

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет
вычислительной техники

Кафедра
«Вычислительная техника»

Направление подготовки: 09.03.01 «Информатика и вычислительная техника»

Профиль подготовки: «Вычислительные машины, комплексы, системы и сети».

БАКАЛАВРСКАЯ РАБОТА

на тему

ИНТЕРНЕТ-МАГАЗИН

ЭЛЕКТРОНИКИ

Студент Капелла В.А.
(подпись, дата) 30.05.2025

Руководитель Карамышева Н.С.
(подпись, дата) 30.05.2025

Нормоконтролёр А.С. Бычков
(подпись, дата) 30.05.2025

Работа допущена к защите (протокол заседания кафедры от 28.05.2025 г. № 16)

Заведующий кафедрой М.А. Митрохин
(подпись)

Работа защищена с отметкой хорошо (протокол заседания ГЭК от 06.06.25 № 6)

Секретарь ГЭК Е.И. Гурин
(подпись)

Пенза 2025

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет
вычислительной техники

Кафедра
«Вычислительная техника»

«Утверждаю»
Заведующий кафедрой
М.А. Митрохин
«06» 05 2025г.

**ЗАДАНИЕ
НА БАКАЛАВРСКУЮ РАБОТУ**

1. Студент Капелла Владислав Андреевич гр.21BVB1факультета вычислительной техники.

2. Тема работы: «Интернет-магазин электроники»

Тема утверждена приказом ПГУ № 604/0 от "06" 05 2025 г.

3. Руководитель бакалаврской работы к.т.н., доцент, каф. ВТ ПГУ Карамышева Н.С.

4. Задание на работу (назначение разработки, исходные данные и т.п.)

Разработать веб-приложение «Интернет-магазин электроники», позволяющее обеспечить простоту и эффективность использования для пользователей. Приложение должно обладать графическим и интуитивно понятным интерфейсом, для взаимодействия с пользователем. Приложение должно предоставлять возможность для взаимодействия с информацией о товарах, а также иметь широкий ассортимент и возможность приобретения. Разрабатываемое веб-приложение также должно выполнять задачи по сортировке товаров.

5. Перечень подлежащих разработке вопросов

В рамках разработки веб-приложения интернет-магазина электроники необходимо провести комплексный анализ предметной области. В первую очередь, требуется изучить существующие интернет-магазины, выделив их достоинства и недостатки. Это позволит выявить лучшие практики и ошибки, которые следует избежать при разработке собственного решения. Следующим шагом будет выбор языка программирования и архитектуры приложения.

6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Изучение предметной области	52	06.05.2025–09.05.2025	
2	Анализ требований	13	10.05.2025	
3	Создание макета графического интерфейса	52	11.05.2025–14.05.2025	
4	Выбор технологий, применяемых при проектировании и разработке	39	15.05.2025–17.05.2025	
5	Реализация графического интерфейса	104	16.05.2022 – 23.05.2022	
6	Разработка основной части программы	70	24.05.2022 – 29.05.2022	
7	Оформление пояснительной записки	32	29.05.2022 – 30.05.2022	

Дата выдачи задания 6 «мая» 2025г.

Руководитель бакалаврской работы  06.05.2025 Карамышева Н.С.
(подпись, дата)

Задание к исполнению принял студент  06.05.2025 Капелла В.А.
(подпись, дата)

Работу к защите допустить
Декан факультета  03.06.2025 Фионова Л.Р.
(подпись, дата)

Реферат

Бакалаврская работа содержит 127 листов, 40 рисунков, 3 таблицы, 13 источников и 3 приложения.

ВЕБ-ПРИЛОЖЕНИЕ, ВИЗУАЛЬНОЕ ПРОГРАММИРОВАНИЕ, БАЗА ДАННЫХ, КЛИЕНТ-СЕРВЕРНОЕ ПРИЛОЖЕНИЕ, JAVASCRIPT, REACTJS, JSON

Цель работы заключается в разработке веб-приложения для интернет-магазина электроники, которое предоставляет пользователям возможность удобно просматривать, выбирать и приобретать электронные устройства и аксессуары. Приложение будет ориентировано на создание комфортного и интуитивно понятного интерфейса для покупателей.

Объектом исследования в рамках работы являются системы электронной коммерции и организации онлайн-продаж, которые способствуют улучшению покупательского опыта.

В проекте применяются методы разработки веб-приложений с использованием HTML и JavaScript, а также программная реализация с использованием фреймворка ReactJS.

В ходе работы был проведен обзор существующих систем электронной коммерции, организующих процессы онлайн-продаж. Также рассмотрены современные платформы визуального программирования, протестированы различные среды разработки и готовые решения, а также определены ключевые характеристики, необходимые для успешного функционирования разрабатываемого интернет-магазина.

					ПГУ 1.090301.8.001 ПЗ			
Изм.	Лист	№ докум.	Подпись	Дата	Интернет-магазин электроники Пояснительная записка	Лит.	Лист	Листов
Разраб.		Капелла В.А.		10.05.21				
Провер.		Карамышева Н.С.		10.05.21			2	113
Н. Контр		Бычков А.С.		10.05.21				
Утверд.		Митрахин М.А.		10.05.21				

Содержание

Введение.....	6
1 Анализ предметной области.....	7
1.1 Описание предметной области	7
1.2 Анализ существующих аналогов.....	8
1.2.1 Интернет-магазин «DNS».....	8
1.2.2 Интернет-магазин «М.Видео»	9
1.2.3 Интернет-магазин «Ситилинк».....	10
1.2.4 Интернет-магазин «Мегамаркет»	11
1.2.5 Интернет-магазин «Ozon»	13
1.2.6 Интернет-магазин «Wildberries».....	13
1.2.7 Сравнение существующих аналогов	14
1.3 Постановка задачи.....	16
1.4 Выбор технологий и средства разработки приложения.....	17
1.4.1 Выбор языка JavaScript.....	19
1.4.2 Выбор языка CSS.....	20
1.4.3 Выбор языка HTML	21
1.5 Выбор инструментов разработки	21
1.5.1 Выбор среды разработки	21
1.5.2 Выбор вспомогательного фреймворка.....	22
1.5.3 Выбор системы управления базами данных	23
1.6 Проектирование приложения.....	25
1.6.1 Паттерн MVC.....	25
1.6.2 Архитектура приложения.....	26
1.6.3 Проектирование базы данных.....	30
1.6.4 Проектирование графического интерфейса	31
1.7 Реализация приложения.....	32
1.7.1 Создание стилей	32

