Интерфейс должен выполнять исключительно служебную функцию, объединять пользователя с сайтом и позволять ему осуществлять ключевые действия. Здесь основным является контент, а интерфейс может повлиять на принятие решения, взять на себя акцент.

Типографика. В прошлом шрифты и типографика не были важной частью, рассматриваемой при проектировании интернет-магазинов. Но оказалось, что иногда шрифты и текст могут говорить громче, чем фотографии или видео. Это философия больших букв и массивного текста. Большие буквы впечатляют и наверняка привлекают внимание людей. В то же время сайт содержит основную информацию, которая нужна покупателям - информацию о товаре, корзину, меню.

Оптимизация логистики. В США набирает популярность модель, когда доставка идет до ближайшего магазина у дома. Интернет-магазинам необходимо искать методы минимизации своих расходов на доставку. Поэтому логистика, на мой взгляд, будет в формате доставки к магазинам у дома.

Сектор B2B. Электронная коммерция кардинально изменила способ покупки в традиционном виде и онлайн. Развивающиеся программные инновации, такие как ориентация на предоставление персонализированных покупок, улучшают комфорт клиентов с каждым днем. Наряду с этим их ожидания также растут. Следовательно, если нам необходимо эффективно привлекать своих клиентов, мы никогда не должны прекращать обновление своего магазина, чтобы удовлетворить их требования.

## Анализ существующих программных решений по теме дипломного проектирования

Наличие высокого спроса на покупку товаров в режиме онлайн привело к тому, что большинство крупных магазинов либо имеют площадку для ведения электронной торговли, либо же планируют ее создание. На данный момент число таких сервисов не ограничивается одним десятком, тематика которых варьируется от продажи гаджетов до оказания различного рода услуг, так что выберем максимально похожие –Conte Shop, Veore.by и Купалинка.

* + 1. Conte Shop

Одной из самых известных площадок белорусских производителей, которые позволяют приобретать в режиме онлайн свою продукцию является компания «Конте Шоп». Его популярность обусловлена целым рядом факторов, среди которых и большой ассортимент предлагаемой продукции, и удобный сервис для демонстрации, и предоставления возможности заказа товаров.

На рисунке 1.2 представлен одна из страниц, на которую попадает пользователь для просмотра конкретной категории каталога.

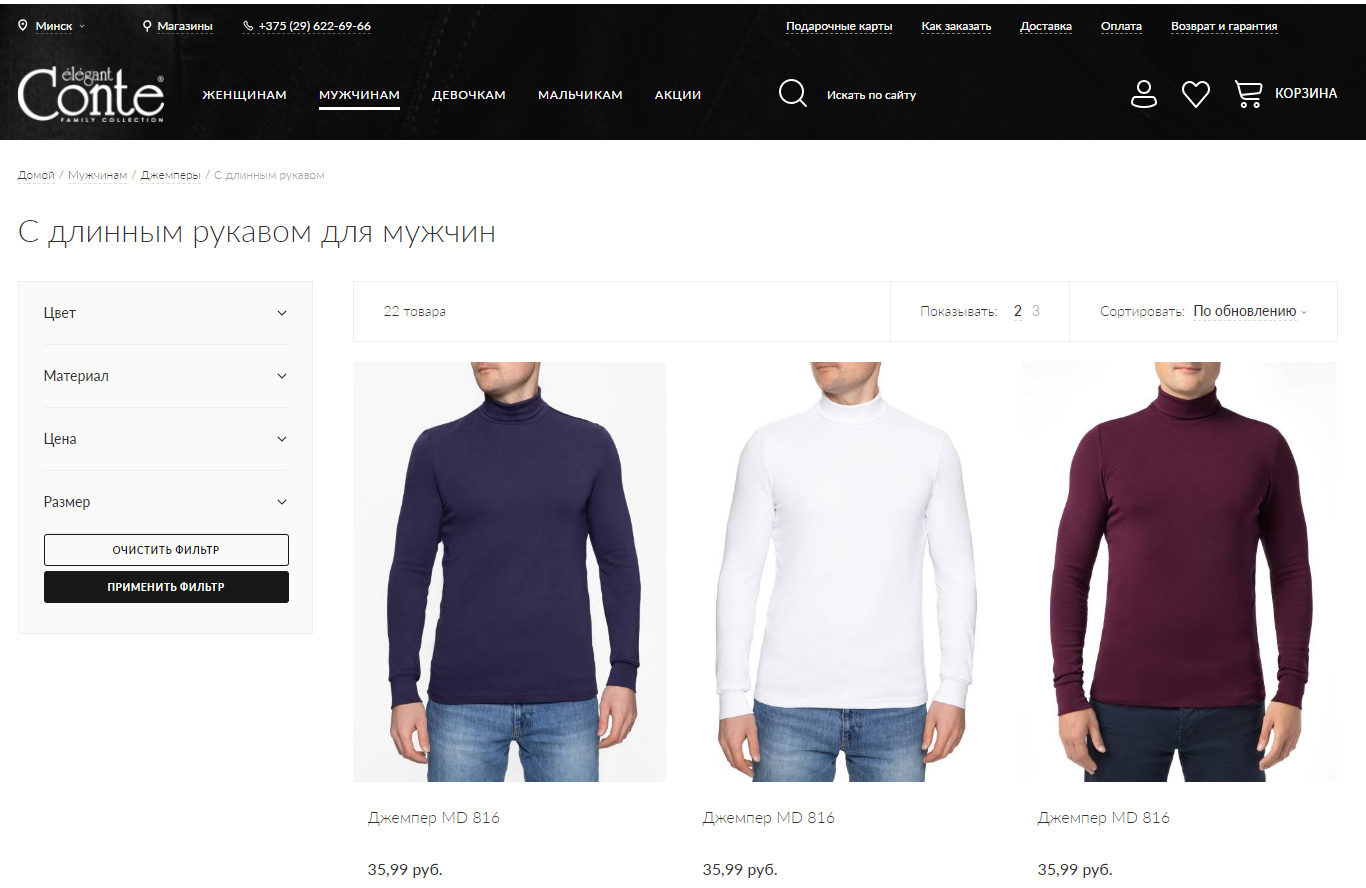


Рисунок 1.2 – Каталог интернет-магазина «Конте Шоп»

Интернет-магазин «Конте Шоп» предоставляет потенциальному покупателю следующую информацию:

* список товаров выбранной категории с наименованием товара и его ценой;
* количество товаров в категории;
* возможность быстрого просмотра карточки товара;
* возможность выбрать количество товаров в блоке;
* возможность сортировки;
* фильтр по таким параметрам, как цвет товара, материал, цена и размер.

На рисунке 1.3 представлена карточка товара, которая открывается при нажатии на блок товара.

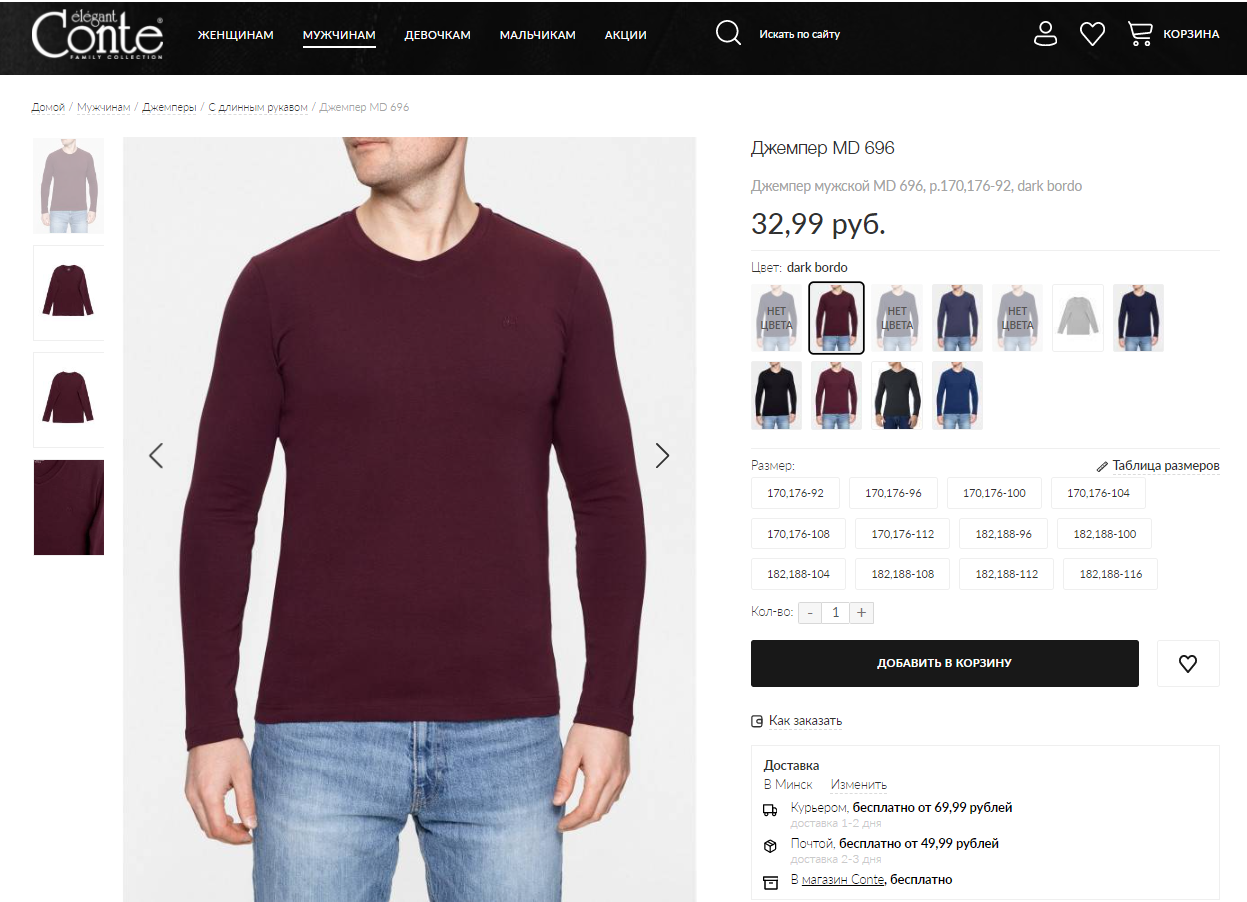


Рисунок 1.3 – Страница определенного товара интернет-магазина «Конте Шоп»

Карточка товара приветствует покупателя большим слайдером с фотографиями выбранного товара, а также возможностью более детально просмотреть фото. Справа от слайдера отображен блок с ценой, допустимыми цветовыми решениями данной модели, доступными комбинациями размера-роста, возможностью выбрать сразу несколько одинаковых товаров для добавления в корзину, а также кнопкой для добавления товара в список желаемых

При нажатии кнопки «Добавить в корзину» появляется popup-блок, который отображает добавленный в корзину товар и предоставляет покупателю выбор между возможностью продолжения покупки и переходом к оформлению заказа. Данное всплывающее окно отображено на рисунке 1.4.



Рисунок 1.4 – Всплывающее окно

Выбрав вариант «Продолжить покупки» всплывающее окно скрывается и нам снова предоставляется возможность «прогулки» по каталогу.

При переходе в корзину покупателю предоставляется:

* возможность просмотра списка выбранного товара;
* возможность изменить количество определенных позиций в списке;
* возможность ввода номера дисконтной карты или кода, предоставляющего скидку;
* сумма к оплате.

Страница с содержимым корзины покупок отображена на рисунке 1.5.

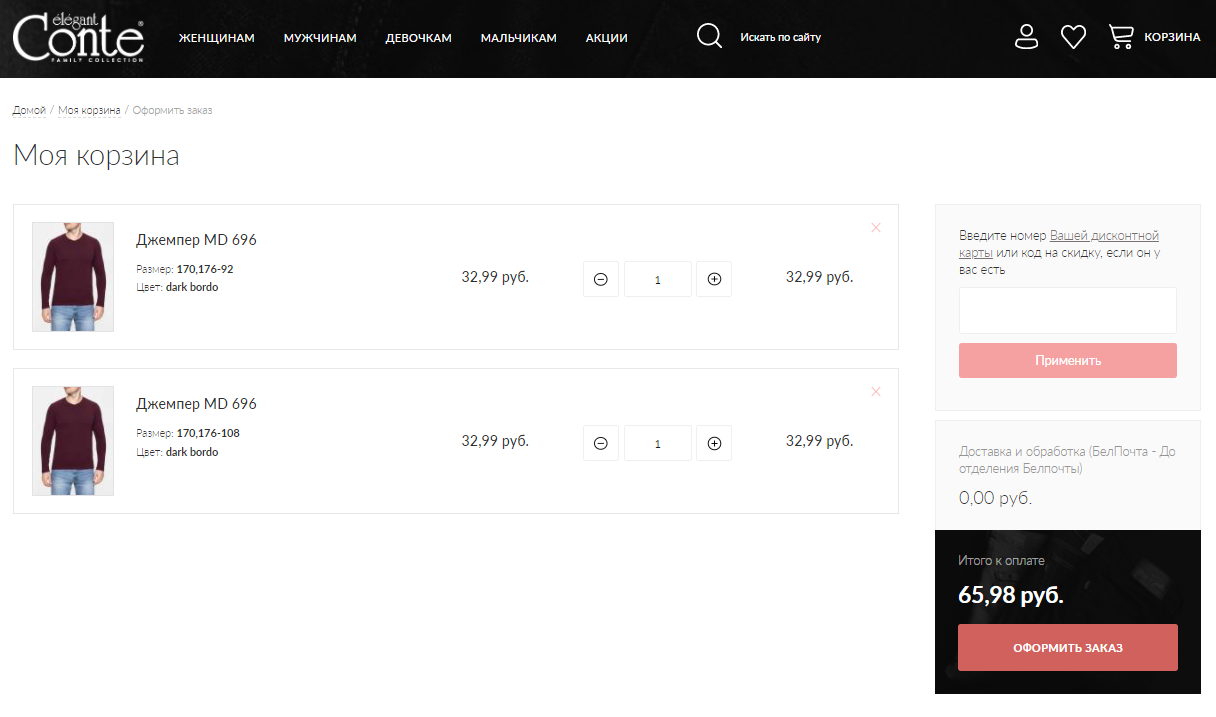


Рисунок 1.5 – Содержимое корзины и оформление заказа в интернет-магазине «Конте Шоп»

Одной из важных отличительных черт данного сервиса можно назвать строго выдержанное оформление, как элементов каталога, так и элементов всего сайта, которое на загромождено большим количеством ненужной информации. Мобильная версия сайта также выполнено в общей стилистике, но имеет ряд недостатков, такие как отсутствие возможности увеличения фотографии без перехода на новую страницу, трудночитаемый шрифт, обусловленный малым размером символов, наличие всплывающих окон, в которых запрашиваются личные данные.

Несмотря на то, что данный сервис обладает большим количеством достоинств, стоит отметить и ряд общих недостатков:

* отображение товаров при отсутствии возможности добавления в корзину заказа;
* отсутствие фото для некоторых цветовых вариаций товара;
* отсутствие информирования о доступном для заказа количестве единиц товара;
* отсутствие детальной информации о добавленном в корзину товаре;
* наличие всплывающих окон с просьбой предоставления адреса электронной почты для последующих рекламных рассылок.

В мобильной версии сайта также было выявлено несколько недостатков: это отсутствие возможности увеличения фотографии без перехода на новую страницу и трудночитаемый шрифт, обусловленный малым размером символов.

* + 1. Veore.by

Еще одним рассматриваемым решением, существующим в данной предметной области, является интернет-магазин компании «Veore». Данный сервис ориентирован на продажу товара разным возрастным группам. В таких направлениях как одежда женщинам, мужчинам, детям, красота, товары для дома и гаджеты имеется более двух тысяч позиций. На рисунке 1.6 отображена предлагаемая продукция одной из возможных категорий.

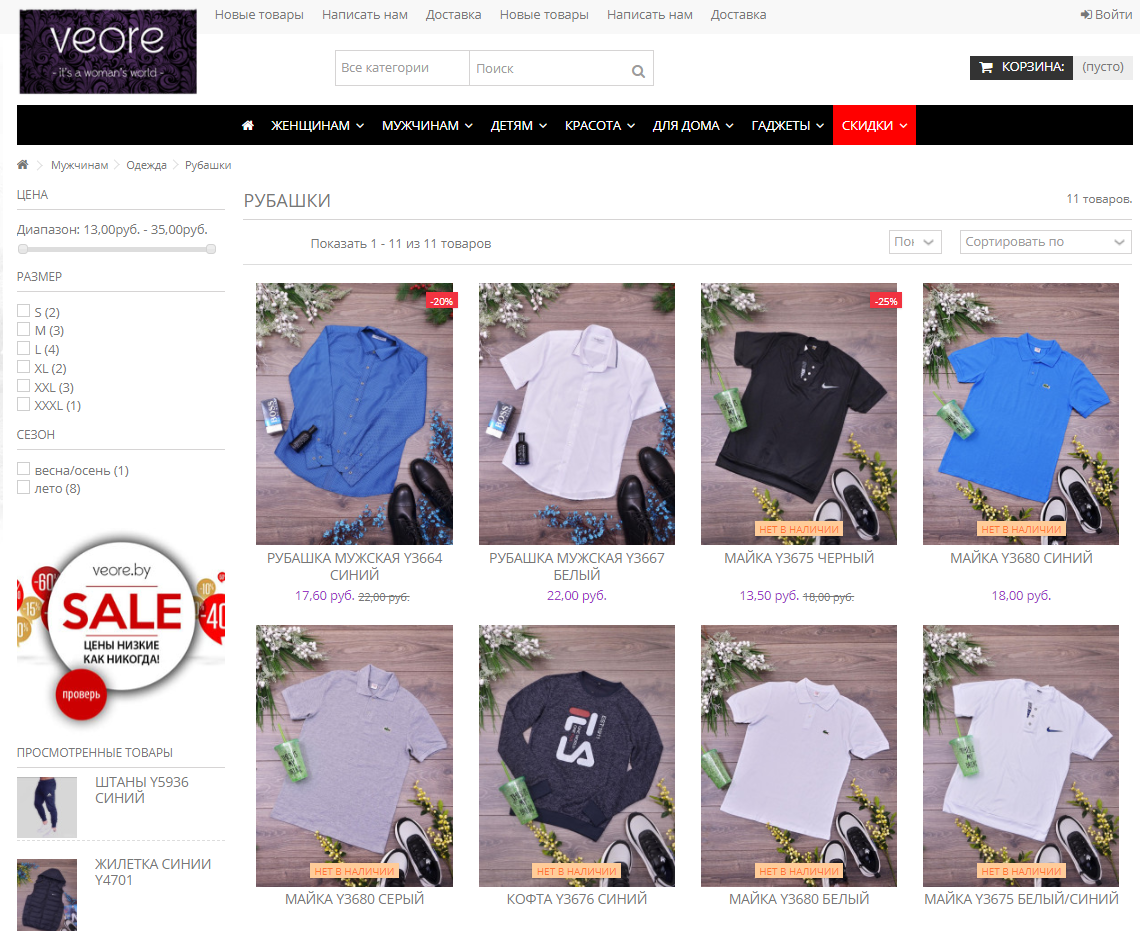


Рисунок 1.6 – Категория «Рубашки» интернет-магазина «Veore»

Самое первое отличие, которое бросается в глаза после рассмотрения интернет-магазина компании «Conte», является то, что данный сервис предполагает не только продукцию легкой промышленности, но и товары таких категорий как «Красота» и «Гаджеты».

Второе отличие, которое становится очевидным при пользовании сервисом – это отсутствие или неработоспособность некоторых привычным всем функций. Фильтр товара в каталоге осуществляется только по трем параметрам: диапазон цен, размер и сезон, а просмотр характеристик возможен только при переходе на страницу конкретного товара. Блок, позволяющий определить количество отображаемых на одной странице товаров некорректно выровнен.