1.7.2 Реализация сервера	33
1.7.3 Реализация базы данных.....	35
1.7.5 Реализация компонентов	36
1.7.6 Руководство пользователя.....	39
2 Охрана труда.....	50
Заключение	54
Список использованных источников	54
Приложение А – UML–диаграммы	57
Приложение Б – Листинг программы	63
Приложение А – Презентация.....	105

Перечень сокращений и обозначений

ГОСТ – региональный стандарт, принятый Межгосударственным советом по стандартизации, метрологии и сертификации Содружества Независимых Государств

СУБД – система управления базами данных

ПК – персональный компьютер

Фреймворк – программная платформа, которая упрощает разработку программного продукта, определяет структуру проекта

API (Application Programming Interface) – описание способов, которыми одна компьютерная программа может взаимодействовать с другой программой

CSS – язык описания внешнего вида документа

IDE (Integrated development environment) – интегрированная среда разработки

ID (Identifier) – уникальный номер объекта

JSON (Java Script Object Notation) – текстовый формат обмена данными

MVC (Model View Controller) – шаблон проектирования архитектуры приложения

MongoDB — система управления базами данных, которая работает с документоориентированной моделью данных

PK (Primary key) – первичный ключ, понятие теории реляционных баз данных, относящееся к ограничениям целостности базы данных

UML (Unified Modeling Language) – унифицированный язык моделирования

.NET – кроссплатформенная платформа для создания различных типов приложений

					ПГУ 1.090301.8.001 ПЗ	Лист
						5
Изм.	Лист	№ докум.	Подпись	Дата		

Введение

Большую долю современной жизни занимают информационные технологии, а сам поток информации с каждым годом становится все больше. Одним из видов источников информации являются интернет-магазины. Они включают в себя методы решения заранее спланированных задач. Жизнь становится гораздо комфортнее благодаря таким сервисам.

Целью дипломной работы является разработка интернет-магазина электроники. Для достижения этой цели требуется разработать веб-приложение отвечающее современным стандартам. В проекте будут задействованы языки программирования JavaScript и Node.js для разработки серверной части, а ReactJs, HTML и CSS для разработки клиентской.

Интернет-магазины начинают занимать ключевую роль нашей повседневной жизни. Они имеют ряд преимуществ в сравнении с оффлайн-магазинами, такие как возможность выбора товаров из дома, широкий ассортимент товаров, частые скидки и главное возможность совершать покупки онлайн. Не менее важным является тот факт, что такие магазины позволяют пользователям совершать покупки в любое время, предлагают сравнение различных товаров и предоставляют обратную связь клиентам.

1 Анализ предметной области

1.1 Описание предметной области

В связи с тем, что с каждым годом популярность интернет-магазинов лишь растет по сравнению с традиционными магазинами, можно с уверенностью сказать, что существует ряд причин из-за чего так происходит.

Главными преимуществами интернет-магазинов являются:

- быстрота и удобство оформления заказа;
- возможность совершать покупки в любое время и в любом месте;
- большой ассортимент товаров;
- более выгодные цены и частые акции;
- система доставки, позволяющая доставить заказ практически в любое место.

Таким образом, интернет-магазины действительно заслуживают своей популярности, ведь имеют ряд преимуществ, которых нет в оффлайн-магазинах.

Предметная область дипломной работы – создание интернет-магазина электроники, что является востребованным в наши дни. С развитием технологий все больше людей предпочитают совершать покупки онлайн, в том числе и приобретать электронику.

Успешная реализация проекта требует провести анализ аналогов и рынка, изучить потребности покупателей и разработать уникальное предложение.

Основные этапы разработки включают в себя:

- выбор платформы для разработки;
- разработка дизайна и удобного интерфейса;
- интеграция платежных систем и методов доставки;
- обеспечение безопасной эксплуатации;
- создание широкого ассортимента товаров.

					ПГУ 1.090301.8.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		7

1.2 Анализ существующих аналогов

Среди наиболее популярных аналогов стоит выделить:

- «DNS»;
- «М.Видео»;
- «Ситилинк»;
- «MegaMarket»;
- «Ozon»;
- «Wildberries».

1.2.1 Интернет-магазин «DNS»

Главный экран интернет-магазина «DNS» и информация о товарах представлены на рисунках 1,2.

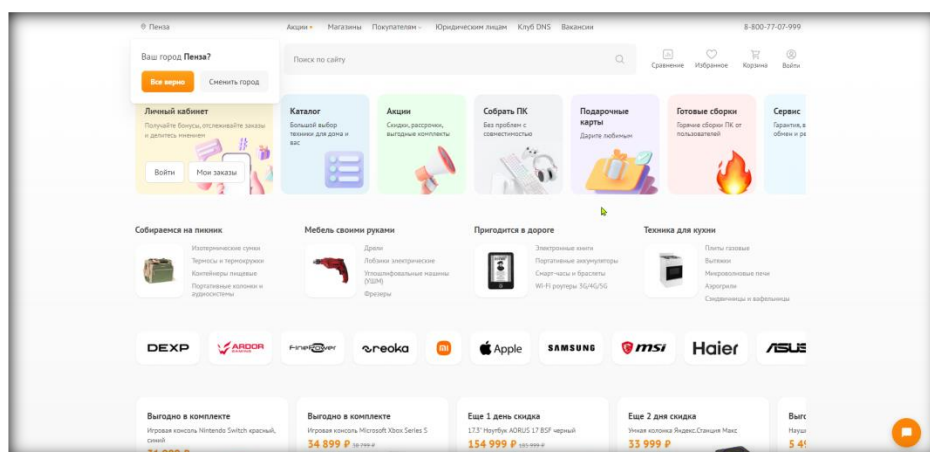


Рисунок 1 – Главный экран интернет-магазина «DNS»

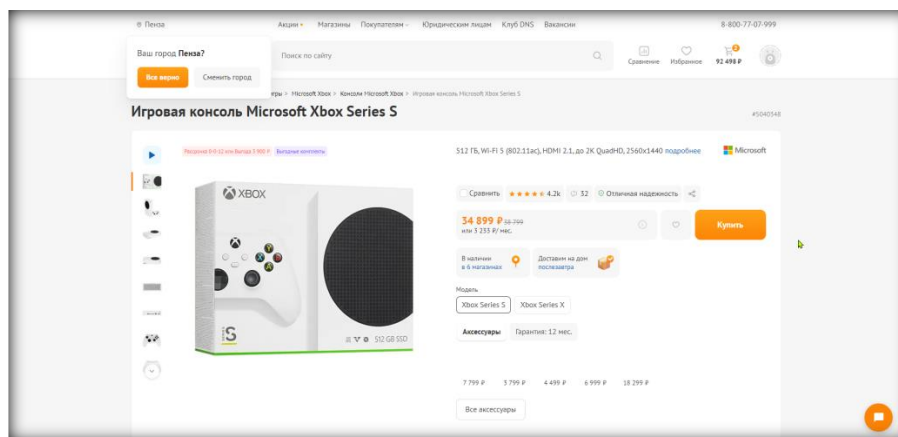


Рисунок 2 – Информация о товарах «DNS»

«DNS» — российская торговая сеть по продаже бытовой техники и электроники, основанная в 1998 году. Компания также выпускает технику под рядом собственных торговых марок. В 2010 году филиальная сеть компании состояла из более чем 100 магазинов в 28 городах России, в которых было занято более 1,5 тыс. сотрудников. На начало 2011 года было открыто более 185 магазинов в 60 городах России, численность коллектива составила более 3,5 тыс. сотрудников [1].

1.2.2 Интернет-магазин «М.Видео»

Главный экран интернет-магазина «М.Видео» и информация о товарах представлены на рисунках 3, 4.

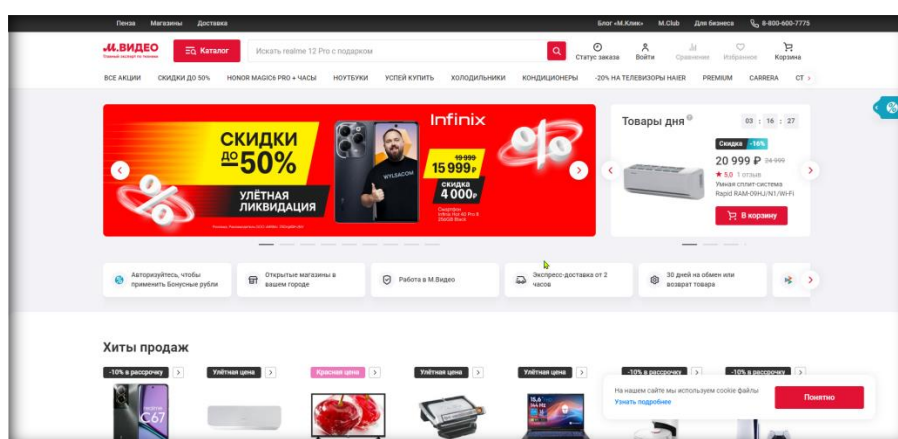


Рисунок 3 – Главный экран «М.Видео»

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

Лист
9

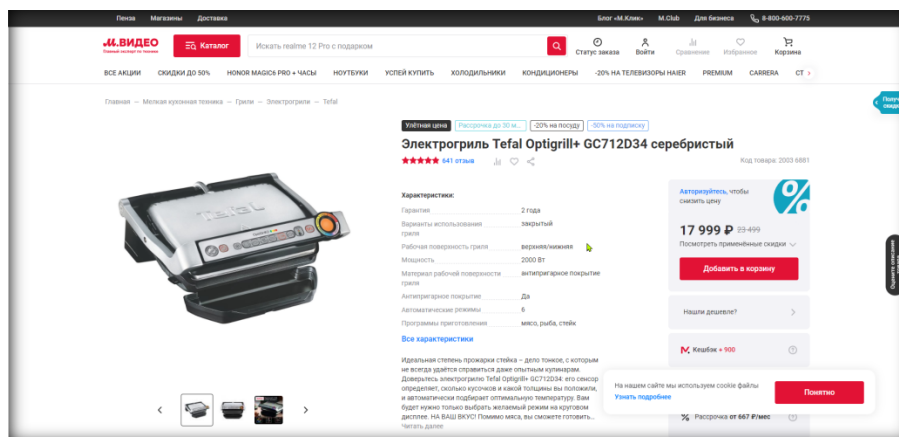


Рисунок 4 – Информация о товаре «М.Видео»

Розничная сеть «М.Видео» — одна из крупнейших розничных сетей по объёмам продаж электроники и бытовой техники в России. «М.Видео» осуществляет свою деятельность с 1993 года и является первой на российском рынке сетью, реализовавшей полноценный уникальный подход к продажам — это единый ассортимент, цена и сервис как при покупке в магазинах, так и онлайн. Ритейлер предлагает своим покупателям несколько десятков тысяч наименований аудио/видео и цифровой техники, мелкой и крупной бытовой техники, медиатоваров, а также аксессуаров. Магазины «М.Видео» имеют единый формат и специальную концепцию дизайна [2].

1.2.3 Интернет-магазин «Ситилинк»

Главный экран интернет-магазина «Ситилинк» и информация о товарах представлены на рисунках 5,6.