Большим плюсом является то, что фильтр реализован с применением технологии Ajax, которая предоставляет возможность изменять отображение товаров при активных позициях фильтра без перезагрузки страницы. Рядом с названием элемента фильтра размещена информация о количестве товаров с одноименным параметром.

После перехода на страницу определенного товара пользователю предоставляется возможность просмотра такой информации, как допустимые размеры, наличие на складе в текущий момент, кнопка «Добавить в корзину», производитель, а также множество дополнительной информации из разделов «параметры», «информация» и «возврат». Результат перехода изображен на рисунке 1.7.

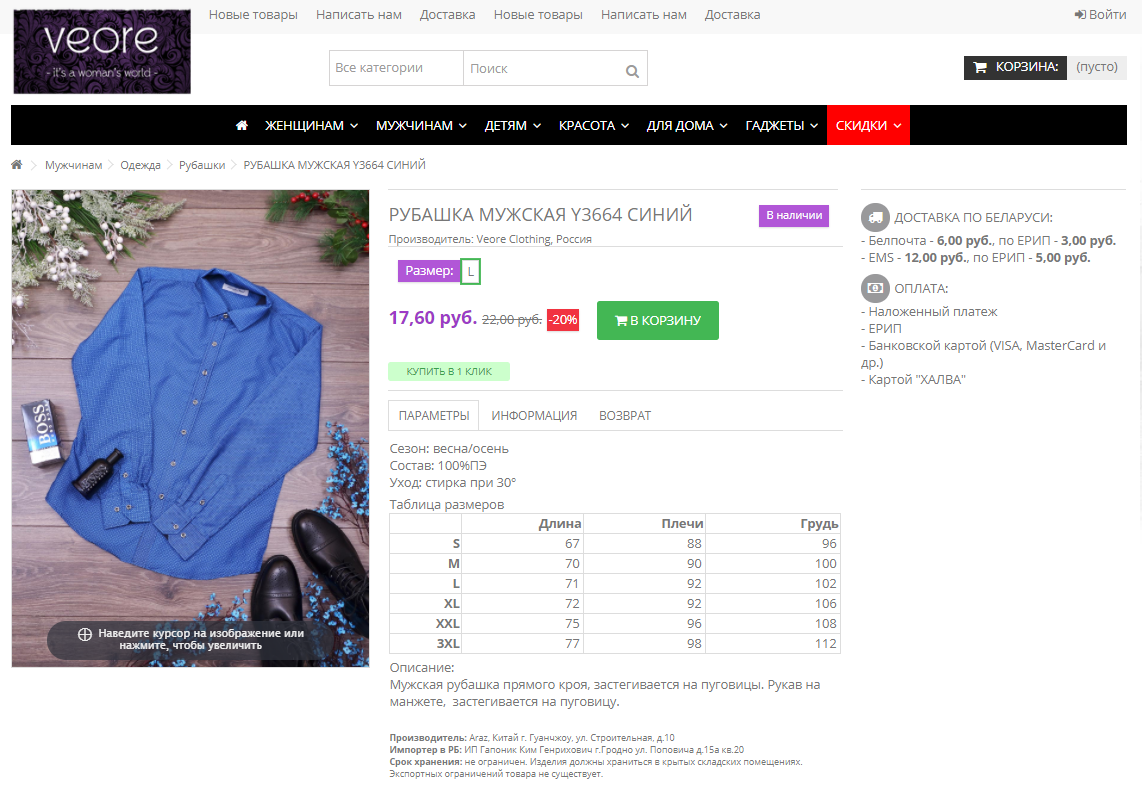
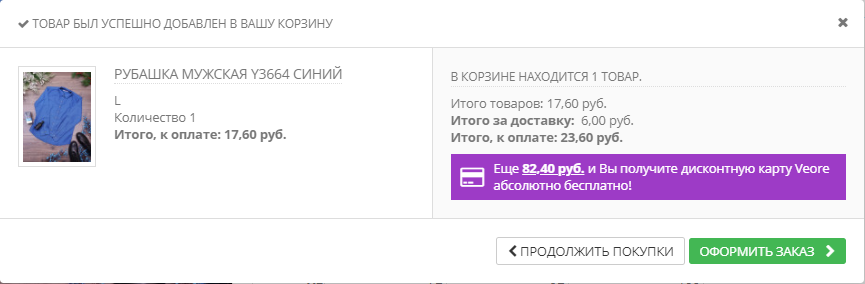


Рисунок 1.7 – Страница с детализацией товара

Отсутствие фильтрации по цвету обусловлено наличием определения цвета в названии товара, что нельзя трактовать как недостаток, хотя это и препятствует поиску товаров по определенному цвету, т.к. глобальный поиск по сайту отобразит товары из всех категорий.

Таблица размеров является уникальной для каждого товара, определяемой на основании принадлежности товара к определенной категории и полу.

После нажатия кнопки «Добавить в корзину» пользователю отображается всплывающее окно, гласящее об успешном выполнении операции и предлагает два варианта продолжения работы: продолжить покупки и перейти к оформлению заказа. Данное окно отображено на рисунке 1.8. Нажатие на кнопку продолжения также не перезагрузит страницу, что положительно скажется на работе покупателя.

Рисунок 1.8 – Всплывающее окно при добавлении в корзину

Перейдя к оформлению заказа пользователю будет отображена страница, содержащая содержимое корзины, выбор способа доставки и оплаты, а также данные аккаунта. Дополнительными функциями являются возможность удаления товара из списка и применение скидочного промокода.

Увеличение количества определенного товара влечет за собой ошибку, текст которой отображается с помощью функции вызова модальных окон *alert()*, что тоже негативно сказывается на работе с данной платформой. При условии сбора корзины не авторизованным пользователем нам будет предложено пройти процесс авторизации либо регистрации, введя данные в соответствующие поля и нажав кнопку «Оформить заказ», предварительно установив флажок в чекбокс «Регистрация?», что не является интуитивно понятным. Это изображено на рисунке 1.9.

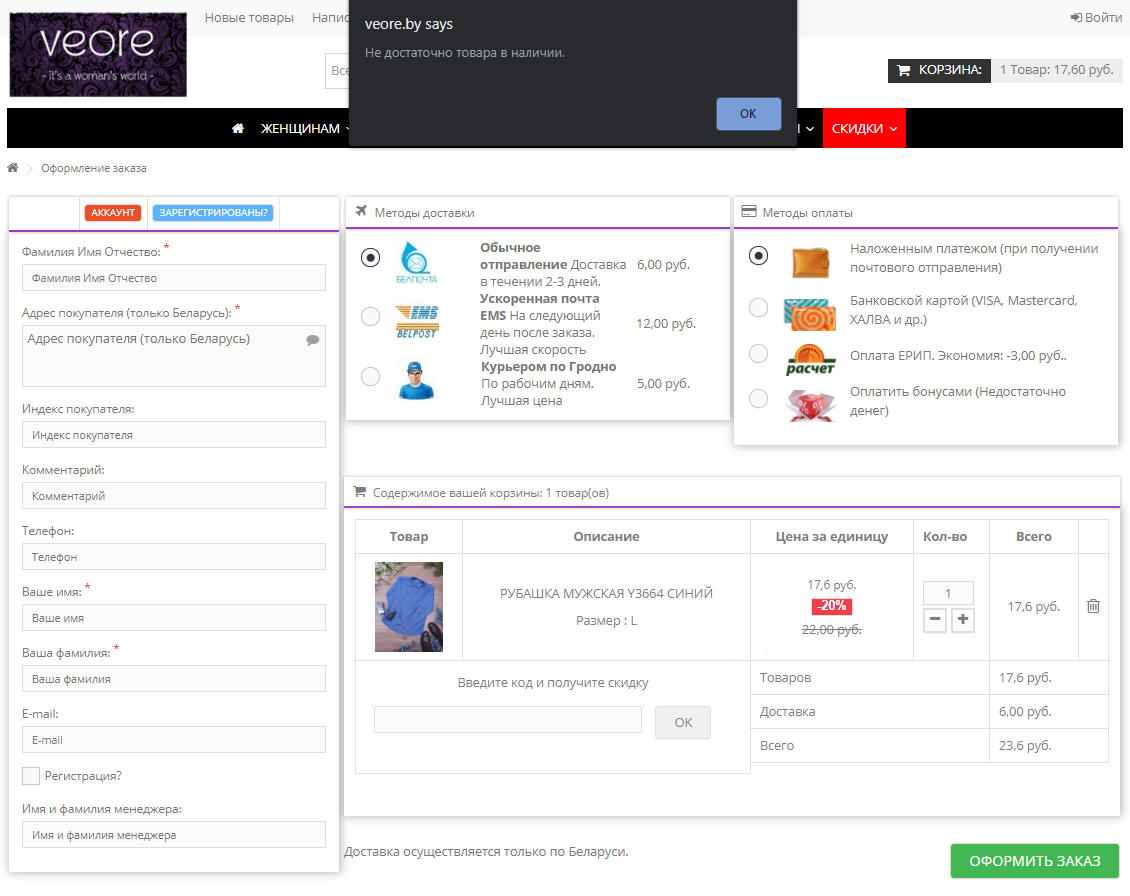


Рисунок 1.9 – Страница оформления заказа

Подведя итоги анализа данного интернет-магазина можно выявить следующие недостатки:

* неработоспособность некоторых доступных пользователю функций
* отсутствие в фильтре множества параметров товаров
* отсутствие возможности выбрать определенное количество товара на странице самого товара, изменение которого производится только в корзине, а попытки изменения приводят к возникновению ошибки
* уведомление посредством модальных окон.
  + 1. ОАО "Купалинка"

Очередным сервисом, существующим в данной предметной области, является интернет-магазин компании «Купалинка».

Сервис предлагает к продаже бельевой и верхний трикотаж для взрослых и детей. По способу взаимодействия с пользователем интернет-магазин имеет сходства с Conte Shop: интерфейс минималистичен, в каждый момент времени отображаются только самая необходимая информация. Такой способ взаимодействия с пользователем позволяет сконцентрировать внимание на подробном изучении информации о товарах.

Чуть ниже, на рисунке 1.10 можно увидеть раздел «Мужчинам» каталога продукции.

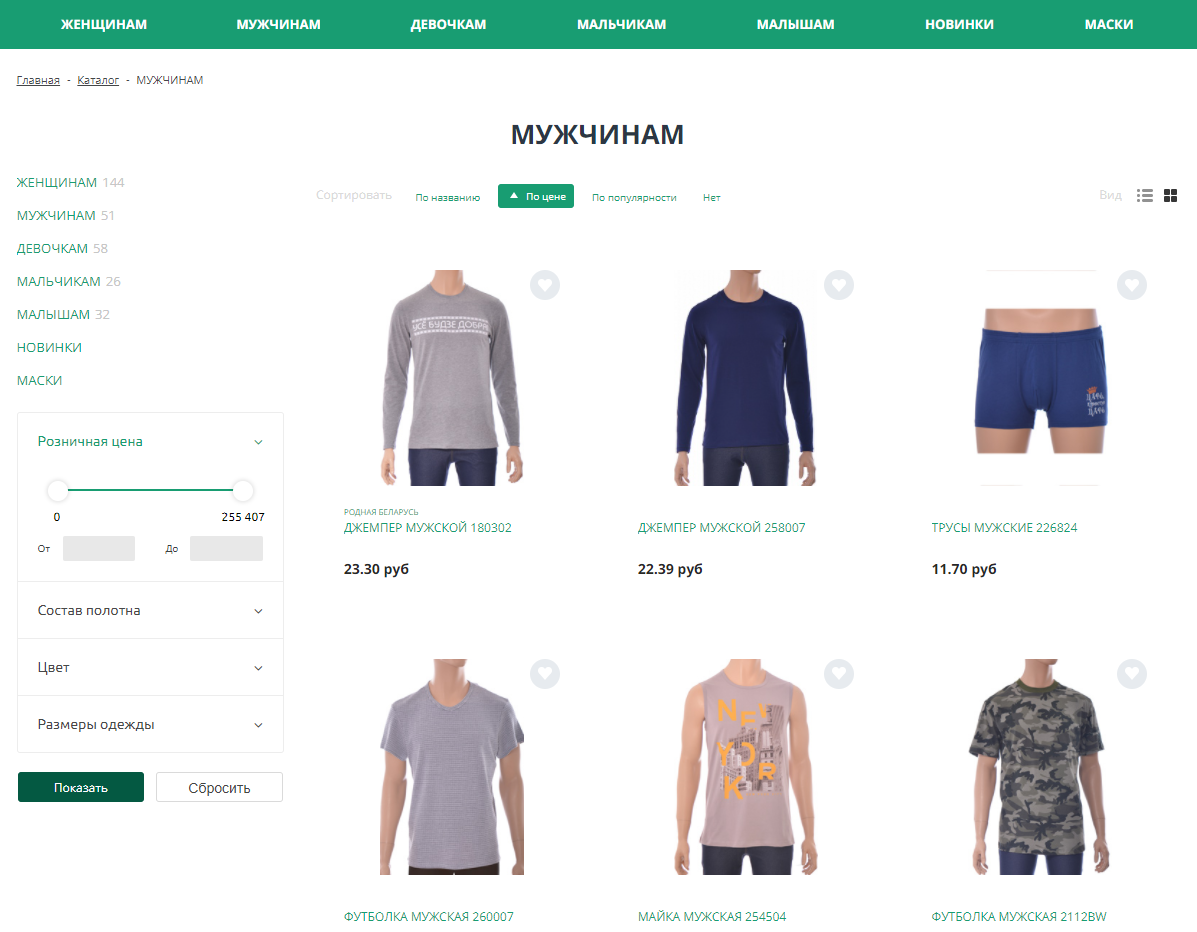


Рисунок 1.10 – Раздел «Мужчинам» интернет-магазина «Купалинка»

Возможности данного сайта весьма обширны, пользователь может выбрать конкретную категорию для просмотра, воспользоваться фильтром, который поможет найти необходимые товары при помощи указания диапазона цен, состава полотна, цвета и размера желаемого продукта. Также имеется возможность сортировки по названию, цене и популярности как от меньшего к большему, так и наоборот, функции «Быстрый просмотр», «Купить в один клик» и добавить товар в список желаемого. Изменяя вариант отображения товаров сайт преображается, что позволяет совершать меньше действий для просмотра таких параметров модели, как диапазон размеров и доступные цвета модели.

Воспользовавшись быстрым просмотром, покупателю отображается всплывающее окно, которое отображено на рисунке 1.11.

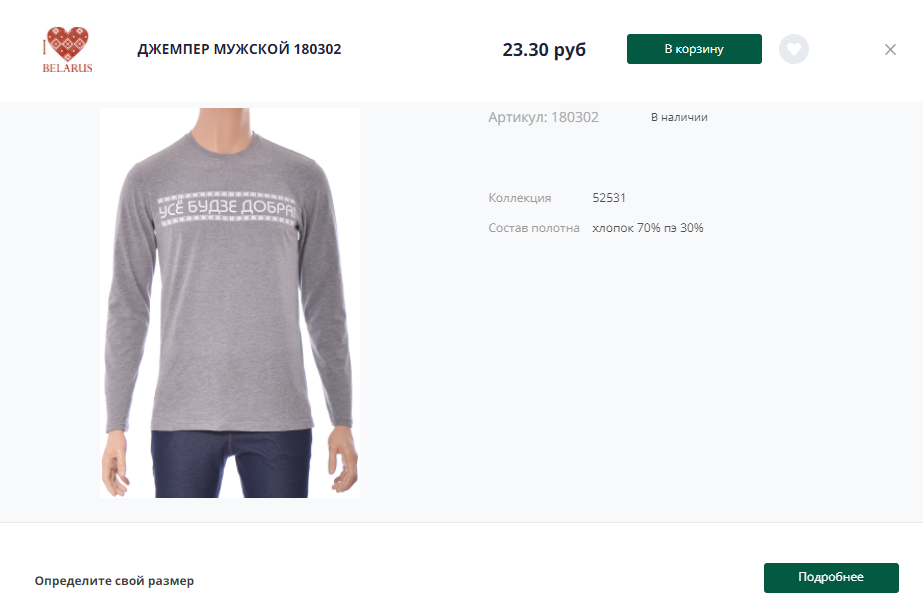


Рисунок 1.11 – Быстрый просмотр товара

Как можно заметить, быстрый просмотр не позволяет в полной мере оценить доступные размерные и цветовые вариации товара, что в некоторой мере сковывает действия покупателя. Клик на картинку либо наименование товара адресует покупателя на страницу товара, в которой можно просмотреть параметры модели и добавить необходимый в корзину. Описание модели строится на основании четырех атрибутов: наименования модели, артикула, цвета и размера.

Добавив необходимое количество товаров, пользователь переходит в корзину для оформления заказа. Страница корзины отображена на рисунке 1.12. Из доступного функционала можно отметить возможность изменения количества определенного товара, автоматический подсчет суммы заказа, поле для ввода кода купона, возможность добавить товар в список желаемого и удалить из корзины необходимые позиции. Сложно не заметить неровное отображение блока с товарами, который искусственно выровнен по правому краю, что создает большое количество неиспользуемого пространства слева от последнего.

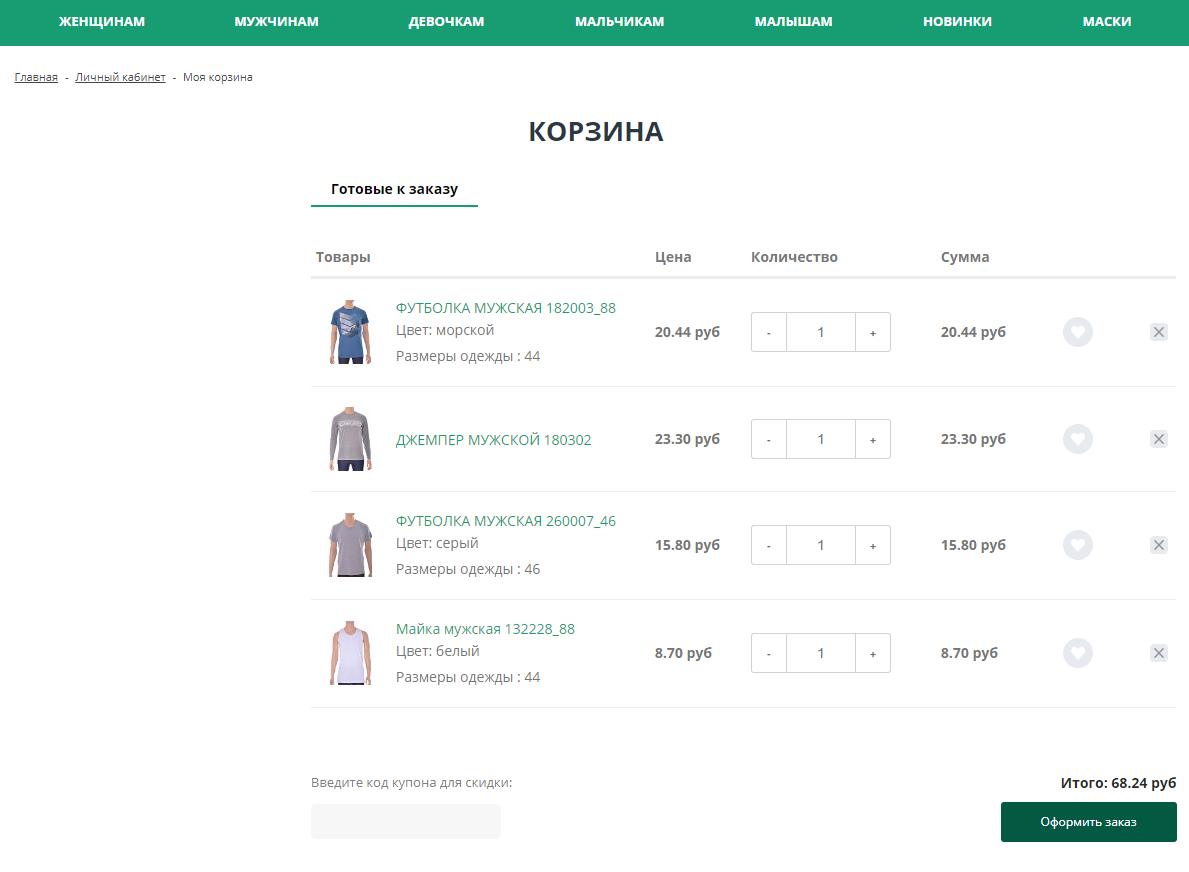


Рисунок 1.12 – Корзина интернет-магазина «Купалинка»

Следующим шагом, который совершает пользователь для заказа является переход к оформлению заказа. Даже авторизованному пользователю сразу же отображается форма для ввода информации о покупателе, среди которых имеются такие поля как название компании, юридический адрес, ИНН, КПП, контактное лицо, что уже смутит подавляющее большинство покупателей, ведь нигде нет информации о том, что интернет-магазин является B2B-ориентированным, что подтверждается еще и наличие только одного вида оплаты заказа – наличными.