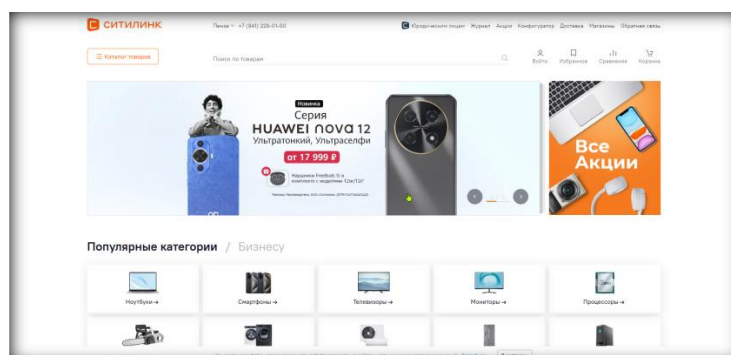


Рисунок 5 – Главный экран интернет-магазина «Ситилинк»

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

Лист
10

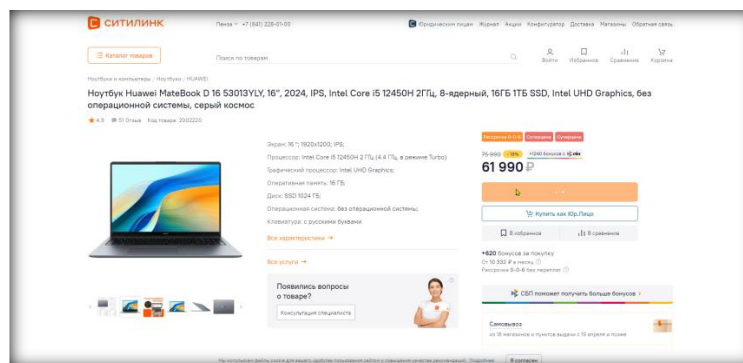


Рисунок 6 – Информация о товаре «Ситилинк»

«Ситилинк» — одна из крупнейших сетей магазинов электроники и бытовой техники, на рынке с 2008 года. Компания входит в топ-5 крупнейших интернет-магазинов России. «Ситилинк» продает порядка 111 тысяч наименований цифровой и бытовой техники, электроники. Сейчас у компании более 830 офлайн-магазинов и пунктов выдачи заказов в 62 регионах России. В портфеле услуг – сборка компьютеров и серверов под заказ, аутсорсинг ИТ-инфраструктуры, монтаж систем видеонаблюдения, установка бытовой и климатической техники, услуги страхования, покупка товаров в лизинг и другие услуги. У компании более 225 тысяч корпоративных клиентов.

Магазины «Ситилинк» оборудованы терминалами для самостоятельного оформления заказов, кассами оплаты и окнами выдачи товаров. Call-центр компании работает 7 дней в неделю и оказывает круглосуточную поддержку покупателям [3].

1.2.4 Интернет-магазин «Мегамаркет»

Главный экран интернет-магазина «Мегамаркет» и информация о товарах представлены на рисунках 7,8.

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

Лист
11

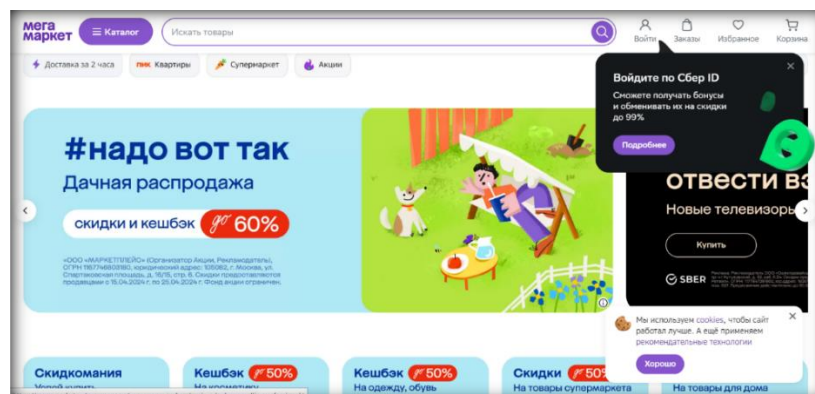


Рисунок 7 – Главный экран интернет-магазина «Мегамаркет»

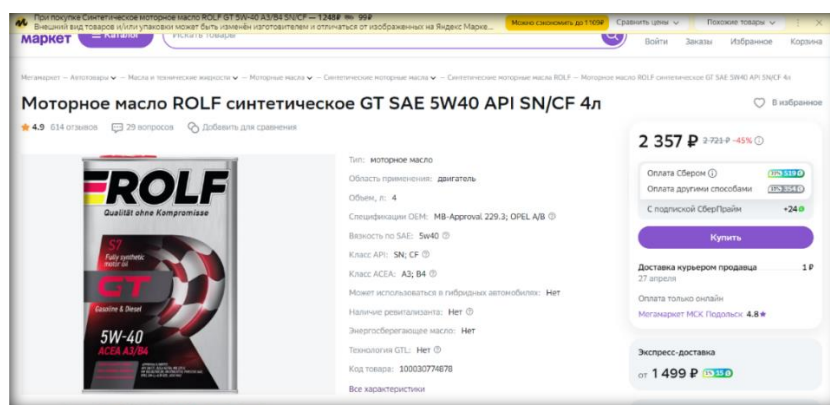


Рисунок 8 – Информация о товаре «Мегамаркет»

«Мегамаркет» — российский маркетплейс, основанный в 2017 году группой «М.Видео-Эльдорадо». С 2021 года принадлежит Сбербанку.

В апреле 2017 года компания запустила бета-тестирование. На старте клиентам была предложена доставка по Москве и Московской области, а также программа лояльности. Ассортимент включал 50 тысяч товаров в 7 категориях: бытовая техника, детские товары, зоотовары, электроника, товары для ремонта, сада и дачи, спорта и отдыха. В 2018 году ассортимент маркетплейса расширился до 600 000 позиций от 1500 продавцов, а выручка в четвёртом квартале превысила 1 миллиард рублей. На конец 2019 года на сайте goods.ru было представлено более 1 600 000 товаров от 3000 продавцов в 16 основных категориях [4].