Детализация заказа, а также возможность написать комментарий к нему размещены снизу от формы для ввода персональной информации.

Несмотря на высокий показатель удобства пользования, данный сервис не обделен недочетами, основными из которых являются:

* некорректно работающие функции сортировки;
* отсутствие ограничения цен в фильтре;
* искажение положений блоков;
* некорректное отображение текста на большинстве страниц при просмотре с мобильного устройства.

## Постановка целей и задач дипломного проектирования

Исходя из результатов анализа предметной области, основными недостатками существующих решений являются следующие:

− наличие недоработок мобильных версий сайта;

− отсутствие детализации предлагаемых продуктов;

− недостаточный уровень удобства пользования программным средством;

− медленная работа сайта, отображение данных с большими задержками;

− отсутствие обмена данными при работе с нескольких устройств одновременно.

Исходя из всего вышеупомянутого, целью дипломной работы является создание интернет-магазина с гибким функционалом, понятным и доступным интерфейсом, возможностью дальнейшего расширения каталога продукции, а также путем добавления новых функций. Данная система будет выполнять функции площадки для ведения дополнительной коммерческой деятельности предприятия.

Реализация поставленной цели предполагает решение следующих задач:

* выбор средства разработки модуля интернет-магазина;
* разработка интернет-магазина в соответствии со структурой базы данных, который будет обладать следующим функционалом:
  + - отображение каталога продукции компании, включая такие поля, как наименование, цена по трем категориям, размерная сетка, состав сырья, допустимые цвета и наличие определенного количества изделий на складе;
    - осуществление сортировки по группам товаров;
    - осуществление возможности поиска по нескольким параметром продукта;
    - осуществление регистрации и авторизации пользователей, а также разграничение прав между ними;
    - создание зарегистрированным пользователем заявки на покупку собранной корзины, а также выбор способа оплаты и доставки;
    - осуществление обратной связи с пользователем через e-mail.

# 2 Моделирование предметной области, разработка функциональных требований и составление их спецификации

## Общие сведения и требования к работе программного продукта

Функциональным назначением разрабатываемого программного решения является предоставление пользователю возможностей для проведения онлайн покупок.

Пользователем в данном случае может выступать любой человек, который имеет персональный компьютер с доступом к сети Интернет. Для использования программного средства не требуется специальная подготовка или обучение пользователей.

Предполагается возможность одновременной эксплуатации разрабатываемого программного продукта большим числом пользователей. При этом отсутствуют какие-либо ограничения, накладываемые на предметную область, в рамках которой возможно его применение.

Исходя из предполагаемого использования, можно заключить, что проектируемое программное решение должно реализовывать следующие три группы функций:

* управление каталогом, в частности товарами и их параметрами;
* управление результатами посещений пользователями, т.е. непосредственная обработка заказов;
* управление пользователями.

## Описание функциональности программного продукта

Средством представления функциональности программного средства будет выступать диаграмма вариантов использования. Настоящий вид UML диаграмм позволяет описать функциональность системы на концептуальном уровне посредством построения взаимосвязей между двумя основными элементами: прецедентами и актерами.

Каждый прецедент Use-Case диаграммы отображает один из вариантов использования программного средства конкретным пользователем. Средства UML позволяют установить отношения обобщения для актеров на диаграмме, таким образом отсутствует необходимость в дублировании одинаковых прецедентов на диаграмме использования.

На рисунке 2.1 представлена обобщенная диаграмма вариантов использования разрабатываемого программного средства, на которой отражены группы функций, доступные для действующих лиц внутри рамок проектируемой системы.

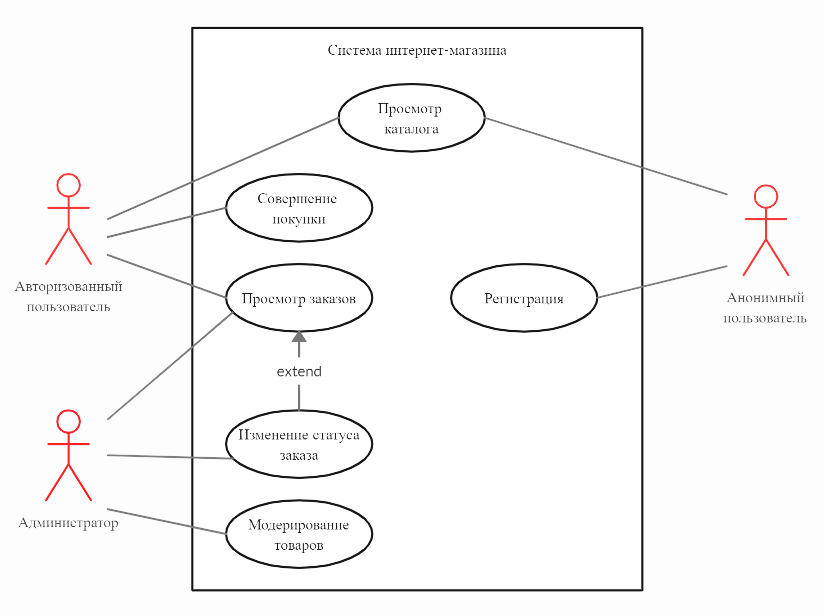


Рисунок 2.1 - Обобщенная диаграмма вариантов использования

В процессе анализа литературы по теме дипломного проектирования, а также в течение обзора существующих программных решений в данной предметной области был определён список действующих лиц и ролей, которые будут взаимодействовать с разрабатываемой системой, и для которых, в свою очередь, данная система должна будет предоставлять ряд функций. Среди таких действующих лиц мы можем выделить следующие:

* анонимный (не авторизованный) пользователь;
* пользователь (авторизованный);
* администратор.

Рассмотрим всех действующих лиц проектируемого решения в порядке их следования. Анонимный пользователь – базовая роль для остальных актеров на диаграмме прецедентов, к данному типу мы относим всех пользователей, о которых разрабатываемое решение не имеет никакой информацией, которая бы позволила идентифицировать в них зарегистрированного пользователя. Однако стоит отметить, что данный тип действующих лиц является одним из основных в рамках программного средства, а многие из пользователей, которые используют системы подобного типа, не имеют персонального аккаунта.

На рисунке 2.2 отображены функциональные возможности анонимного пользователя.

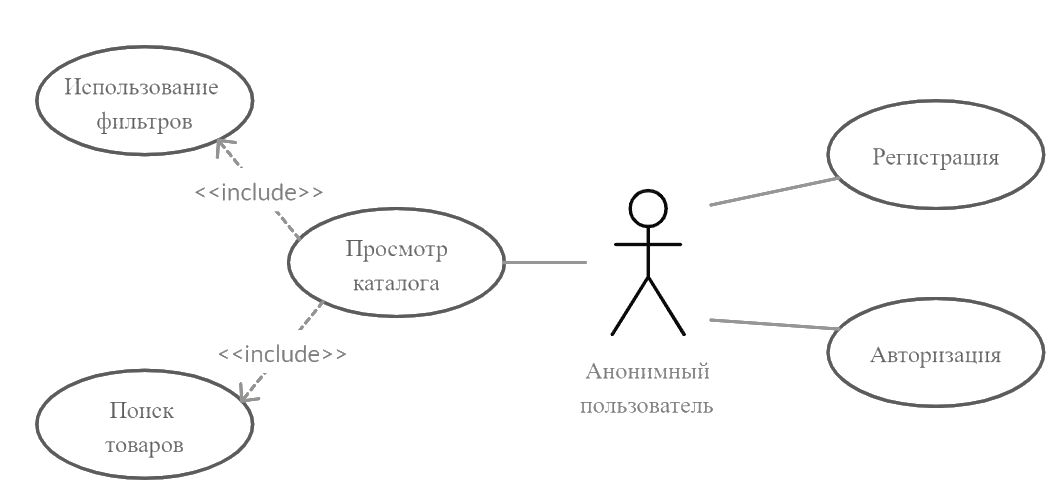


Рисунок 2.2 – Функциональные возможности анонимного пользователя

Наличие отдельного пользовательского профайла обусловлено тем, что для правильного функционирования системы необходимо некоторое количество пользовательских данных. Для этого есть несколько причин: возможность формирования заказа предусматривает бронирование определенного количества товаров для пользователя, совершение оплаты и доставка также не осуществимы по той же причине.

Данному типу пользователей предоставляются следующие функции в рамках системы:

* регистрация;
* авторизация;
* просмотр каталога товаров.

Регистрация пользователя в системе подразумевает создание персонального аккаунта, а также предоставление пользователям прав роли типа «Пользователь». В процессе регистрации пользователю необходимо ввести персональные данные среди которых: адрес электронной почты, пароль, ФИО и город. В последствии адрес электронной почты и персональный пароль будут использоваться при осуществлении идентификации пользователей системы.

Процессом авторизации будем называть идентификацию пользователя по предоставленным адресу электронной почты и персональному паролю, зарегистрированного в системе пользователя и дальнейшее предоставление ему прав доступа в соответствии с занимаемой в системе ролью.

Основной функцией, предоставляемой программным решением для пользователей с данным типом роли является возможность просмотра каталога товаров, разделенных на категории, фильтрация отображаемых данных посредством ввода определенных критериев отображения данных, а также возможность использования поиска.

Следующим видом актера, изображенным на диаграмме прецедентов, является авторизованный пользователь. Данный тип роли наследует все функциональные возможности анонимного пользователя, но кроме уже рассмотренных функций, пользователю также предоставляется возможность управления личными данным, формированием заказа и выбором способов оплаты и доставки.

На рисунке 2.3 детально отображены возможности, которые доступны для пользователей с данной ролью.

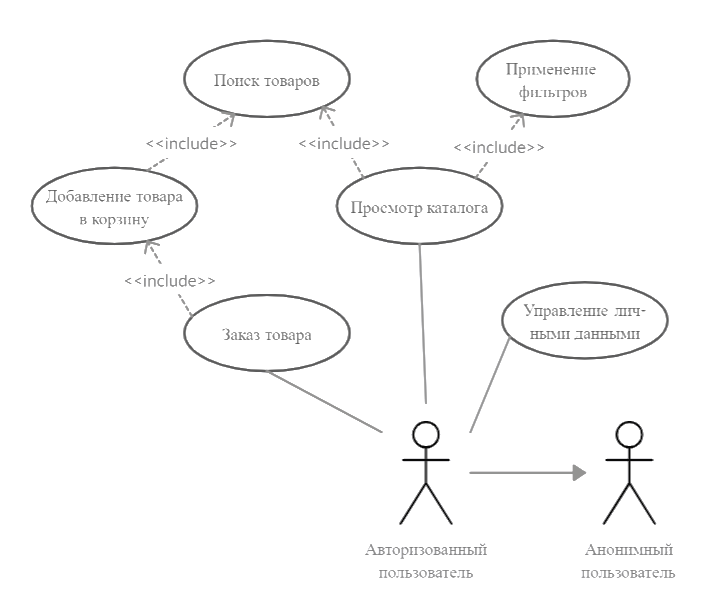


Рисунок 2.3 – Возможности авторизованного пользователя

Основной функцией, предоставляемой программным решением является возможность осуществления заказа товаров из собранной пользователем корзины. Данная функция подразумевает открытие каталога товаров, осуществление логики переходов между страницами в процессе работы пользователя с программным средством, формирование корзины, которая основывается на выбранных пользователем товарах в процессе работы с сервисом, сохранение данных по окончанию процесса поиска и выбора товаров, а также оформление заказа.

В процессе формирования заказа должны осуществляться все необходимые проверки введённых и выбранных пользователем значений, осуществляться отображение элементов управления, а также следует обеспечить корректное функционирование переходов между отдельными страницами сайта.

Последним типом пользовательских ролей, представленных на диаграмме прецедентов, является администратор. Администратор – тип роли, предназначенный для администраторов системы. Пользователи с данной ролью должны обладать максимальным набором функций из списка возможных. Администратор включает в себя возможности остальных типов ролей. Основное предназначение данной роли – администрирование системы в целом и контроль над всеми процессами, происходящими внутри системы. Исходя из этого, пользователи данного типа агрегируют возможности остальных ролей. При развертывании программного средства в системе должен существовать как минимум один пользователь с ролью данного типа.

Полный список возможностей, внесенных для пользователей данного типа, изображен на рисунке 2.4.

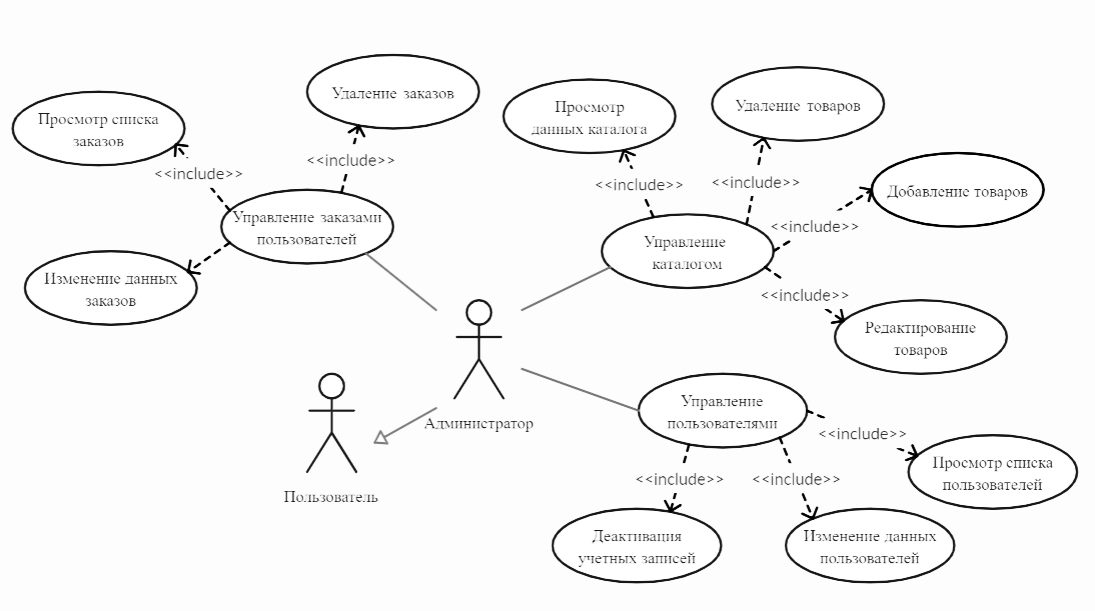


Рисунок 2.4 – Функциональные возможности администратора

Для обеспечения работы сервиса важно знать количество существующих пользователей, иметь возможность просмотреть список пользователей. Но вместе с тем, стоит учитывать, что доступ к персональным данным пользователей должен быть у строго ограниченного числа лиц. В рамках системы только пользователи с ролью типа «Администратор» имеют доступ к персональным данным других пользователей, который позволяет им устанавливать роли для этих учетных записей, а также просматривать все внутренние данные, которые с ними связаны.

С течением времени в подобного рода сервисах скапливается достаточное количество пользователей, которые не осуществляют работу с системой, или же которые по разным причинам утратили доступ к своему аккаунту. Для повышения производительности системы и очистки неактуальных данный, администратор может воспользоваться деактивацией старых аккаунтов. При деактивации аккаунта происходит удаление пользовательских данных из системы, однако удаление совершенных заказов и списка просмотренных им товаров затронут не будет. Это реализовано с целью возможности формировать статистические данные.

## Разработка информационной модели

На основании функциональной модели была разработана информационная модель программного решения. На рисунке 2.5 представлена информационная модель проектируемой системы.

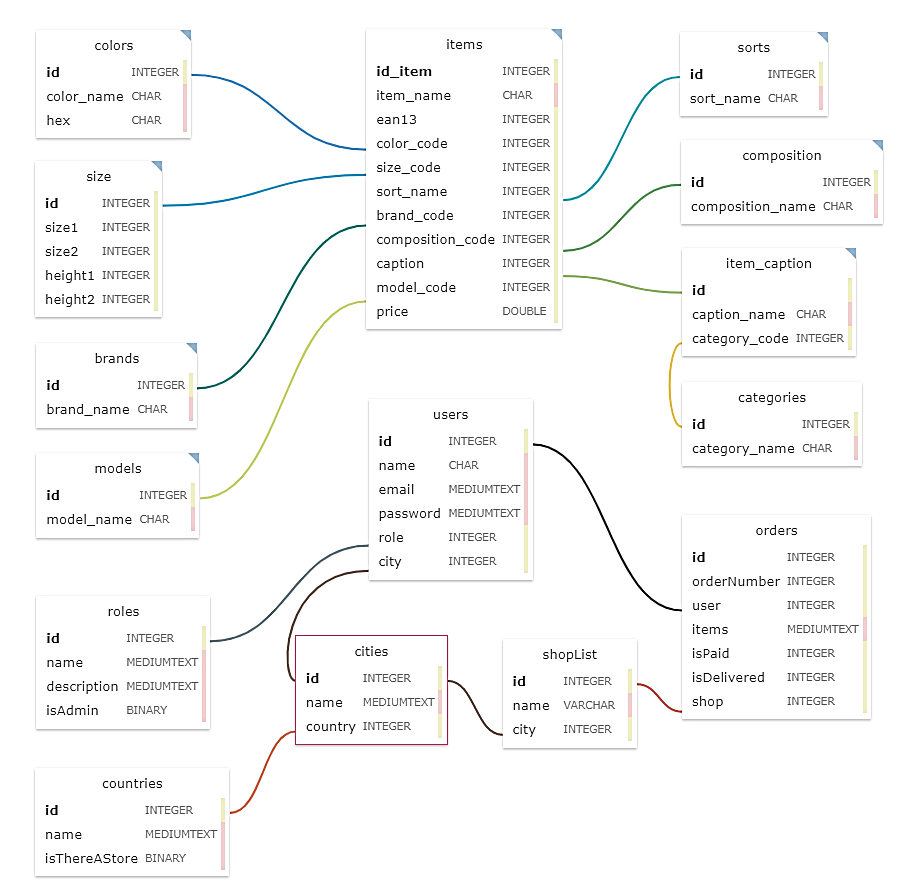


Рисунок 2.5 – Информационная модель проектируемой системы

В процессе анализа предметной области были выделены следующие типы сущностей:

User – сущность, хранящая представляющая собой список пользователей в рамках системы. Атрибут Id – идентификатор пользователя, Name – имя пользователя, Surname – фамилия пользователя, Email – электронный адрес пользователя, Password – хэш пароля пользователя, PasswortSalt – случайное значение, использованное при хешировании пароля, RoleType – тип роли пользователя, City – город, указанный пользователем при регистрации.

Role – таблица, хранящая все заданные типы ролей учетных записей. Атрибут Id – идентификатор роли, Name – краткое наименование роли, Description – полное описание роли, IsAdmin – флаг, указывающий на принадлежность данной роли к администраторам.

Cities – список городов. Атрибут Id – идентификатор города, name – название города, country – страна, isThereAStore – флаг, означающий наличие хотя бы одного магазина в данном городе.

Country – список стран. Атрибут id – идентификатор, name – название страны, StoreCount –число, означающее количество магазинов в данной стране.

ShopList – сущность, содержащая список всех действующих магазинов для реализации возможности самовывоза. Атрибут id – идентификатор магазина, name – название магазина, city – номер города, address – адрес магазина.

Items – одна из главных сущностей, которая содержит в себе всю информацию о имеющихся товарах. Атрибут id\_item – идентификатор товара, item\_name – наименование товара, ean13 – штрих-код в формате EAN-13, color\_code – код цвета, size\_code – код размера, sort\_name – код сорта товара, brand\_code – код торговой марки либо коллекции, composition, caption – полное наименование товара, model\_code – номер модели, priсe – цена на момент получения товара.

Colors – список цветовых вариаций моделей. Атрибут id – идентификатор цвета, color\_name – наименование цвета, hex – код цвета в формате HEX.

Size – сущность, хранящая в себе данные о всевозможно допустимых вариациях размера-роста. Атрибут id – идентификатор размера, size1 – размер 1, size2 – размер 2, height1 – рост 1, height2 – рост 2, fullSize – полное обозначение размера.

Brands – список коллекций и торговых марок. Атрибут id – идентификатор коллекции, brand\_name – наименование коллекции, isBrand – флаг, отвечающий за обозначение торговой марки или бренда.

Models – сущность, представляющая собой сведения о моделях. Атрибут id – идентификатор модели, model\_name – номер модели, sizeRange – диапазон размеров модели.

Sorts – список всевозможных сортов изделий. Атрибут id – идентификатор сорта, sort\_name – обозначение сорта.

ItemCaption – сущность, содержимым которой является список наименований изделий. Атрибут id – идентификатор наименования, caption\_name – наименование изделия, category\_code – код категории, которой будет присвоен товар при добавлении в таблицу Items.

Categories –список категорий, на которые будет разделяться весь объем товаров. Атрибут id – идентификатор категории, category\_name – наименование категории, level –уровень категории.

Orders – вторая по важности сущность. В ней содержится вся информация о совершенных пользователями заказах. Атрибут id – идентификатор заказа, orderNumber – внутриорганизационный номер заказа, user – идентификатор пользователя, совершившего заказ, items – список пар «идентификатор товара – количество», отвечающих за определение содержимого корзины пользователя на момент совершения заказа, isPaid – флаг, положение которого меняется в зависимости от статуса оплаты заказа, isDelivered – показатель доставки заказа, shop – идентификатор магазина, в котором будет собран заказ для последующего самовывоза покупателем, orderDate – дата и время совершения заказа.

## Разработка модели взаимодействия пользователя с интерфейсом

В процессе анализа программных решений в области дипломного проектирования, отмечалась необходимость обеспечения высокого уровня интерактивности при взаимодействии с пользователем. На рисунке 2.6 изображена модель взаимодействия пользовательского интерфейса, управляющей логики и данных программного средства, именующаяся MVC.

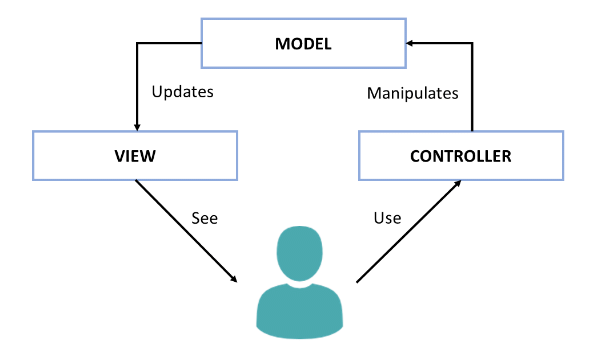


Рисунок 2.6 – Схема MVC

На данной диаграмме отражены классические элементы MVC. Это способ организации кода, который предполагает выделение блоков, отвечающих за решение разных задач. Один блок отвечает за данные приложения, другой отвечает за внешний вид, а третий контролирует работу приложения.

На данной модели отображены следующие компоненты:

* модель (Model) предоставляет данные и реагирует на команды контроллера, изменяя своё состояние;
* представление (View) отвечает за отображение данных модели пользователю, реагируя на изменения модели;
* контроллер (Controller) интерпретирует действия пользователя, оповещая модель о необходимости изменений.

Модель предоставляет данные и методы работы с ними: запросы в базу данных, проверка на корректность. Модель не зависит от представления, т.е. фактически ей не известно, каким способом визуализировать данные и контроллера, не имеющего точек взаимодействия с пользователем, просто предоставляя доступ к данным и управлению ими. Модель строится таким образом, чтобы отвечать на запросы, изменяя своё состояние, при этом может быть встроено уведомление «наблюдателей». Модель, за счёт независимости от визуального представления, может иметь несколько различных представлений для одной «модели».

Представление отвечает за получение необходимых данных из модели и отправляет их пользователю. Представление не обрабатывает введённые данные пользователя.

Контроллер обеспечивает «связь» между пользователем и системой. Контролирует и направляет данные от пользователя к системе и наоборот. Использует модель и представление для реализации необходимого действия.

## Разработка спецификации функциональных требований

Исходя из результатов анализа исходных данных для проектируемого программного средства, можно выделить следующее: основной целью работы является создание программного продукта, который позволил бы разрешить существующие несовершенства программных средств в области, рассматриваемой в процессе дипломного проектирования.

Среди недостатков программных решений стоит отметить следующие:

− наличие недоработок мобильных версий сайта;

− отсутствие детализации предлагаемых продуктов;

− недостаточный уровень удобства пользования программным средством;

− медленная работа сайта, отображение данных с большими задержками;

− отсутствие обмена данными при работе с нескольких устройств одновременно.

В ходе проектирования и разработки необходимо продумать и реализовать следующие функциональные возможности:

* управление каталогом;
* наличие категорий и подкатегорий товаров;
* изменение вида каталога продукции;
* сортировка товаров каталога по трем параметрам;
* выбор типа валюты для отображения цен в каталоге;
* глобальный поиск по нескольким параметрам товара;
* выбор диапазона цен и размеров в каталоге;
* выбор одного либо нескольких цветов изделий;
* регистрация и авторизация пользователей;
* перемещение товаров в корзину;
* оформление заказов в режиме онлайн;
* выбор способа доставки и оплаты.

Разрабатываемое решение представляет собой программное средство, предназначенное для предоставления покупателям возможности совершать покупки в режиме онлайн. Основными функциями данного программного средства являются следующие:

* функции идентификации и аутентификации;
* система должна поддерживать создание заказов;
* система должна корректно работать на разных устройствах и операционных системах;
* система должна иметь удобный и интуитивно понятный интерфейс;
* возможность назначения администраторов системы.

Идентификация и аутентификация должны быть выполнены в соответствие со следующими требованиями:

* авторизация должна быть осуществлена по адресу электронной почты и персональному паролю;
* следует предусмотреть возможность аутентификации одного и того же пользователя с нескольких устройств.

## Разработка технических требований к программному продукту

Разрабатываемое программное решение должно обеспечивать корректное функционирование при развертывании программных модулей на сервере со следующими техническими характеристиками:

* процессор – Xeon 2.2 Ггц или более быстродействующий процессор с поддержкой инструкций типа SSE;
* оперативная память – 4 Гбайт или более;
* доступ к сети Интернет;
* доступный объем дискового пространства – не менее10 Гбайт.

Для нормального функционирования клиентской части программного продукта должны выполняться следующие технические требования:

* Intel Pentium 4/Athlon 64 или более поздней версии;
* свободное место на диске 350 Мб и более;
* оперативная память 2 Гб или более;
* 32- или 64-битная версия операционных систем Windows 7, 8, 10, Mac OS X 10.6 или более поздней версии;
* Браузер Microsoft Edge версии 11.x , Google Chrome версии 46.x, Opera версии 12.x, Mozilla Firefox 41.x.

# Проектирование архитектуры программного средства

## 3.1 Разработка архитектуры программного продукта

Закончив формулирование функциональных требований к разрабатываемой системе, а также исходя из результатов анализа существующих программных средств, можно определить основные моменты организации системы, в рамках которой будет функционировать разрабатываемое программное решение.

Процесс проектирования архитектуры программного обеспечения включает в себя сбор требований клиентов, их анализ и создание проекта в соответствии с требованиями. Удачная разработка программных средств заключается в балансе компромиссов и противоречащих требований, соответствии принципам проектирования и рекомендованным методам, выработанным со временем и дополняется современным оборудованием, сетями и системами управления.

Сопоставив цели определенного компонента программного средства со сведениями о его реализации в коде, мы получим архитектуру программного средства. Правильное понимание архитектуры обеспечит оптимальный баланс требований и результатов. Только программное обеспечение с хорошо продуманной архитектурой способно выполнять указанные задачи с параметрами исходных требований, одновременно обеспечивая максимально высокую производительность.

Программные средства в данной предметной области построены на базе клиент-серверной архитектуры. Использование других подходов приводит к сложностям связанным, со сбором результатов заполнения форм, а также синхронизацией моделей форм между отдельными клиентами внутри системы.

На рисунке 3.1 представлена диаграмма развертывания программного средства. На данной диаграмме нашел свое отражение клиент-серверных подход. Помимо блоков, отражающих рабочие устройства пользователей, на диаграмме представлен сервер, на котором развернуто приложение.

Сервер приложений представляет собой устройство, на котором развернут веб-сервер и серверная часть приложения. В рамках клиент-серверной архитектуры веб-сервер представляет собой посредника между логикой программного средства и внешним миром. Его основная задача: принимать запросы от пользователей и вызывать соответствующие методы обработки данных запросов, а также давать ответ браузеру пользователя, который, в свою очередь, будет отрисовывать страницу пользователю.

Помимо прочего, среди функций, реализуемых веб-сервером, есть и возможность возврата статического контента. Статический контент включает изображения, стили, клиентские модули приложения.

Диаграмма развертывания изображена на рисунке 3.1.

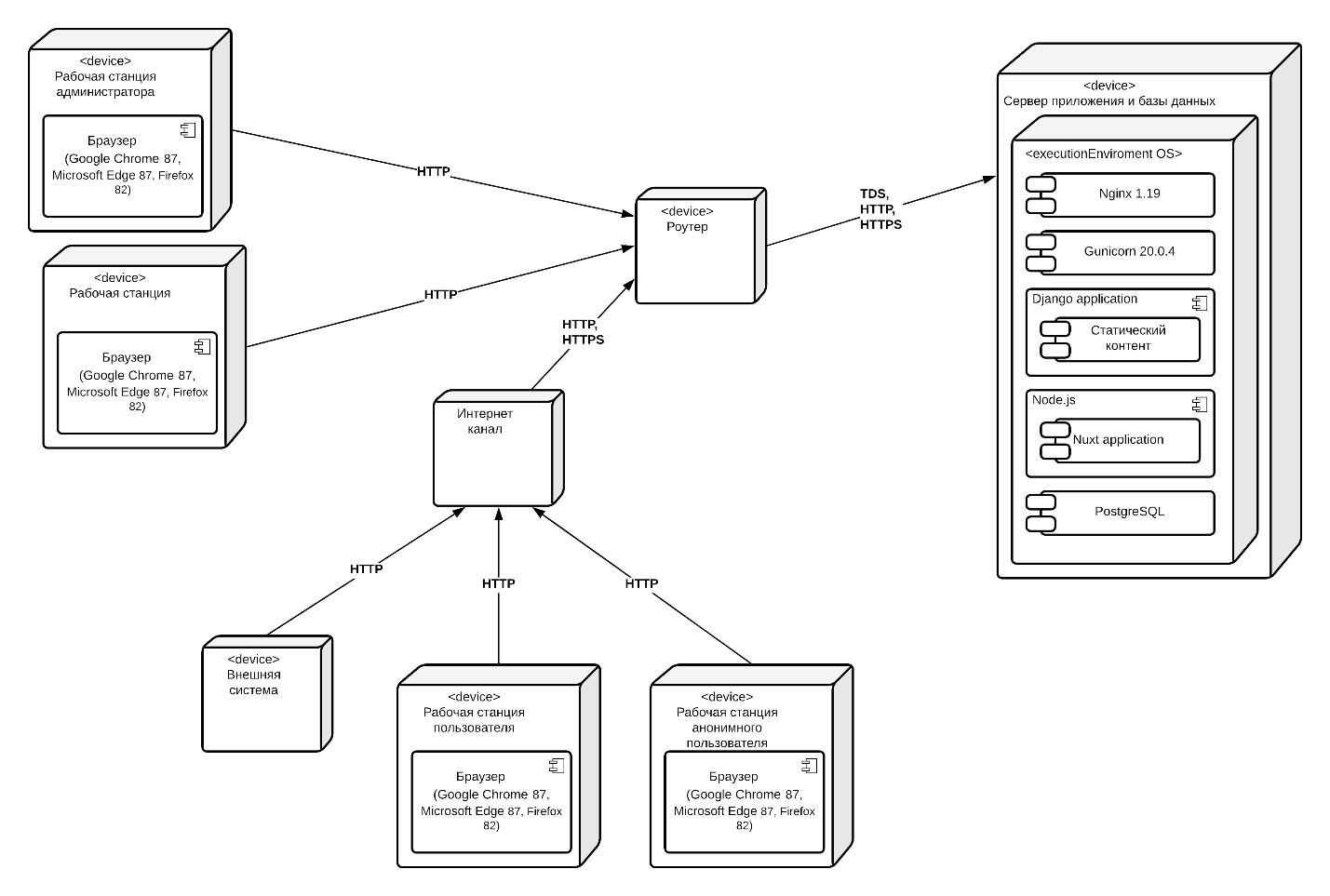


Рисунок 3.1 – Диаграмма развертывания

В качестве хранилища данных необходимых для функционирования системы используется PostgreSQL. Сервер базы данных может быть размещен как на сервере, на котором располагается само приложение, так и на других серверах, доступных в рамках данной сети. Помимо прочего существует возможность использования кластера для обеспечения работы базы данных.

В процессе исследования программных решений в области проведения онлайн-исследований было отмечен факт того, что зачастую клиентская часть приложения реализуется в виде SPA (Single Page Application).

Одностраничное приложение – это веб-приложение или веб-сайт, использующий единственный HTML-документ как оболочку для всех веб-страниц и организующий взаимодействие с пользователем через динамически подгружаемые HTML, CSS, JavaScript [9]. Клиентский модуль проектируемого решения должен быть выполнен в соответствии с этой парадигмой.

Все функции, реализованные в результате разработки приложения разбиты на ряд самостоятельных компонентов. Основные архитектурные компоненты программного средства отображены на рисунке 3.2.

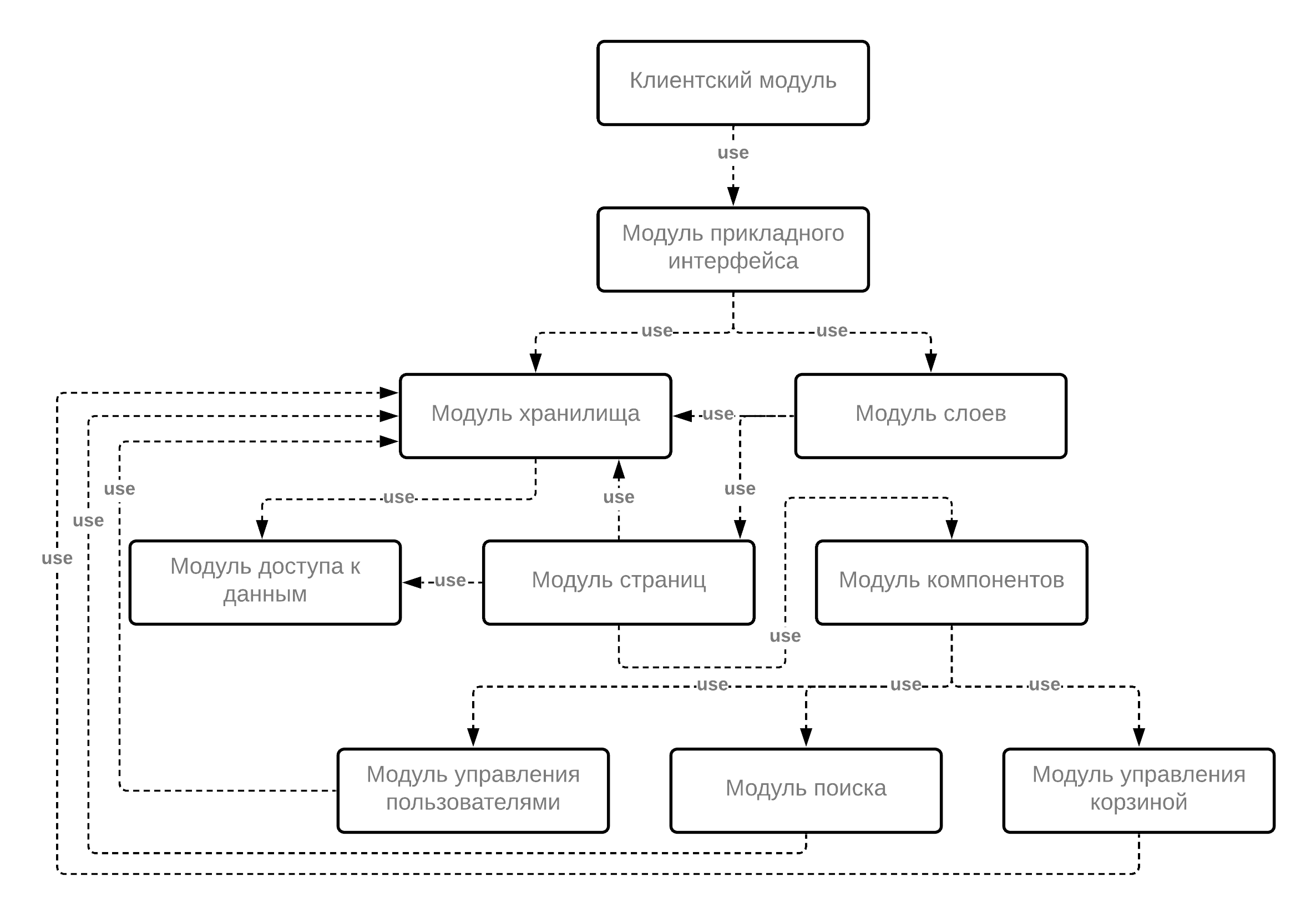


Рисунок 3.2 – Модель архитектуры программного средства

Модулями в данной архитектуре является отдельные компоненты клиенткой части приложения, выполняющие определенную роль.

Модуль слоев. Одна важная вещь, что следует отметить – разделение ответственности – это не то же самое, что разделение на файлы по типу. В современной разработке UI вместо разделения кодовой базы на три огромных слоя, что тесно переплетаются друг с другом, имеет больше смысла делить их на слабо связанные компоненты и компоновать уже их. Внутри компонента, его шаблон, логика и стили неразрывно связаны между собой, что позволяет сделать компонент более сплочённым и удобным в поддержке.

Модуль хранилища. В центре любого Vuex-приложения находится хранилище. «Хранилище» – это контейнер, в котором хранится состояние вашего приложения. Два момента отличают хранилище Vuex от простого глобального объекта: хранилище Vuex реактивно. Когда компоненты Vue полагаются на его состояние, то они будут реактивно и эффективно обновляться, если состояние хранилища изменяется; нельзя напрямую изменять состояние хранилища. Единственный способ внести изменения – явно вызвать мутацию. Это гарантирует, что любое изменение состояния оставляет след и позволяет использовать инструментарий, чтобы лучше понимать ход работы приложения.

Модуль компонентов. Компоненты – это переиспользуемые экземпляры Vue со своим именем.

На рисунке 3.3 представлена схема работы программы в режиме работы администратора. На данной диаграмме наиболее полно отражены функции, реализуемые программным средством.