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

Лист
12

1.2.5 Интернет-магазин «Ozon»

Главный экран интернет-магазина «Ozon» и информация о товарах представлены на рисунках 9,10.

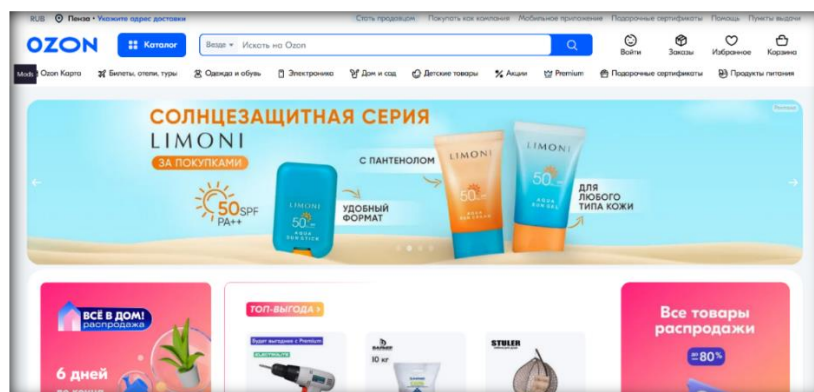


Рисунок 9 – Главный экран интернет-магазина «Ozon»

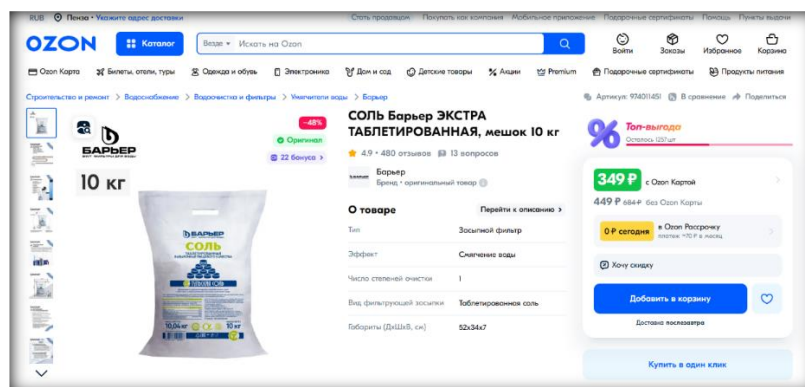


Рисунок 10 – Информация о товаре «Ozon»

«Ozon» — российский маркетплейс. Основан в 1998 году как интернет-магазин по продаже книг и видеокассет. Помимо торговой площадки, компания развивает экспресс-доставку товаров повседневного спроса Ozonfresh, доставку товаров из-за рубежа, Global.Ozon использует склады и сортировочные центры для хранения и доставки товаров. На конец 2024 года маркетплейс управлял 3,5 млн м² логистической инфраструктуры в России и СНГ (около 200 логистических объектов). Сеть пунктов выдачи развивается по партнёрской модели, работает более 60 тысяч точек [5].

Изм.	Лист	№ докум.	Подпись	Дата

1.2.6 Интернет-магазин «Wildberries»

Главный экран интернет-магазина «Wildberries» и информация о товарах представлены на рисунках 11,12.

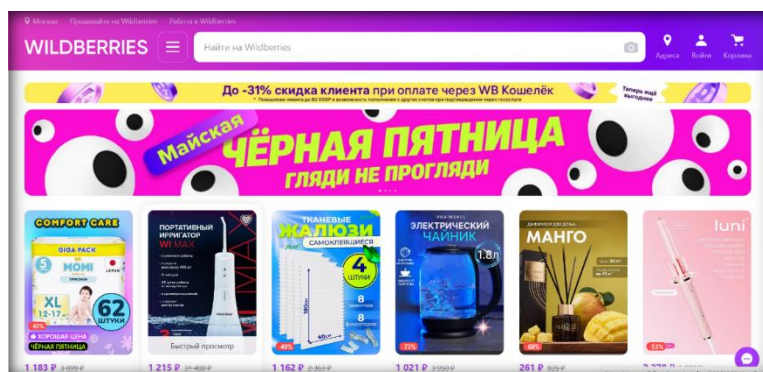


Рисунок 11 – Главный экран интернет-магазина «Wildberries»

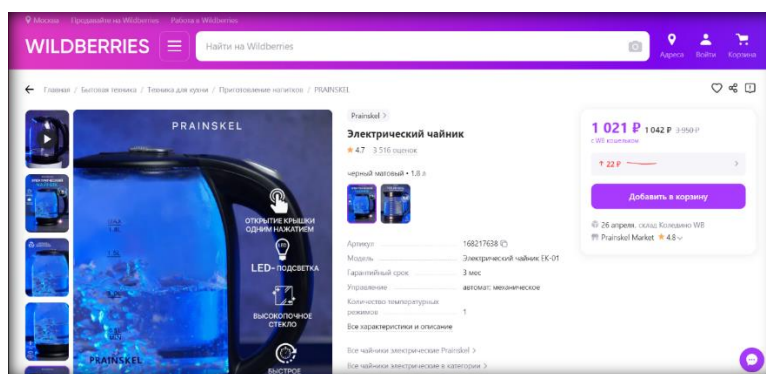


Рисунок 12 – Информация о товаре «Wildberries»

«Wildberries» – крупнейший онлайн-ритейлер России, работающий на рынке 17 лет. Оборóт «Wildberries» в 2021 году вырос на 93% и достиг 844 млрд руб. Ежедневно на «Wildberries» оформляется 4 млн заказов, ежемесячная аудитория онлайн-ритейлера - свыше 113 млн пользователей (MAU). Ассортимент «Wildberries» включает около 28 млн SKU товаров от 300 000 российских и зарубежных брендов: одежда и обувь, аксессуары, бьюти-товары, детские товары, техника и электроника, товары для дома, ювелирные украшения, продукты питания, книги, цифровые товары и многое другое. Около 50% товаров произведены в России [6].

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

Лист
14

1.2.7 Сравнение существующих аналогов

В таблице 1 представлено сравнение существующих аналогов.