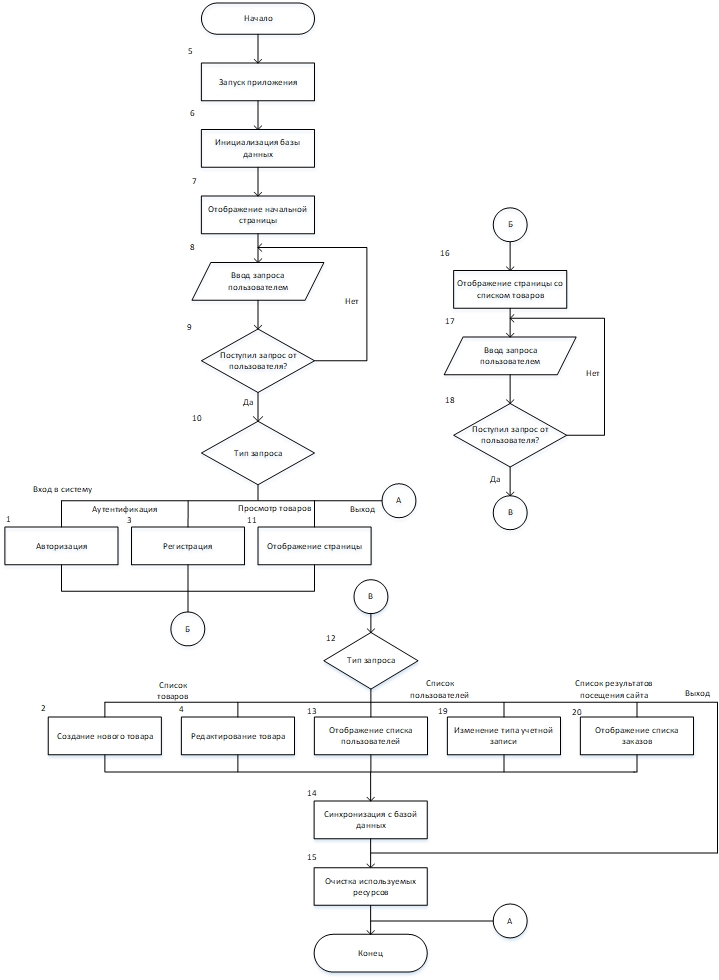


Рисунок 3.3 – Схема работы программы в режиме администратора

## 3.2 Проектирование архитектуры базы данных

Разработанная во второй главе информационная модель нашла свое отражение в физической модели базы данных. При проектировании архитектуры было отмечено, что не все логические сущности необходимы на уровне базы данных. В результате анализа, была проведена денормализация базы данных и как результат, хранение части сущностей осуществляется в полях формата JSON. Модель базы данных приняла вид, изображенный на рисунке 3.4.

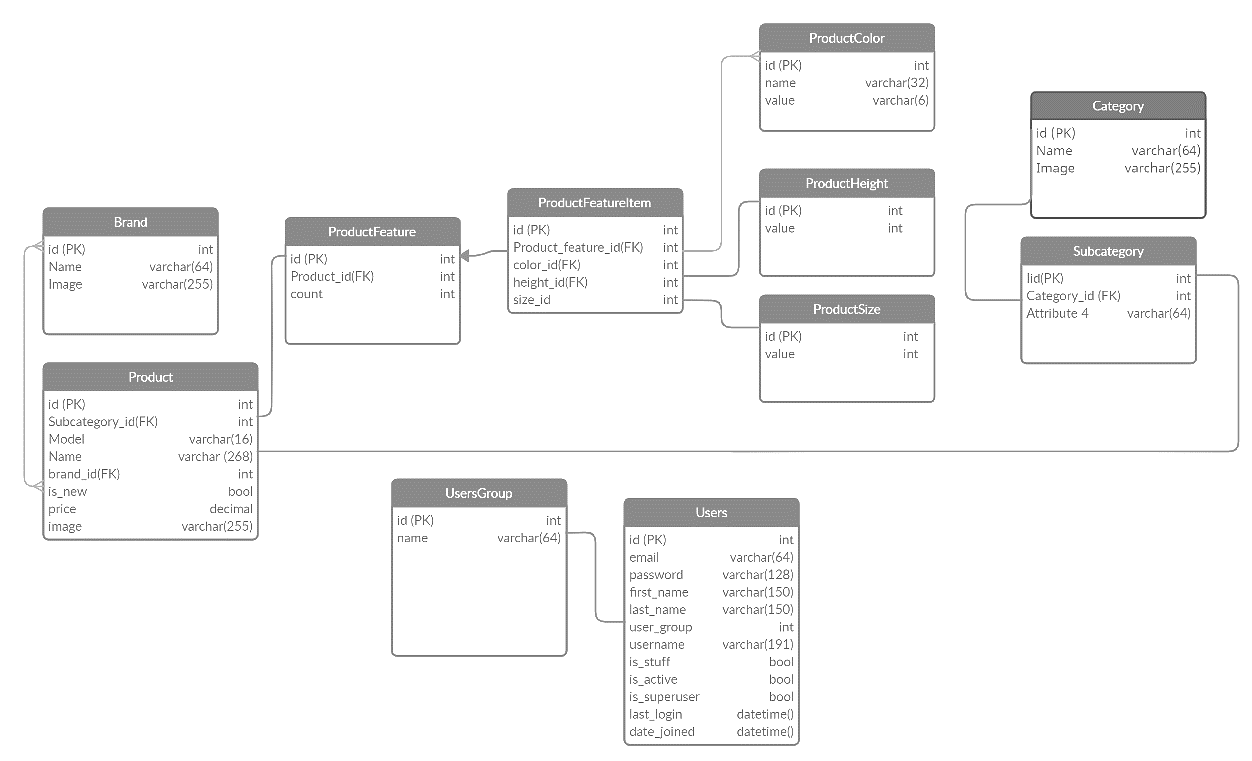


Рисунок 3.4 – Физическая модель базы данных

## 3.3 Проектирование алгоритмов ПС

В процессе проектирования архитектуры был определении детализирован ряд алгоритмов реализации логики функций приложения. Более детально были исследованы функции реализации условной логики для каталога товаров, такие как алгоритм применения фильтров и алгоритм поиска товаров, а также алгоритм регистрации пользователей на сайте.

3.3.1 Алгоритм применения фильтров

Одной из функциональных возможностей проектируемого приложения является возможность задания правил отображения товаров на странице. Схема алгоритма, применяемого при реализации данной функциональности, приведена на рисунке 3.5.

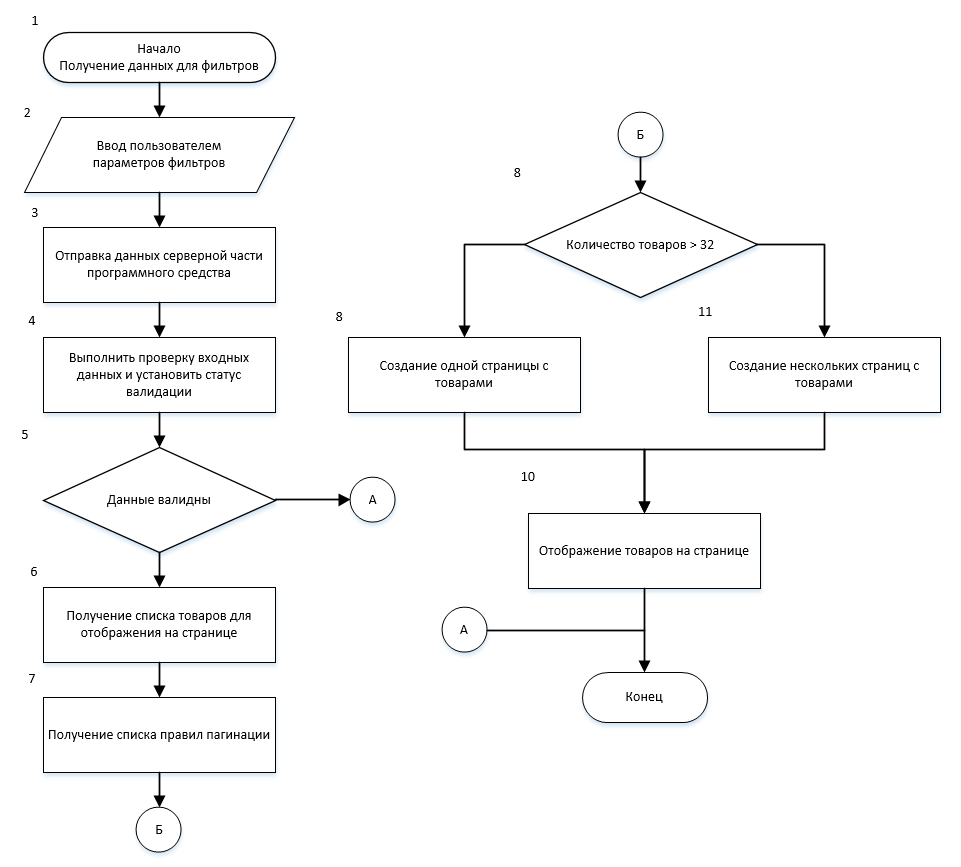


Рисунок 3.5 – Схема алгоритма применения фильтров

Как мы видим из схемы алгоритма, логика работы клиентского модуля основывается на принципах реактивного программирования. После того, как пользователь задаст необходимые ему данные в фильтрах и выполнения заданных проверок корректности введенных данных, выполняется соответствующее действие из вспомогательного класса.

В свою очередь действия вызывают диспетчер, что приводит к обновлению уже отрисованных компонентов. Списковая компонента запрашивает правила отображения для каждого элемента на странице, после чего пробегается по полученному списку правил в поисках такого, условие которого выполняется.

В случае, если найдено более одного правила, система должна должна отобразить каждое найденное правило. Таким образом мы имеем возможность однозначно определить необходимое поведение системы.

В зависимости от типа правила отображения, делается вывод о необходимости отображение данного товара на странице.

В процессе реализации данного алгоритма следует использовать функциональный подход. Использование чистых функций положительно сказывается на реактивных интерфейсах, так как сам подход предполагает отсутствие состояний.

3.3.2 Алгоритм поиска товаров

Еще одной возможностью, схожей с логикой правил отображения является логика поиска товаров, удовлетворяющих критериям пользователя. Принципиальное отличие в данном случае заключается в том, что критерием выбора результата является один из двух параметров: наименование модели и ее номер.

Как и в случае с фильтрами, в случае, если критерий отображения выполнен в более чем одном правиле, будет отображено несколько товаров, правила выбора которых были выполнены.

В ситуации, когда не был выполнен ни один критерий, будет отображаться страница с надписью: «Товары не найдены». Алгоритм поиска товаров приведен на диаграмме 3.6.

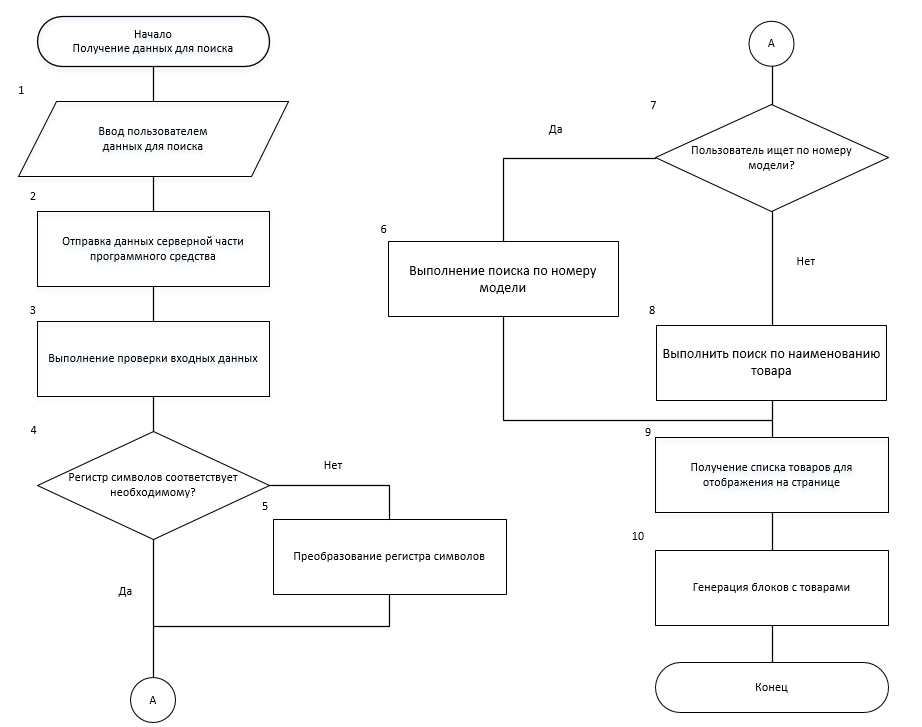


Рисунок 3.6 – Схема алгоритма поиска товаров

# Разработка программного средства

На основе спецификации функциональных требований и спроектированной архитектуры программного средства, а также требований к техническим характеристикам аппаратного обеспечения был произведен выбор наиболее подходящих технологий для разработки программного решения.

## 4.1 Выбор и обоснование языков программирования

Программное средство можно разделить две основные части: серверная часть, обеспечивающая работу с базой данных и реализующая сохранение результатов работы с программным средством, и клиентская часть, реализующая интерфейс для работы с программным средством.

4.1.1 Серверная часть ПС

Серверная часть программного средства реализована с использованием языка программирования Python и фреймворка для веб-приложений Django.

Django – свободный фреймворк для веб-приложений на языке Python, использующий шаблон проектирования MVC [4].

Сайт на Django строится из одного или нескольких приложений, которые рекомендуется делать отчуждаемыми и подключаемыми. Это одно из существенных архитектурных отличий этого фреймворка от некоторых других Один из основных принципов фреймворка – DRY (Don't repeat yourself).

Также, в отличие от других фреймворков, обработчики URL в Django конфигурируются явно при помощи регулярных выражений.

Для работы с базой данных Django использует собственный ORM, в котором модель данных описывается классами Python, и по ней генерируется схема базы данных.

Архитектура Django похожа на «Модель-Представление-Контроллер» (MVC). Контроллер классической модели MVC примерно соответствует уровню, который в Django называется Представление (View), а презентационная логика Представления реализуется в Django уровнем Шаблонов (Template). Из-за этого уровневую архитектуру Django часто называют «Модель-Шаблон-Представление» (MTV).

Первоначальная разработка Django как средства для работы новостных ресурсов достаточно сильно отразилась на его архитектуре: он предоставляет ряд средств, которые помогают в быстрой разработке веб-сайтов информационного характера. Так, например, разработчику не требуется создавать контроллеры и страницы для административной части сайта, в Django есть встроенное приложение для управления содержимым, которое можно включить в любой сайт, сделанный на Django, и которое может управлять сразу несколькими сайтами на одном сервере. Административное приложение позволяет создавать, изменять и удалять любые объекты наполнения сайта, протоколируя все совершённые действия, и предоставляет интерфейс для управления пользователями и группами (с пообъектным назначением прав).

Некоторые возможности Django:

* ORM, API доступа к БД с поддержкой транзакций;
* встроенный интерфейс администратора, с уже имеющимися переводами на многие языки;
* диспетчер URL на основе регулярных выражений;
* расширяемая система шаблонов с тегами и наследованием;
* система кеширования;
* интернационализация;
* подключаемая архитектура приложений, которые можно устанавливать на любые Django-сайты;
* «generic views» – шаблоны функций контроллеров;
* авторизация и аутентификация, подключение внешних модулей аутентификации: LDAP, OpenID и другие;
* система фильтров («middleware») для построения дополнительных обработчиков запросов, как например включённые в дистрибутив фильтры для кеширования, сжатия, нормализации URL и поддержки анонимных сессий;
* библиотека для работы с формами (наследование, построение форм по существующей модели БД);
* встроенная автоматическая документация по тегам шаблонов и моделям данных, доступная через административное приложение.

4.1.2 Клиентская часть ПС

В качестве языка и фреймворка, на которых велась разработка клиентского модуля приложения были выбраны язык программирования JavaScript и Nuxt.js.

Nuxt.js – это фреймворк, позволяющий создавать универсальные веб-приложения на Vue.js с использованием Node.js. С помощью него можно рендерить UI на сервере и генерировать статические сайты.

Основные принципы и возможности фреймворка Nuxt:

* Серверные страницы предоставляют полную HTML-страницу, уже готовую для рендеринга, а затем страница «гидрируется» с помощью JavaScript в клиенте для добавления интерактивности и превращается в одностраничное приложение (SPA). Это означает, что приложения Nuxt представляют собой отдельные приложения JavaScript, которые загружают данные с использованием интерфейса API, а не внедряются в другие серверные фреймворки приложений или накладываются на статический HTML.
* Предварительная загрузка асинхронных данных. Чтобы реализовать на сервере преимущества предварительно отображаемых страниц, вам необходимо обеспечить, чтобы рендерер на сервере имел все необходимые данные перед началом рендеринга. Это просто для статических страниц, но для динамических приложений, зависящих от вызовов API, вам необходимо обеспечить, чтобы все критические данные извлекались до того, как страница будет отображена и отправлена с сервера. Даже для чистых SPA-приложений полезно иметь хуки, которые позволят указать, какие данные необходимы, прежде чем страница будет визуализирована, а что может быть заполнено позже.
* Разбор страниц на макеты, страницы и компоненты. Одна из лучших особенностей архитектуры на основе компонентов заключается в том, что все можно рассматривать как компонент. Однако при переходе в систему, использующую для создания отдельных страниц маршрутизацию, полезно добавить еще некоторую структуру поверх этой абстракции. Nuxt позволяет делать это, используя понятия страниц и макетов. Страница соответствует маршруту и, естественно, тому, как мы привыкли думать о сети. Каждая страница может иметь макет, который отображается внутри нее, поэтому макеты становятся способом создания общей структуры на разных страницах.
* Организация структуры файлов по ролям. Один из первых вопросов в любом виде приложений – как организовать файлы. Nuxt использует относительно простой подход, сохраняя при этом структуру. Файлы разделены в соответствии с ролями, с каталогами для components, layouts, pages, middleware, plugins, store, компилируемых assets и полностью static ресурсов.

## 4.2 Выбор среды разработки

В качестве среды разработки для написания исходного кода программного средства была выбрана Visual Studio Code.

Visual Studio Code – редактор исходного кода, разработанный Microsoft для Windows, Linux и macOS. Позиционируется как «лёгкий» редактор кода для кроссплатформенной разработки веб- и облачных приложений. Включает в себя отладчик, инструменты для работы с Git, подсветку синтаксиса, IntelliSense и средства для рефакторинга. Имеет широкие возможности для кастомизации: пользовательские темы, сочетания клавиш и файлы конфигурации. Распространяется бесплатно, разрабатывается как программное обеспечение с открытым исходным кодом, но готовые сборки распространяются под проприетарной лицензией.

Палитра команд представляет собой подобие командной строки, которая вызывается сочетанием клавиш.

Visual Studio также позволяет заменять кодовую страницу при сохранении документа, символы перевода строки и язык программирования текущего документа.

С 2018 года появилось расширение Python для Visual Studio Code с открытым исходным кодом, которое предоставляет разработчикам широкие возможности для редактирования, отладки и тестирования кода.

## Диаграммы классов программного средства

Для наглядной демонстрации всех классов, разработанных в процессе работы над серверной частью программного средства, был построен ряд диаграмм классов, которые, в свою очередь подразделены на определенные модули, каждый из которых отвечает за логику описываемого компонента программного средства.

Для удобства описания диаграммы классов ее можно подразделить на три условные части: классы, отвечающие за логику работы с товарами, логику разделения ролей и авторизации, а также классы, реализующие работу с сессиями.

Первая рассматриваемая часть – модули product и reference, отвечающие за логику работы с товарами. Классы и связи данного модуля изображены на рисунке 4.1.

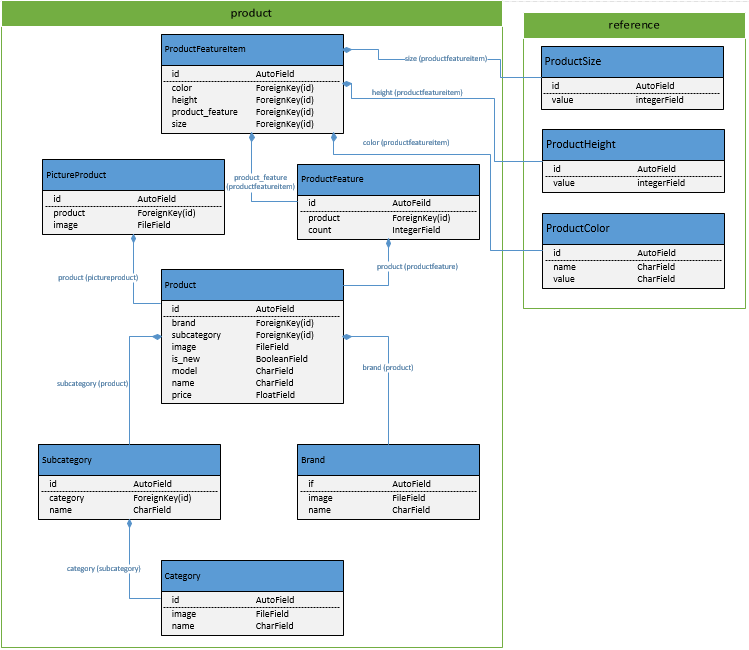


Рисунок 4.1 – Диаграмма классов модуля работы с товарами

На данной модели перечислены все классы, которые используются для реализации логики работы с товарами. Большая часть классов представляет собой типы данных описывающих модели предметной области, а также перечисления, которые используются, для реализации логики в предметной области приложения.

Основным классом данного модуля является product, который хранит информацию о товарах, отображаемых в каталоге. Вспомогательными классы, описывающими детализацию продуктов являются subcategory и category, определяющие категорию и подкатегорию заданного товара, brand, отвечающий за торговую марку товара, PictureProduct, - изображения товара, ProductFeature – количество имеющихся единиц товара.

Следующей рассматриваемой частью являются модули администрирования и авторизации (рисунок 4.2).

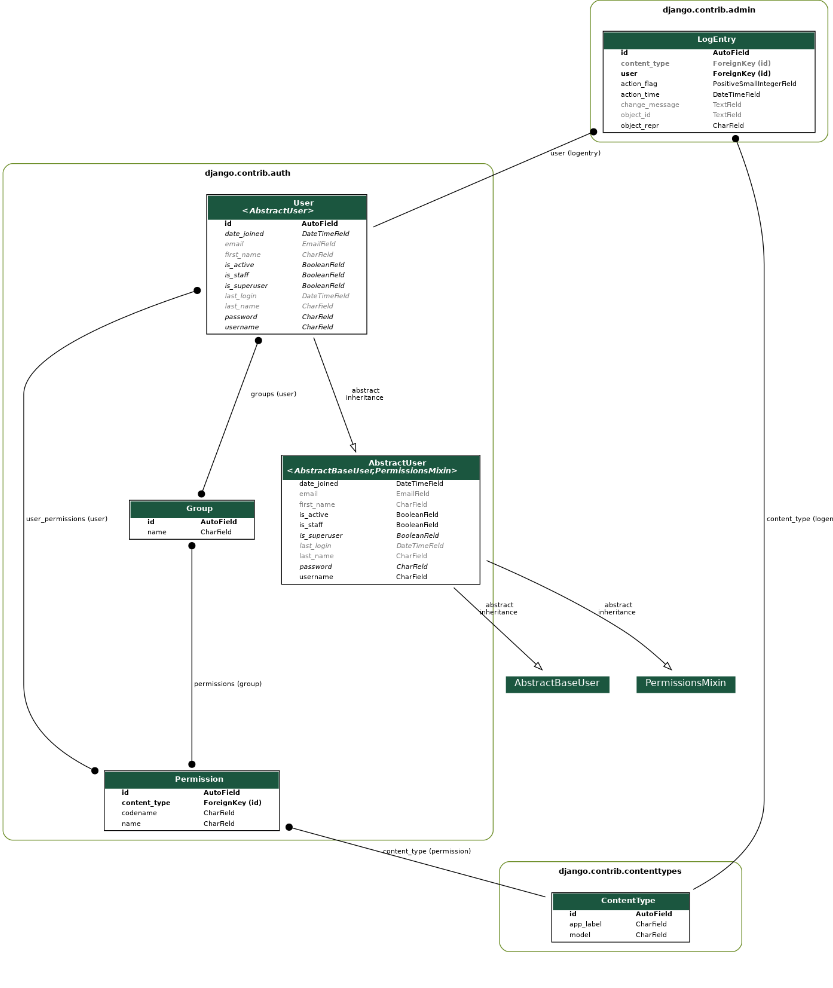


Рисунок 4.2 – Диаграмма классов авторизации и администрирования

На рисунке изображены классы, реализующие авторизацию пользователя в системе, а также разделения ролей между пользователями. Логика авторизации реализована с использованием паттерна посредник. Данный подход позволяет использовать разработанную компоненту при реализации аналогичного модуля в любом другом приложении. Изображенные на диаграмме класс осуществляют проверку корректности ключевого значения, а также генерацию токена при входе пользователя в систему.

Заключающим модулем является группа классов, реализующая работу приложения с сессиями пользователей. Данная группа классов изображена на рисунке 4.3.

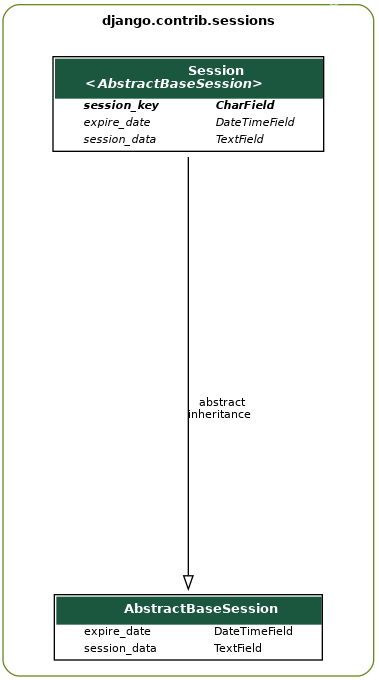


Рисунок 4.3 – Диаграмма классов логики сессий

Изображенный на диаграмме класс осуществляет логику работы программного средства с сессиями пользователей. Django полностью поддерживает сессии для анонимных пользователей, позволяет сохранять и получать данные для каждого посетителя сайта. Механизм сессии сохраняет данные на сервере и самостоятельно управляет сессионными cookies.

## Описание компонентов клиентской части программного продукта

Для реализации клиентской части на Nuxt все элементы управления были разбиты на отдельные компоненты (модули). Каждая отдельная компонента предназначена для реализации отображения одного атомарного элемента.

Некоторые разработанные компоненты представляют собой сложные и используют другие. Такие высокоуровневые компоненты реализуют целые страницы или формы приложения.

Все разработанные компоненты можно отнести к одной их следующих групп: компоненты страниц, компоненты хранилища, компоненты слоев и вспомогательные компоненты.

Каждая описанная компонента содержит ряд методов, описывающих ее работу. Среди них можно выделить следующие: getInitialState(), ComponentDidMount(), componentWillMount(), ComponentWillUnmount(), render(), getProps().

Метод getInitialState() содержит программный код, который осуществляет инициализацию состояния компоненты перед первым отображением компоненты на странице. Данный метод вызывается один раз и позволяет задать начальные условия отображения, или же осуществить подготовку данных.

Метод componentWillMount() также вызывается один раз перед внедрением компоненты в дерево узлов на веб-странице. В данном методе как правило осуществляется подписка на события, по которым будет осуществляться дальнейшая перерисовка компоненты. Как правило в данном методе осуществляется подписка на обновление значений в хранилище данных.

Единственным обязательным для каждой компоненты методом является метод render(). В данном методе осуществляется генерация кода HTML, который будет отображаться в месте подключения генерируемой компоненты. Внутри данного метода могут использоваться как другие пользовательские компоненты, так и стандартные для Nuxt компоненты.

Передача параметров между компонентами осуществляется через использование свойств. Согласно лучшим практикам написания кода, данные свойства должны быть описаны в методе getProps(). Данный метод возвращает список свойств, поддерживаемых данным компонентом.

# Тестирование программного средства

Для проверки правильности работы программного обеспечения и реализации функциональных требований проведено тестирование модулей разработанного программного обеспечения.

На ранних этапах тестирование проводилось при помощи юнит-тестирования. Юнит-тесты позволяли еще на начальном этапе разработки выявлять ошибки в программном средстве, а также позволяли более полно и детально проверить самые основные и важные модули программного средства.

На более поздних этапах в качестве метода тестирования программного обеспечения выбран метод функционального тестирования.

Функциональное тестирование является основным видом тестирования программного обеспечения. Каждая функция программы тестируется и при этом делается вывод об ее правильности. Очевидно, что по всей области определения проверить функцию невозможно и поэтому каждая функция проверяется на правильность в некоторых точках области её определения.

Данный метод тестирования позволяет:

− обнаружить некорректные или отсутствующие функции;

− обнаружить ошибки интерфейса;

− обнаружить ошибки во внешних структурах данных (файлы, базы данных).

В рамках разработки дипломного проекта функциональное тестирование реализовано с помощью набора тест-кейсов, собранного в тестовый сценарий, представленный в таблице 5.1.

Таблица 5.1 – Результаты тестирования приложения

| № | Тестовый случай | Ожидаемый результат | Фактический результат |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 1 | 1)Открыть приложение.  2)Нажать на кнопку «Регистрация». | Форма регистрации нового пользователя. | Совпадает с ожидаемым. |
| 2 | 1)Открыть приложение.  2)Ввести данные учетной записи администратора.  3)Нажать кнопку Login. | Пользователь успешно авторизован, открыта главная страница программного средства. | Совпадает с ожидаемым. |
| 3 | 1)Отрыть приложение повторно после авторизации в системе, в течение 5 минут с момента первой авторизации. | Пользователь попрежнему авторизован в системе | Совпадает с ожидаемым. |
| 4 | 1)Открыть приложение.  2)Ввести данные учетной записи администратора.  3)Нажать кнопку «Войти».  4)Нажать кнопку «Выйти». | Переход на главную страницу. | Совпадает с ожидаемым. |
| 5 | 1)Перейти в защищенную область приложения по прямому адресу. | Переход на страницу No permission | Совпадает с ожидаемым. |
| 6 | 1)Открыть приложение.  2)Нажать на кнопку «Регистрация».  3)Заполнить поля формы  4)Нажать кнопку «Регистрация» | Произошла регистрация пользователя. Появление окна авторизации. | Совпадает с ожидаемым. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

В результате итогового тестирования серьезных дефектов выявлено не было. Все тестовые случаи были пройдены успешно. Фактический результат совпал с ожидаемым результатом в каждом из тестовых случаев.

# Руководство пользователя

Для корректной работы программного средства предъявляются следующие минимальные требования к техническим характеристикам сервера:

* операционная система: Linux (Ubuntu 18);
* веб-сервер и прокси сервер nginx;
* HTTP-сервер интерфейса шлюза веб-сервера Python – Gunicorn;
* процессор Xeon 2.2 Ггц или более быстродействующий;
* оперативная память 4 Гбайт или более;
* сетевая карта Ethernet 1 Гбит.
* доступный объем дискового пространства 10 Гбайт.

Дополнительно необходимо наличие развернутой системы управления базами данных PostgreSQL, пакетный менеджер npm.

## 6.1 Развертывание сервера приложения

Надо ли?

## 6.2 Использование программного продукта

6.2.1 Авторизация пользователя

При обращении к системе пользователя не имеющего аккаунта в системе или не прошедшего авторизацию, ему будет отображена страница входа в систему (рисунок 6.1).

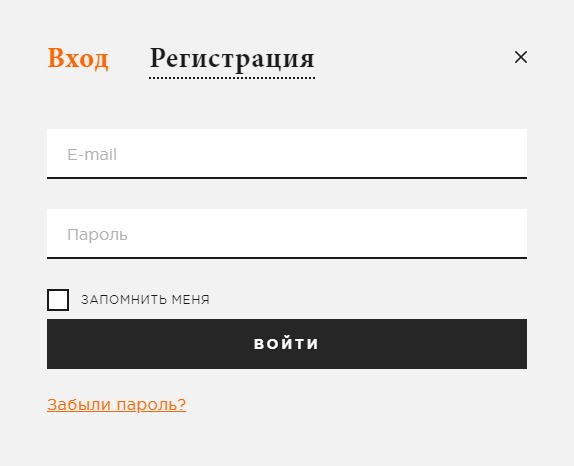


Рисунок 6.1 – Форма авторизации пользователя в системе

Для входа пользователя в систему необходимо указать адрес электронной почты и пароль, введенные при регистрации. Задав переключателю «Запомнить меня» активное положение, процесс выхода из учетной записи будет осуществлен спустя значительное количество времени.

6.2.2 Регистрация в системе

В случае, если новый пользователь, не имеющий своей учетной записи, желает создать собственный аккаунт, ему необходимо зарегистрироваться в системе, нажав кнопку «Регистрация». Окно регистрации нового пользователя представлена на рисунке 6.2.

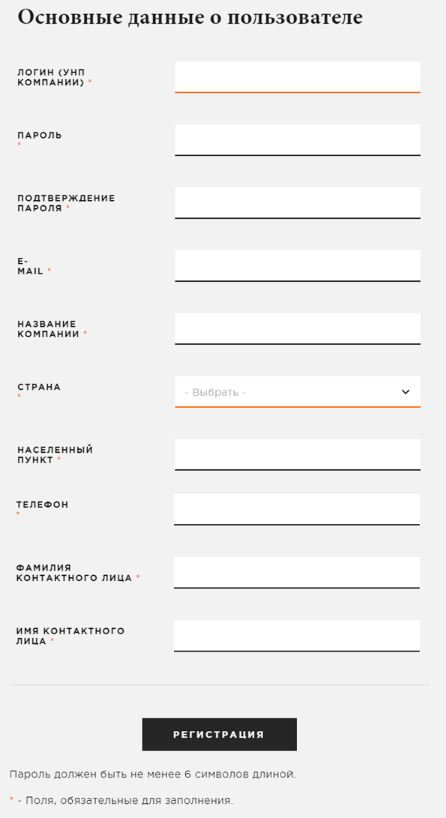


Рисунок 6.2 – Форма регистрации пользователя в системе

В процессе развертывания системы создается учетная запись администратора, для того чтобы иметь возможность сконфигурировать систему с ее использованием. Параметры учетной записи администратора могут быть сконфигурированы в файле конфигурации приложения.

Для регистрации в системе, необходимо ввести адрес электронной почты, пароль, имя и фамилию пользователя, а также дополнительные опциональные данные. По нажатию на кнопку «Регистрация» будет осуществлено создание новой учетной записи. В случае успешного завершения процессе будет осуществлен переход на страницу авторизации в системе.

6.2.3 Управление каталогом

При успешном прохождении авторизации администратора в системе будет отображена главная страница приложения. На данной странице расположены ссылки на основные части приложения. Пример главной страницы администратора отображен на рисунке 6.3.

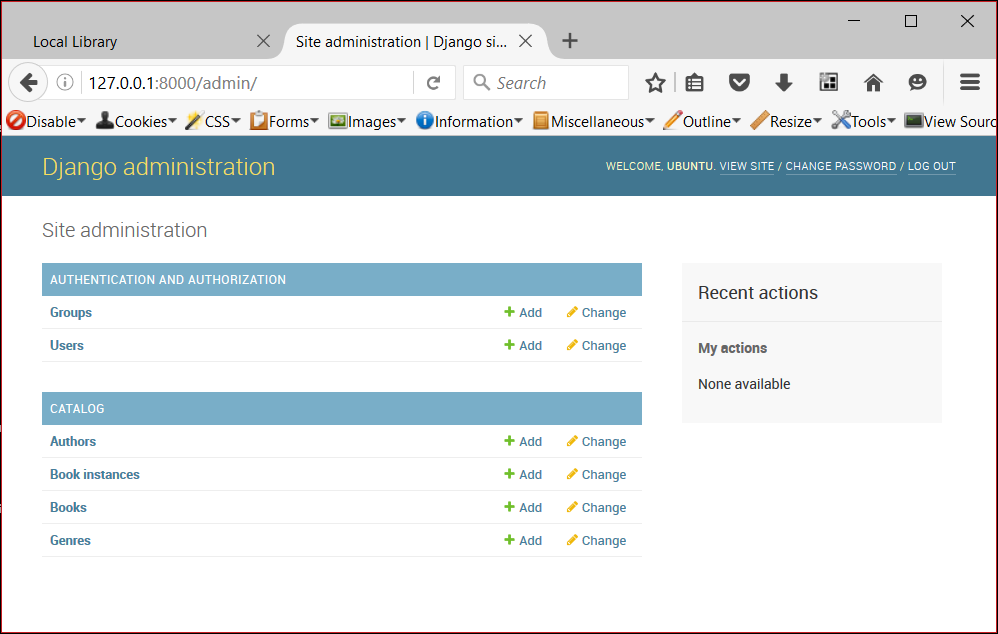


Рисунок 6.3 – Главная страница администратора

6.2.4 Управление пользователями

Нажатие на кнопку «Users» приводит к переходу на страницу со списковой формой пользователей. Данная форма содержит перечисление, всех пользователей, зарегистрированных в системе. Для каждого из существующих пользователей отображаются его адрес электронной почты, фамилия и имя, а также тип учетной записи.

Администратор может осуществить наделение любого их этик пользователей правами администратора, нажав на кнопку «Activate». На этой же форме возможно и осуществление обратного текущему действия. Внешний вид формы доступен на рисунке 6.4.

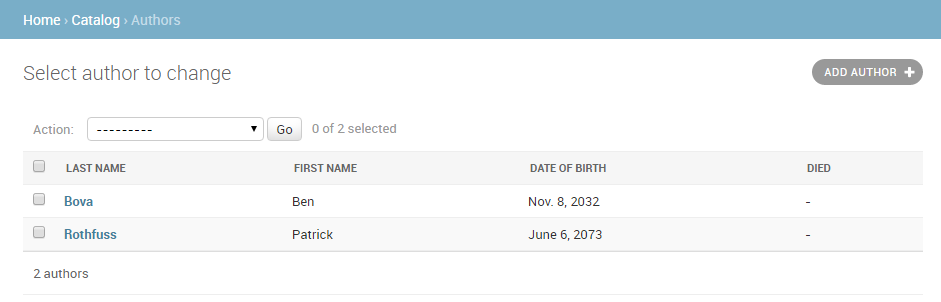


Рисунок 6.4 – Списковая форма пользователей в системе.

6.2.5 Просмотр каталога

При успешном прохождении авторизации пользователем в системе ему будет отображена главная страница. На данной странице расположены основные части, такие как фильтры, расположенные слева от основного блока с товарами, виды сортировки, отображенным над товарами, категориями каталога, находящиеся сверху страницы и другие. Пример главной страницы отображен на рисунке 6.3.