Таблица 1 – Сравнение существующих аналогов

Название	Большой ассортимент электроники	Цена выше рыночной	Высокое качество электроники	Быстрая доставка	Быстрая поддержка
«DNS»	+	–	+	+	+
«М.Видео»	+	+	–	–	–
«Ситилинк»	+	–	+	–	–
«Мегамаркет»	–	–	–	+	+
«Ozon»	–	+	–	+	+
«WildBerries»	–	+	–	+	+

Исходя из таблицы ясно, что выбор каждого интернет-магазина электроники исходит из индивидуальных предпочтений пользователей.

Важными факторами при выборе могут являться:

- большой ассортимент электроники;
- ценообразование;
- качество электроники;
- время доставки;
- поддержка.

Каждый магазин при этом имеет свои сильные и слабые стороны.

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

Лист

15

1.3 Постановка задачи

Разработанное веб-приложение должно отвечать таким требованиям, как удобное взаимодействие, безопасность, простое оформление заказа. Также веб-приложение должно быть поддерживаемым на современных платформах.

Рассмотрим каждый пункт подробнее.

Удобное взаимодействие.

- пользователи не должны испытывать затруднений во время эксплуатации приложения;
- навигация по сайту должна быть понятной для каждого, должен быть реализован поиск товаров и корзина товаров.

Безопасность:

- веб-приложение должно гарантировать сохранность данных каждого пользователя;
- необходимо использовать шифрование данных, механизмы аутентификации и авторизации, а также надежные платежные системы;

Простое оформление заказа:

- должны быть предложены удобные способы оформления заказа и различные варианты оплаты.

Современные платформы и технологии:

- должны быть использованы современные веб-технологии (HTML, CSS, JavaScript);
- разработать пользовательский интерфейс при помощи фреймворка (например React или Angular);
- серверная часть приложения описана при помощи языков таких, как Node.js, Python или PHP с использованием RESTfulAPI.
- должна быть использована нереляционная база данных для хранения информации о товарах.

- операционная система: Windows (7,8, 10 и 11)), macOS X 10.11 или Linux (Ubuntu, Debian и Redhat);
- процессор: 1,6 ГГц или выше;
- память: 1 ГБ ОЗУ;
- 500 МБ свободного дискового пространства;
- Microsoft .NET Framework 4.5.2.

UML–диаграмма прецедентов пользователя и администратора представлена на рисунке 13.



Рисунок 13 – UML–диаграмма прецедентов пользователя и администратора

1.4 Выбор технологий и средства разработки приложения

В таблице 2 представлены технические характеристики различных интернет-магазинов электроники.

Изм.	Лист	№ докум.	Подпись	Дата

Таблица 2 – Технические характеристики магазинов

Название Магазина	Ссылка на магазин	Хостинг	База данных	Backend	Frontend
«DNS»	https://www.dns-shop.ru/	Amazon Web Services	Postgre SQL, Oracle	Java, Python	JavaScript, HTML, CSS
«М.Видео»	https://www.mvideo.ru/	Amazon Web Services	Oracle, MySQL	Java, Python Spring Framework	React, HTML, CSS, JS
«Ситилинк»	https://www.citilink.ru/	Amazon Web Services	Oracle Database	JavaScript, Django, Python	AngularJS, jQuery, HTML, CSS

Из представленной таблицы следует, что каждая площадка имеет свои различия в использовании технологий и средств разработки.

При выборе языка программирования стоит учитывать, как он будет влиять на производительность всего приложения, безопасность и удобство обслуживания.

Рассмотрим критерии выбора языка программирования:

- веб-приложение должно быть динамическим;
- выбранный язык программирования должен поддерживать растущую базу данных пользователей;

- выбранный язык должен отвечать современным стандартам безопасности и быть устойчив к уязвимостям;
- язык программирования не должен быть устарелым, как и его поддержка.

Разработка веб-приложения предлагает использование многих языков программирования, каждый из которых имеет свои плюсы и минусы. Одним из наиболее популярных является Python. Его синтаксис и простота привлекают все больше разработчиков, так как процесс создания кода становится в разы легче и быстрее, а выборка библиотек и фреймворков, таких как Flask и Django, ускоряет разработку всего приложения. Однако Python не всегда используется для разработки высоконагруженных приложений. С такими задачами лучше справляются C# или Java.

Язык программирования C# известен своей отличной интеграцией с платформой .NET, что облегчает создание мощных и производительных приложений. Язык также поддерживает объектно-ориентированное программирование и современные методы разработки. Сопутствующая документация часто обновляется, что и делает этот язык одним из хороших выборов для разработчиков.

Язык программирования Java часто используется при веб-разработке, так как имеют высокую производительность и возможность дальнейшего масштабирования. Этот язык имеет обширную коллекцию библиотек и фреймворков, таких как Spring. Однако синтаксис Java по сравнению с Python или C# более сложный, а компиляция может быть дольше [7].

1.4.1 Выбор языка JavaScript

При разработке веб-приложения был выбран язык JavaScript.

Преимущества языка JavaScript:

- универсальность языка, то есть он поддерживается всеми современными браузерами. С помощью него клиентская часть будет описана

проще, так как язык предоставляет возможность создавать интерактивные веб-страницы, которые будут работать на любой платформе;

- легок в обучении, так как имеет простой синтаксис, а дополнительное использование фреймворков таких, как React или Angular упрощают разработку приложения;

- поддерживает асинхронную работу приложения и возможность работы с данными для дальнейшей отправки на сервер;

- один из популярных и развивающихся языков, что означает постоянное наличие актуальной документации [8].

1.4.2 Выбор языка CSS

CSS – язык стилей, использующийся для оформления веб-страниц. Главная особенность данного языка заключается в том, что разработчику не нужно изменять HTML код, чтобы изменить внешний вид приложения, то есть язык дает возможность разделить оформление.

Преимущества CSS при разработке веб-приложений заключаются в его гибкости, это позволяет создать стиль для каждого компонента или элемента. Язык можно адаптировать под любую платформу, также он использует довольно простой синтаксис. После создания стилей они будут одинаково отображаться в разных браузерах, это обеспечивает единый вид всего приложения, а его возможность стилизации, например Flexbox или Grid, обеспечивает более точное создание макетов. Ко всему прочему, CSS поддерживает использование классов, идентификаторов и других спецификаторов. Разработчик всегда может создать анимированные эффекты для своего приложения, не используя JavaScript.

В связи с этим, CSS служит хорошим и удобным инструментом для стилизации веб-приложений. Его зачастую используют для frontend-разработки, и наш проект не будет исключением [9].

1.4.3 Выбор языка HTML

HTML – язык разметки, использующийся для создания веб-страниц. Он работает во всех браузерах, так что наше приложение можно будет открыть в каждом из них.

HTML состоит из тегов и элементов, при помощи которых можно организовать и настроить последовательность элементов на странице. Его универсальность позволяет тратить меньше времени на разработку приложений, а в сочетании с JavaScript появляется возможность добавления интерактивных элементов [9].

1.5 Выбор инструментов разработки

1.5.1 Выбор среды разработки

Среди самых популярных сред разработок стоит выделить:

- Eclipse;
- NetBeans;
- Microsoft Visual Code;
- JetBrains IntelliJ IDEA.

Открытая интегрированная среда разработки Eclipse поддерживает множество языков программирования так, как имеет плагины для работы с Python, C/C++, Java и другими языками. Разработчики данной среды предлагают пользователям тонкую настройку под свои задачи, а постоянно обновляющаяся актуальная документация дает возможность быть в курсе последних изменений. Однако сама настройка может вызывать затруднения, а производительность самой среды при использовании большего количества плагинов может уступать другим в некоторых аспектах [10].

Среда разработки NetBeans обладает понятным интерфейсом и простой пользования. Многие разработчики на Java используют именно ее, так как среда является бесплатной и доступной и имеет интеграцию с различными системами сборки. Но при всех плюсах NetBeans имеет меньше

возможностей по сравнению с аналогами и ограничение по количеству подключенных плагинов. Сама производительность тоже уступает другим средам разработки [10].

Среда JetBrains IntelliJ IDEA имеет полезные функции дополнения и исправления кода, что позволяет оптимизировать процесс разработки. Она поддерживает многие языки программирования, предлагает различные версии для веб-разработки, например WebStorm. Интерфейс среды тоже является ее огромным плюсом, как и ее возможность интеграции с системой контроля версий приложения. Главным же ее недостатком является то, что полная версия приложения платная, а сама среда нуждается в больших ресурсах системы при работе [10].

После анализа решено было выбрать среду разработки Microsoft Visual Code. Быстрый и простой редактор кода в сочетании с различными удобными функциями по его заполнению и исправлению делают эту среду одной из самых удобных. MVC поддерживает множество языков, это позволяет разработчику создавать проект, не прибегая к сторонним приложениям. Также ключевым аспектом при выборе стала доступность среды, она является бесплатной, а постоянное обновление и добавление документации позволяют всегда найти интересующие материалы.

1.5.2 Выбор вспомогательного фреймворка

В качестве вспомогательного фреймворка был выбран ReactJS.

ReactJS является одной из самых популярных библиотек JavaScript для разработки пользовательских интерфейсов. Данная библиотека обладает рядом преимуществ, что делает процесс разработки приложения более удобным.

Главная особенность фреймворка – возможность разделения проекта на независимые друг от друга компоненты, что дает возможность использовать их в разных частях приложения. Также ReactJS предлагает к использованию Virtual DOM, в следствии чего при разработке и использовании приложения

					ПГУ 1.090301.8.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		22

будут обновляться те страницы, которые были изменены, а не все приложение полностью. Более того, при разработке мы можем использовать React Hooks – API, представленный в ReactJS 16.8 и позволяющий использовать возможности React без создания классов внутри компонентов.

1.5.3 Выбор системы управления базами данных

Выбирая систему управления базами данных стоит обратить внимание на ее производительность, эффективность, совместимость и простоту использования. Для начала рассмотрим две наиболее популярных системы MySQL и MongoDB.

Плюсы MySQL перед иными СУБД вытекают из задач, под которые данный программный продукт предназначен.

Преимущества:

- имеет в своем функционале различные функции безопасности, включая возможность установки привилегий пользователя, шифрование данных, аутентификацию и аудит, чтобы обезопасить информацию, хранящуюся в базах данных;
- обладает хорошей производительностью и быстродействием благодаря оптимизированным алгоритмам выполнения запросов и умеет работать с большими объемами данных с минимальными задержками;
- может использоваться как для небольших веб-приложений, так и для серьезных корпоративных систем. СУБД предлагает различные методы масштабирования, включая горизонтальное и вертикальное масштабирование, что дает возможность расширять базы данных при возникновении необходимости [11].

Недостатки:

- может иметь проблемы с производительностью при обработке больших объемов данных или когда требуется обработка сложных запросов или подключение большого количества клиентов;

– имеет ограниченный набор типов данных по сравнению с некоторыми другими СУБД, например, поддержка временных данных и географических данных ограничена по сравнению с некоторыми другими СУБД;

– может столкнуться с ограничениями в масштабируемости и доступности при взаимодействии с большим количеством клиентов [11].

MongoDB, в отличие от MySQL, предлагает разные удобные функции, такие как возможность хранения данных в формате JSON. Эта нереляционная база данных позволяет хранить сами данные в виде документов, что делает их дальнейшее редактирование удобнее.

Преимущества:

– позволяет хранить данные в формате JSON с произвольной структурой;

– операции выполняются локально, что позволяет избежать необходимость сетевых запросов, ускоряет время отклика приложения;

– проще в развертывании и настройке, требует меньше ресурсов и времени по сравнению с реляционными СУБД;

– лучший контроль безопасности из-за формата данных.