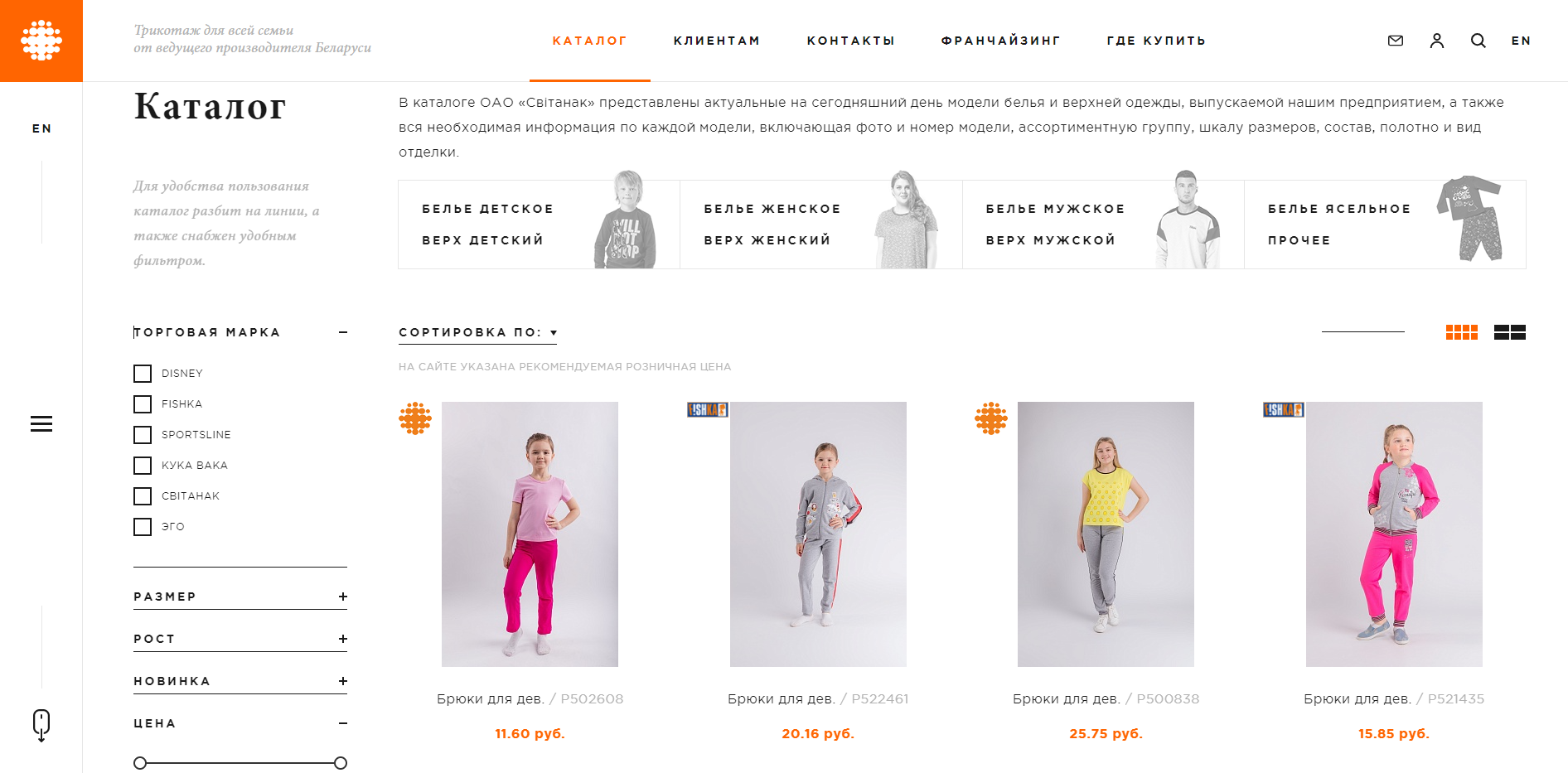


Рисунок 6.5 – Просмотр каталога на главной странице

6.2.6 Применение фильтров для управления каталогом

Для обеспечения удобства использования программного средства были реализованы возможности сортировки, фильтрации и поиска товаров. Для использования фильтров необходимо из всех допустимых вариантов, ограничивающихся торговой маркой (брендом), размером, ростом, ценой, выбрать интересующие пользователя параметры и дождаться обновления ассортимента товаров.

Сортировка реализована по трем параметрам, это: наименование товара, номер модели и цена. Выбор определенного вида сортировки влечет изменение отображение товаров в порядке «от меньшего к большему».

Также существует два вида отображения блока с товарами – сокращенный и подробный вид. Сокращенный вариант отображения товаров изображен на рисунке 6.6, а подробный вид – на рисунке 6.7.

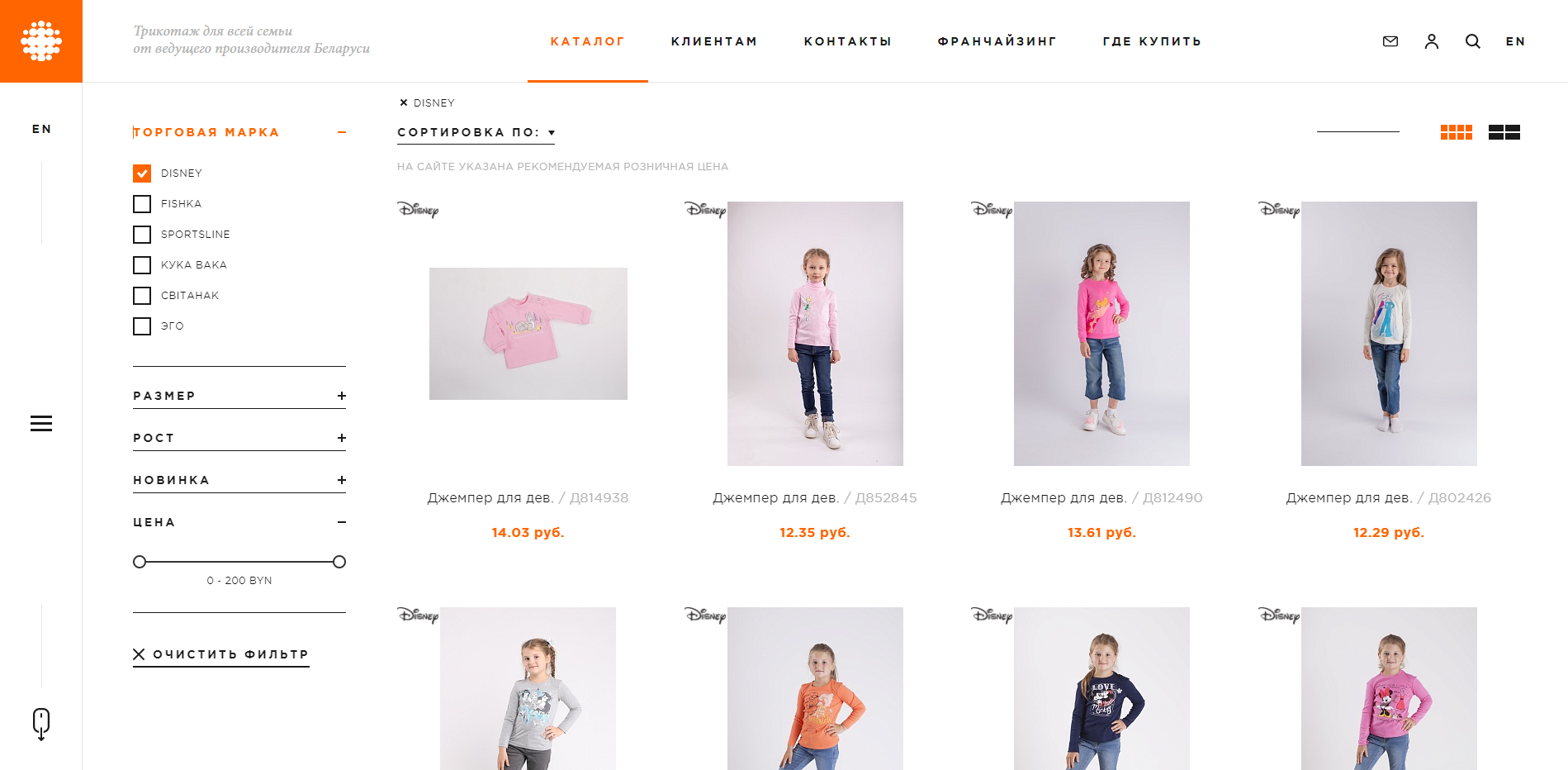


Рисунок 6.6 – Просмотр каталога с использованием фильтров в сокращенном виде

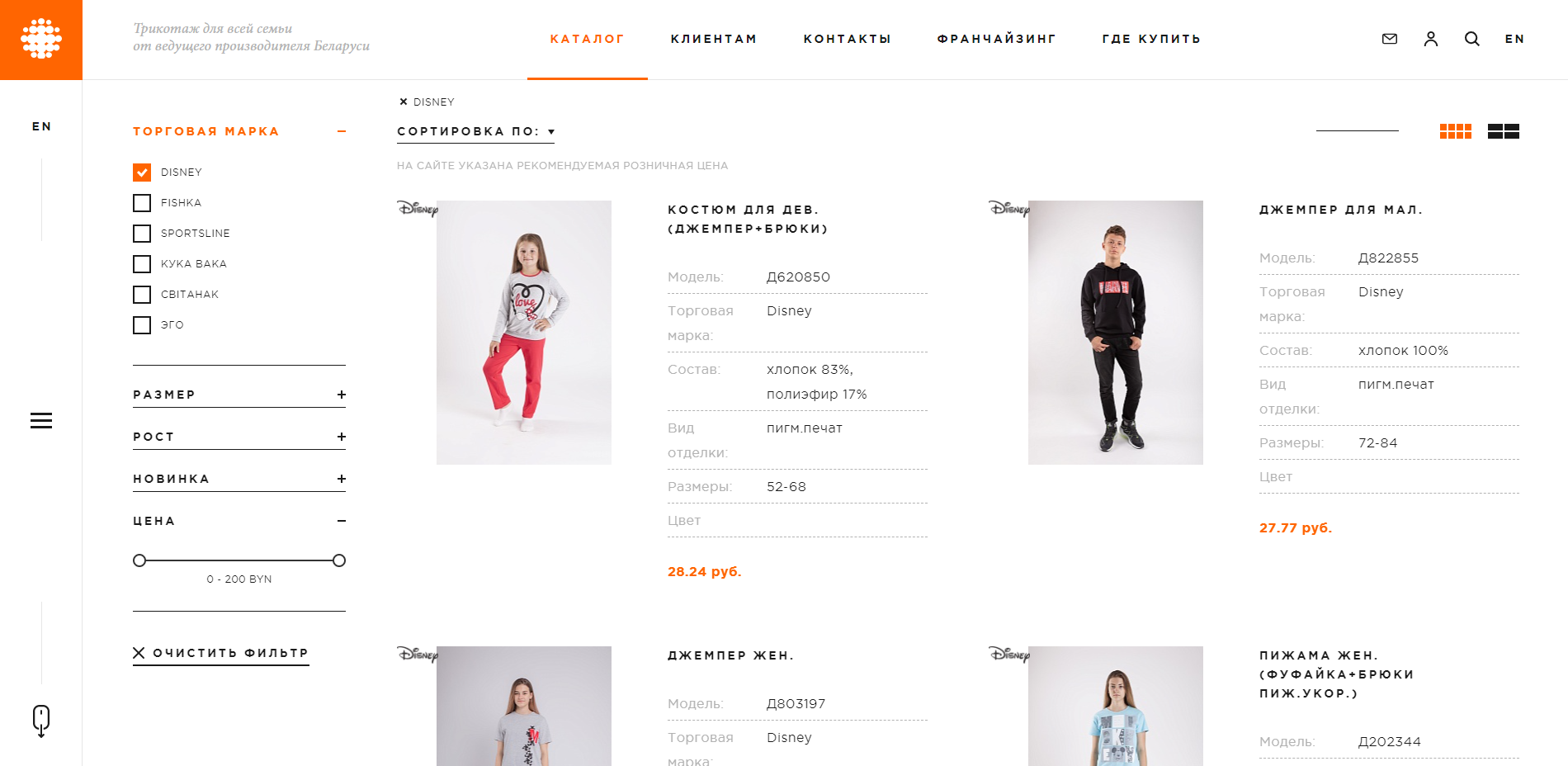


Рисунок 6.7 – Просмотр каталога с использованием фильтров в подробном виде

6.2.7 Поиск товаров

Еще одном фактором, удовлетворяющим требованиям к разрабатываемому программному средству является поиск товаров. Для использования поиска пользователю необходимо нажать на кнопку со значком «лупа», находящуюся в верхнем правом углу страницы и задать для запроса либо наименование товара, либо же номер модели. Поиск срабатывает при указании двух и более символов. Результат поиска товаров отображен на рисунке 6.8.

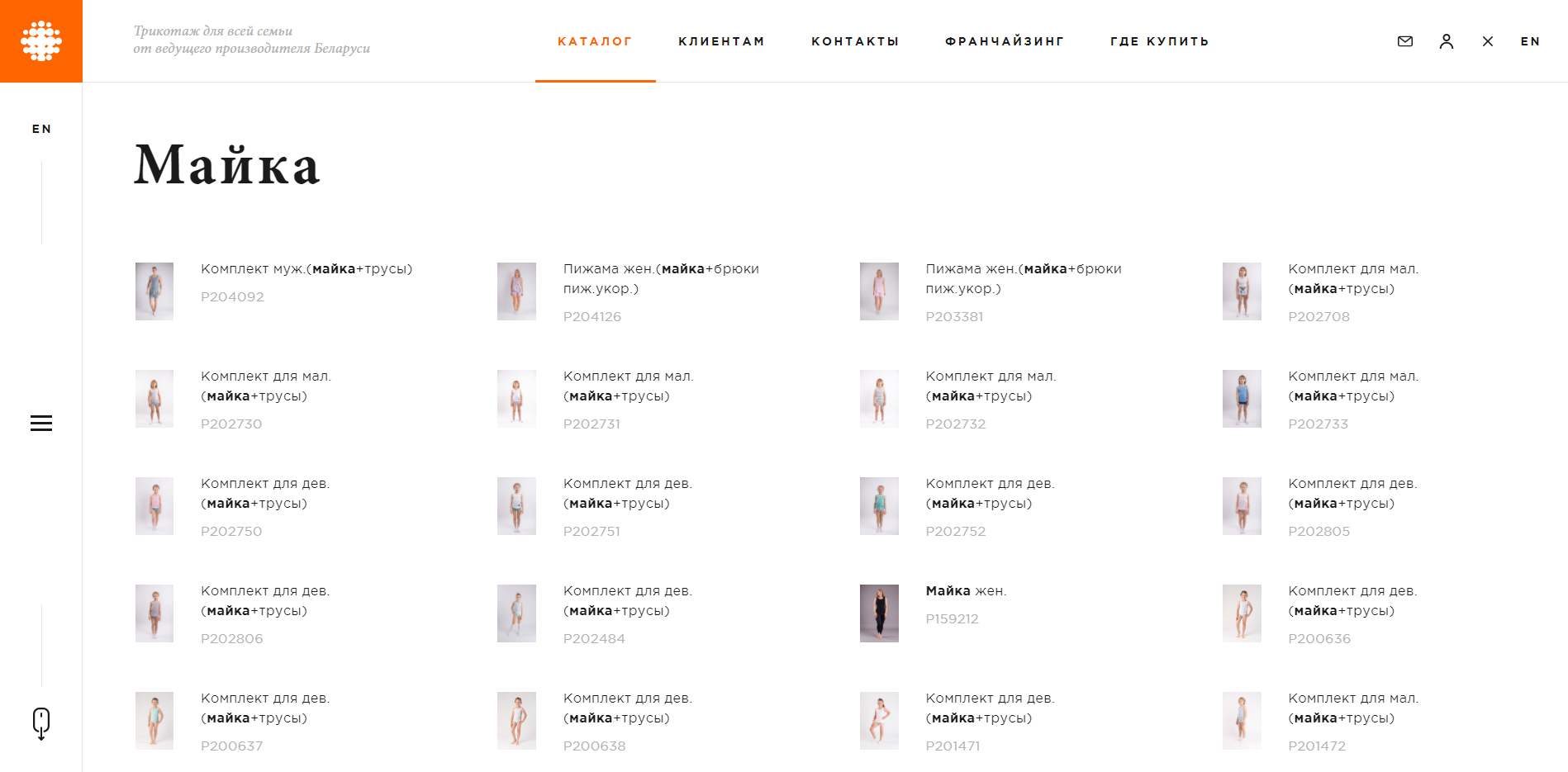


Рисунок 6.8 – Поиск товара

6.2.8 Просмотр добавленных в корзину товаров

Наиболее важной функциональной возможностью, отличающей интернет-магазин от интернет-витрин, является возможность формировать виртуальную корзину и оформлять заказ товаров, находящихся в ней. Пример добавления товаров в корзину изображен на рисунке 6.9.

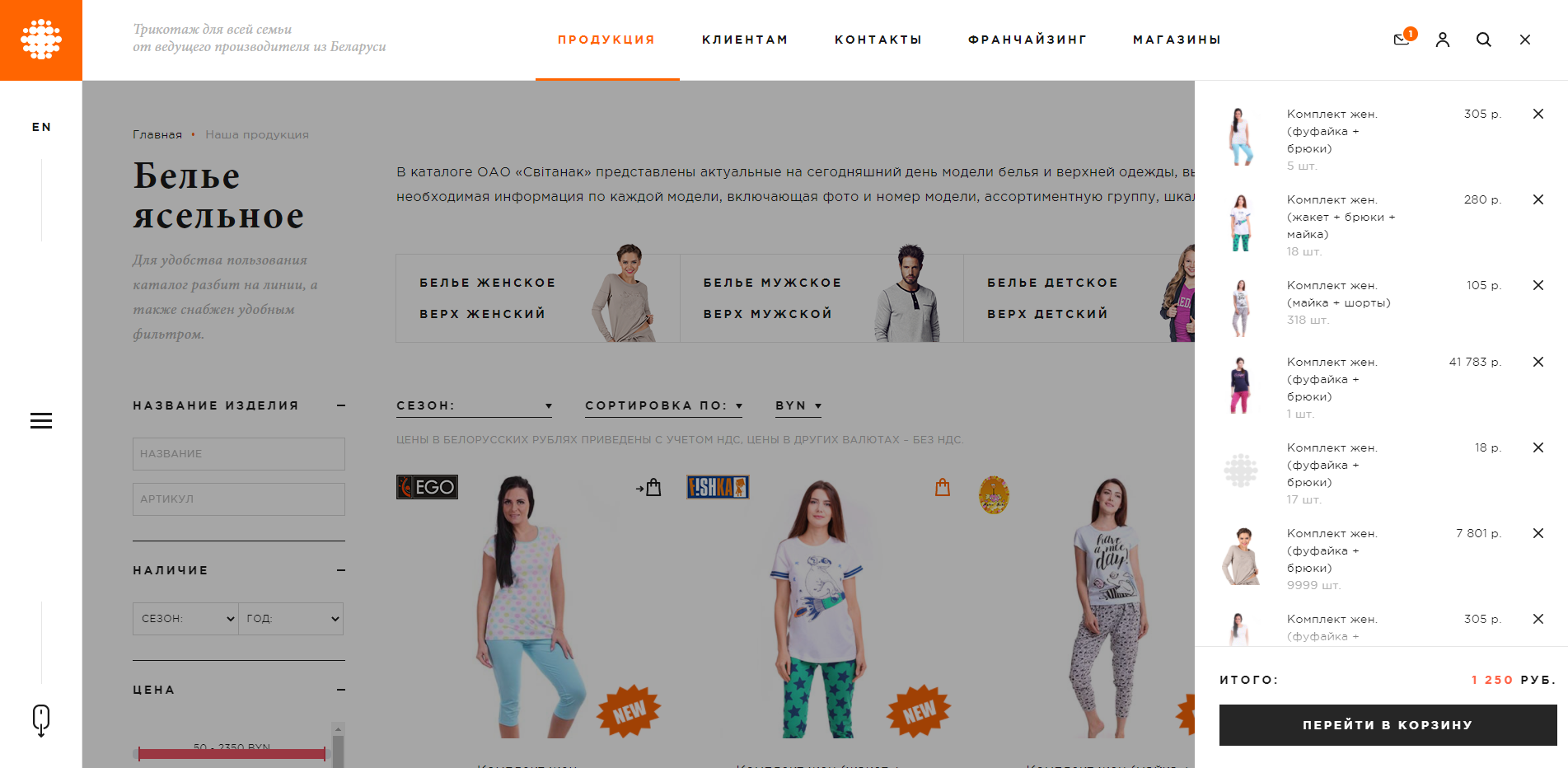


Рисунок 6.9 – Просмотр корзины

6.2.9 Оформление заказа

Следующий шаг, ведущий пользователя к приобретению товаров это оформление заказа. Для совершения данного действия необходимо после предпросмотра товаров в корзине нажать кнопку «Перейти в корзину». После загрузки страницы пользователю отображена детализация товаров, выбранных на предыдущих этапах, это: наименование, модель, размер, рост, количество товаров, цвет, стоимость одного товара, итоговая стоимость всех товаров в корзине. Содержание данной страницы изображено на рисунке 6.10.

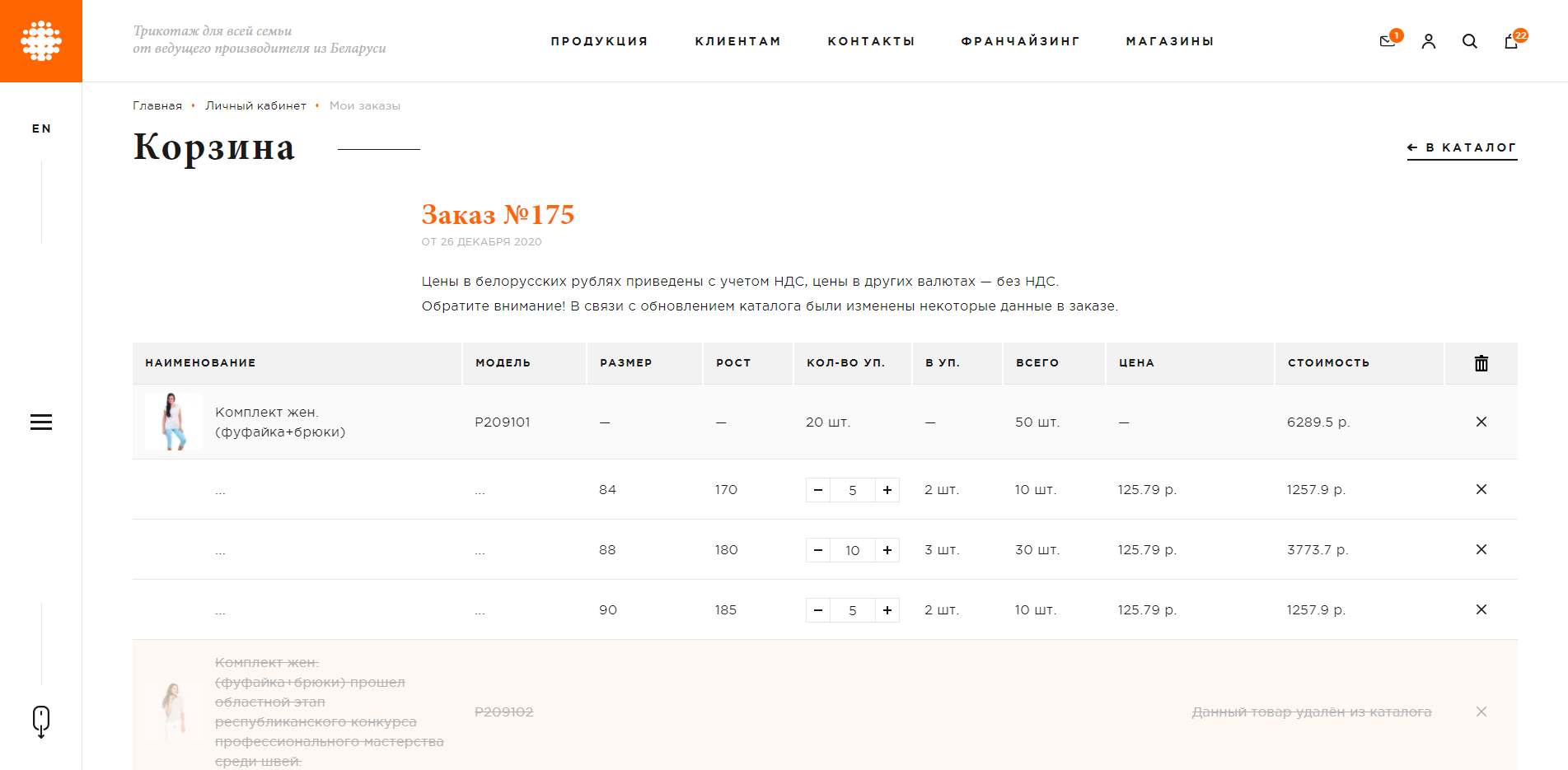


Рисунок 6.10 – Оформление заказа

6.2.10 Подтверждение заказа

Завершающим этапом заказа товаров является непосредственное оформление заказа. Для совершения данного действия пользователю необходимо указать личные данные для того, чтобы в последующем менеджер компании смог связаться с покупателем. В случае, если пользователь при регистрации указал все необходимые для оформления заказа данные, повторное указании не требуется, данные будут загружены автоматически. Также пользователю предоставляется возможность указать комментарий к заказу. Пример приведен на рисунке 6.11.

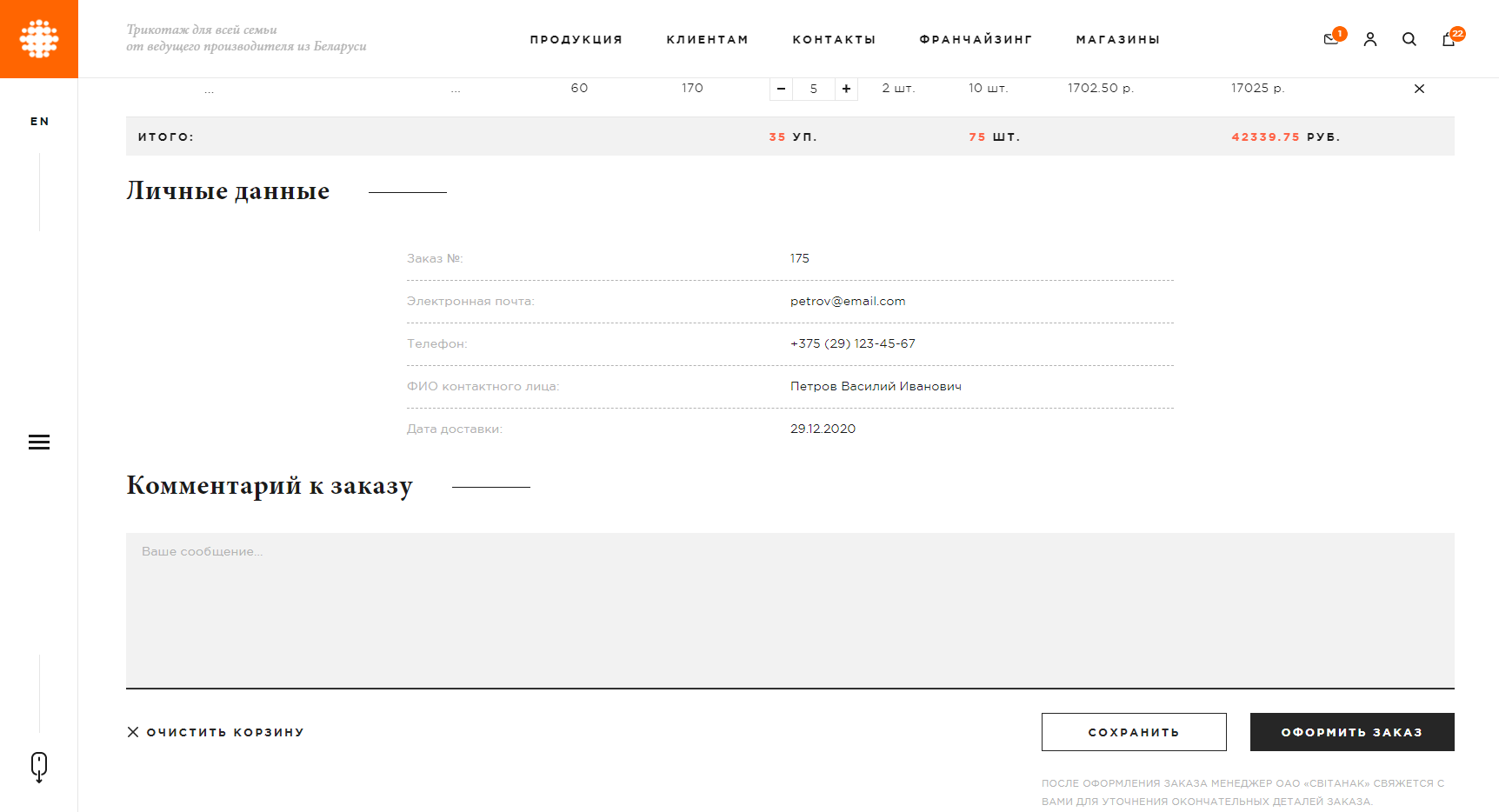


Рисунок 6.11 – Подтверждение заказа

# Технико-экономическое обоснование разработки интернет-магазина для ОАО «Свiтанак»

## 7.1 Описание функций, назначения и потенциальных пользователей программного продукта

Разрабатываемое в дипломном проекте программное средство предоставляет пользователю возможность проведения покупок в режиме онлайн. Также данной программное средство предназначено упростить процессы заказа продукции, которые на текущий момент зачастую выполняются вручную, тем самым освобождая часть рабочего времени работников магазинов и менеджеров по продажам. Пользователем программного средства может выступать любой человек, который имеет персональный компьютер либо мобильное устройство с доступом к сети Интернет.

Данное программное средство разрабатывается собственными силами предприятия для собственных нужд с целью повышения эффективности продаж и снижения временных затрат сотрудников на обработку заказов.

Программное средство будет являться дополнительной площадкой для ведения коммерческой деятельности организации ОАО «Свiтанак».

## 7.2 Расчет затрат на разработку программного средства

Задачей данного раздела дипломного проекта является подтверждение актуальности и экономической целесообразности разработки программного средства и его использования потенциальными пользователями. Раздел включает в себя следующие пункты:

* затраты на основную заработную плату разработчиков;
* затраты на дополнительную заработную плату разработчиков;
* отчисления на социальные нужды;
* прочие затраты (амортизация оборудования, расходы на электроэнергию, командировочные расходы, накладные расходы и т.п.)

7.2.1 Расчет затрат на основную заработную плату разработчиков

Затраты на основную заработную плату определяются составом команды, которая занимается разработкой программного средства, месячным окладом специалистов и трудоемкостью процесса разработки и рассчитываются по формуле:

где n – количество исполнителей, занятых разработкой конкретного ПО;

ЗЧi – часовая заработная плата i-го исполнителя (руб.);

ti – трудоемкость работ, выполняемых i-м исполнителем (ч).

Для разработки данного программного продукта была выбрана команда разработчиков в составе бизнес-аналитика, системного архитектора, тестировщика, дизайнера и двух инженеров-программистов, один из которых является ведущим. Причиной этого является то, что конечный продукт должен состоять из двух частей: клиентской части, непосредственно с которой будут работать пользователи, а также серверной части, на которой осуществляется долговременное хранение результатов работы с программным средством и обработка информации. Весьма целесообразной является параллельная разработка обеих частей программного средства, которой можно добиться, поручив клиентскую часть инженеру-программисту, а вторую часть отдать на выполнение ведущему инженеру-программисту, являющемуся специалисту в области разработки серверных решений. Такое разделение позволит закончить проект вовремя с учетом рисков, связанных с разработкой, и выполнить его качественнее благодаря специализации разработчиков.

Расчетная норма рабочего времени принята равной 168 часам. Данные по заработной плате команды разработчиков предоставлены ОАО «Свiтанак» на 13 ноября 2020 года.

Расчет затрат на основную заработную плату осуществлен в форме таблицы 7.1.

Таблица 7.1 – Расчет затрат на основную заработную плату команды разработчиков.

| № п/п | Участник команды | Вид выполняемой работы | Месячная заработная плата, руб | Часовая заработная пла-та, руб | Трудоем-кость работ, ч | Зарплата по тарифу, руб |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 1 | Бизнес-аналитик | Анализ потребностей клиента с их последующим улучшением, написание требований для ПС и их спецификаций. | 1 890 | 11,25 | 16 | 180 |
| 2 | Систем-ный архитектор | Разработка и подготовка документации, описание сущностей, взаимосвязей и процессов предметной области. | 2 459,52 | 14,64 | 24 | 351,36 |
| 3 | Ведущий инженер-програм-мист | Координация проекта, непосредственное участие в разработке продукта. | 2 926,56 | 17,42 | 56 | 975,52 |
| 4 | Инженер-програм-мист | Разработка ПС, т.е. написание программного кода. | 2 640,96 | 15,72 | 120 | 1 886,4 |
| 5 | Тестировщик | Тестирование ПС с целью выявления ошибок в его работе и последующее исправление оных. | 1 990,8 | 11,85 | 16 | 189,6 |
| 6 | Дизайнер | Проектирование и создание интерфейса ПС, обеспечение максимальной удобности пользования. | 1 668,24 | 9,93 | 24 | 238,32 |
| Премия (50%) | | | | | | 1 910,6 |
| Итого затраты за основную зарплату разработчиков | | | | | | 5 731,8 |

7.2.2 Расчет затрат на дополнительную заработную плату

Дополнительная заработная плата исполнителей проекта., определяется по формуле:

где Зо – затраты на основную заработную плату, (руб.);

Нд – норматив дополнительной заработной платы (15%).

Таким образом, затраты на дополнительную заработную плату составят

:

7.2.3 Расчет отчислений на социальные нужды

Отчисления в фонд социальной защиты населения и на обязательное страхование определяются в соответствии с действующими законодательными актами по формуле:

где Зо – затраты на основную заработную плату, (руб.);

Зд – затраты на дополнительную заработную плату, (руб.)

Нсоц – норматив отчислений в фонд социальной защиты населения и на обязательное страхование (34,6%).

Таким образом, отчисления в фонд социальной защиты населения и на обязательное страхование составят:

7.2.4 Расчет прочих затрат

Прочие затраты включают затраты, связанные с разработкой конкретного программного обеспечения напрямую, а также связанные с функционированием организации-разработчика в целом. Расчет прочих затрат выполняется в процентах от затрат на основную заработную плату команды разработчиков с учетом премии по формуле:

где Зо – затраты на основную заработную плату, (руб.);

Нпз – норматив прочих затрат (125%).

Таким образом:

Полная сумма затрат на разработку программного обеспечения находится путем суммирования всех рассчитанных статей затрат. Расчет приведен в таблице 7.2.

Таблица 7.2 – Затраты на разработку программного обеспечения.

|  |  |
| --- | --- |
| Статья затрат | Сумма, р |
| 1 | 2 |
| Основная заработная плата команды разработчиков | 5 731,8 |
| Дополнительная заработная плата команды разработчиков | 859,77 |
| Отчисления на социальные нужды | 2 280,68 |
| Прочие затраты | 6 878,16 |
| Общая сумма затрат на разработку | 15 750,41 |

Рассчитанное значение полной себестоимости, которое составило 15750,41 руб., будет использоваться в дальнейшем для определения экономического эффекта, а также уровня рентабельности разработки и внедрения программного средства.

## 7.3 Оценка экономического эффекта от использования ПС

Разрабатываемое программное средство предоставляет экономический эффект. На консультацию покупателя время у маркетолога примерно составляет 10 минут, что в процентном соотношении от часа примерно равно 15%. Среднее время, затрачиваемое на консультацию клиента и дистанционный подбор товара с использованием программного средства, сократится примерно на 30% для одного заказа. В двух отделах розничной торговли в сумме работает 12 менеджеров по продажам, для которых экономия выразится в освобождении времени на консультации по телефону и предоставление подробной информации о товарах посредством электронной почты, что позволит увеличить количество обрабатываемых заказов.

Рассчитаем сэкономленное за год время, поместив в первые скобки произведение количества месяцев, среднего количества рабочих дней в месяце и процента сэкономленного за час времени сотрудника, а во вторые – количество часов в рабочем дне и количество работников:

Часовая заработная плата менеджера по продажам составляет 10,2 руб.

На основе сэкономленного времени вычислим экономию затрат:

Рассчитаем из полученных данных отчисления в ФСЗН и получим следующее:

а экономия затрат после вычета отчислений составит:

Годовая экономия текущих затрат составит 22 285,98 рублей.

Экономический эффект рассчитывается по следующей формуле:

, (7.5)

где Эз  - экономия текущих затрат, полученная в случае применения программного средства, руб;

∆Зтек – прирост текущих затрат, связанных с использованием программного средства, руб;

Нп – ставка налога на прибыль в соответствии с действующим законодательством, %.

Таким образом, экономический эффект составит:

Уровень рентабельности рассчитывается по формуле 7.6:

где П – прибыль, получаемая отреализации данного ПС (руб.);

Зр – общая сумма затрат на разработку ПС (руб.).

Рассчитаем показатель рентабельности:

По данным затрат на разработку 15 750,41 руб., экономия затрат составит 5 359,16 руб. При этом уровень рентабельности разработки и внедрения программного средства составляет 34%.

# Заключение

В ходе дипломного проектирования проведен анализ литературы по теме дипломного проекта, изучены наиболее известные решения в сфере Интернет-торговли. В процессе работы были исследованы и выделены основные современные тенденции в области электронной коммерции. Для каждого из упомянутых в отчете программных средств выявлены достоинства и недостатки использования, которые учитывались при разработке спецификации требований к программному средству.

Также в ходе моделирования предметной области была разработана функциональная и информационная модели программного обеспечения. На основе разработанных моделей сформулирована функциональная спецификация к программному обеспечению.

Исходя из полученных на этапе моделирования данных, была спроектирована архитектура программного решения, которая смогла бы позволить точно реализовать требования, указанные в функциональной спецификации. Также на данном этапе были определены архитектурные принципы, в соответствие с которыми должна производиться разработка программного решения.

На основании информационной модели была спроектирована модель базы данных. Данная модель призвана покрыть все необходимые аспекты, связанные с хранением сущностей, используемых внутри системы.

Большую часть времени дипломного проектирования заняло написание клиентской части приложения.

Таким образом, задачи, поставленные в рамках индивидуального задания, были выполнены. Знания и опыт полученные в процессе прохождения дипломного проектирования будут полезны при дальнейшей работе по специальности.

# Список использованной литературы

1. ГОСТ 34.003-90. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Термины и определения. // Информационная технология. Комплекс стандартов и руководящих документов на автоматизированные системы. – М.: Изд-во стандартов, 1991.
2. Appendix D. SQL Conformance. PostgreSQL 9 Documentation. PostgreSQL Global Development Group [Электронный ресурс]. – 2009 – Режим доступа: https://www.postgresql.org/docs/current/features.html – Дата доступа: 20.11.2020
3. Реляционные базы данных обречены? [Электронный ресурс]. – 2016 – Режим доступа: habrahabr.ru/post/103021/ – Дата доступа: 25.11.2020
4. Django FAQ about MVC in Django [Электронный ресурс]. – 2020 – Режим доступа: https://docs.djangoproject.com/en/dev/faq/general/#django-appears-to-be-a-mvc-framework-but-you-call-the-controller-the-view-and-the-view-the-template-how-come-you-don-t-use-the-standard-names – Дата доступа: 06.12.2020
5. Хассан, Г. UML-проектирование систем реального времени параллельных и распределенных приложений / Г. Хассан – М.: ДМК Пресс, 2011 – 704c.
6. Дипломные проекты (работы) общие требования СТП 01–2010 [Электронный ресурс]: стандарт предприятия / БГУИР – Электронные данные. – Режим доступа: СТП П2010 бгуир.pdf
7. Технико-экономическое обоснование дипломных проектов: Методическое пособие для студентов всех специальностей БГУИР. В 4-ч. Ч. 4: Проекты программного обеспечения / В.А. Палицын. – Минск: БГУИР, 2006 г. – 76 с.
8. Statista – Business Data Platform [Электронный ресурс] 2017 – Режим доступа: https://www.statista.com/statistics/379046/worldwide-retail-e-commerce-sales/
9. Майкл Миковски, Джош Пауэлл. Разработка одностраничных веб-приложений = Single Page Web Applications: JavaScript End-to-end. – ДМК Пресс, 2014. – 512 с. – ISBN 978-5-457-83457-6.
10. Nuxt.js: a Minimalist Framework for Creating Universal Vue.js Apps [Электронный ресурс] – 2019 – Режим доступа: https://www.sitepoint.com/nuxt-js-universal-vue-js/ – Дата доступа: 13.12.2020

# Приложение А

**(обязательное)**

**Текст программного средства**