Недостатки:

– отсутствие жестких схем может привести к проблемам целостности данных;

– требуется больше ресурсов для выполнений сложных запросов;

В результате анализа было принято решение об использовании MongoDB из-за способности обрабатывать большие объемы информации, что является ключевым фактором при создании интернет-магазина электроники. Более того скорость доступа к данным влияет на более комфортное использование приложения, а простота развертывания может оптимизировать процесс разработки.

1.6 Проектирование приложения

1.6.1 Паттерн MVC

Архитектурный шаблон MVC (Model–View–Controller) – основная цель применения этой концепции состоит в отделении бизнес-логики от её визуализации. За счёт такого разделения повышается возможность повторного использования кода. Наиболее полезно применение данной концепции в тех случаях, когда пользователь должен видеть те же самые данные одновременно в различных контекстах и/или с различных точек зрения [12].

Шаблон разделен на три компонента: модель, представление и контроллер. Модель отвечает за хранения данных, а также за выполнение логики приложения. Представление – интерфейс для взаимодействия пользователя с приложением, который отображает информацию и предоставляет возможность пользователю взаимодействовать с приложением. Контроллер связывает два этих компонента, обрабатывает запросы от пользователя, работая с моделью и получая данные от представления.

Логическая схема MVC–приложения представлена на рисунке 14.

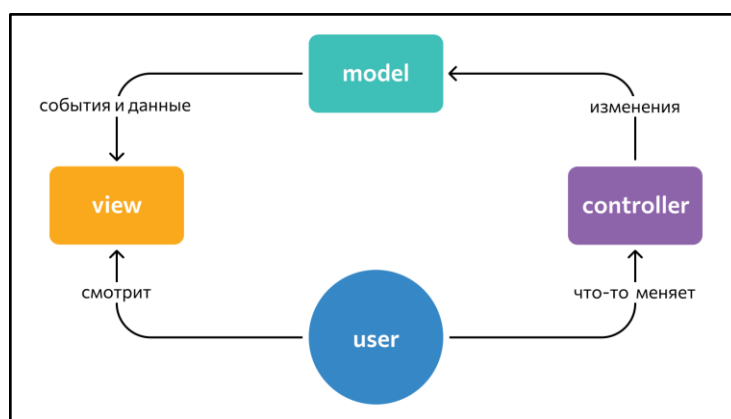


Рисунок 14 – Логическая схема MVC–приложения

После взаимодействия пользователя с представлением начинается обмен данными. При обновлении модели контроллер обновляет представление, записывая туда актуальные данные. В следствии чего пользователь видит обновленные данные.

Главным преимуществом MVC является четкое разделение задач, это упрощает организацию кода. В следствии чего присутствует возможность тестирования каждого компонента отдельно, давая возможность выявить и устранить ошибки на этапе разработки. Каждый компонент может быть использован повторно в других проектах, экономя время разработчиков. Архитектура MVC обеспечивает масштабируемость, что позволяет легко добавлять или удалять компоненты для адаптации приложения. Данный шаблон часто используют для разработки веб-приложений, где браузер является представлением, а сервер содержит модель и контроллер.

Примером наглядной работы архитектуры MVC будет любое веб-приложение, предоставляющее услуги коммерции. Данные о товарах, пользователях и заказов содержаться в модели. Страница оформления заказа, каталога товаров и данных о пользователе включаются в представление. За обработку запросов, получение данных и их отображение в представлении, действия пользователей по добавлению товаров в корзину или оформление заказа отвечает контроллер.

1.6.2 Архитектура приложения

Выбор архитектуры приложения определяется несколькими компонентами, которые должны обеспечить функциональность и производительность системы. Клиентская часть отвечает за обработку ввода информации от пользователя и ее вывода. Она взаимодействует с сервером, используя протоколы передачи данных, такие как HTTP.

Серверная часть отвечает за обработку запросов от клиентов, делает запросы сама и взаимодействует с базой данных. Описываемая часть может быть поделена на несколько уровней: веб-сервер, сервер приложений и слой

					ПГУ 1.090301.8.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		26

доступа к данным. Такое разделение обеспечивает лучшую структурированность кода, что упрощает поддержку приложения.

API протоколы обеспечивают обмен данными, сюда же относится и работа с СУБД. Сетевые запросы дают возможность компонентам взаимодействовать между собой, обеспечивая управление состоянием приложения и передачу данных.

При правильной спроектированной архитектуре клиент–серверного приложения с базой данных открывается возможность расширяемости и безопасности приложения, что является важным аспектом при адаптировании к новым требованиям в будущем, удовлетворении запросов пользователей и развитии приложения.

UML– диаграмма компонентов представлена на рисунке 15.

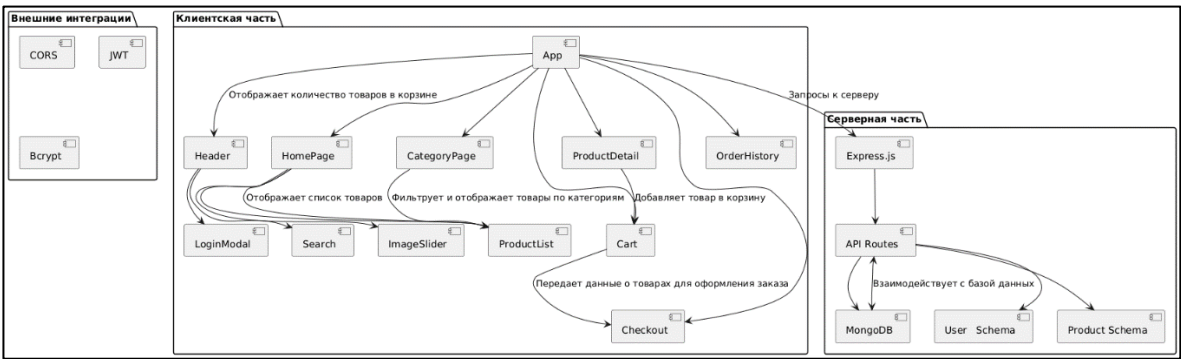


Рисунок 15 – UML – диаграмма компонентов

UML–диаграмма развертывания приложения находится на рисунке 16.

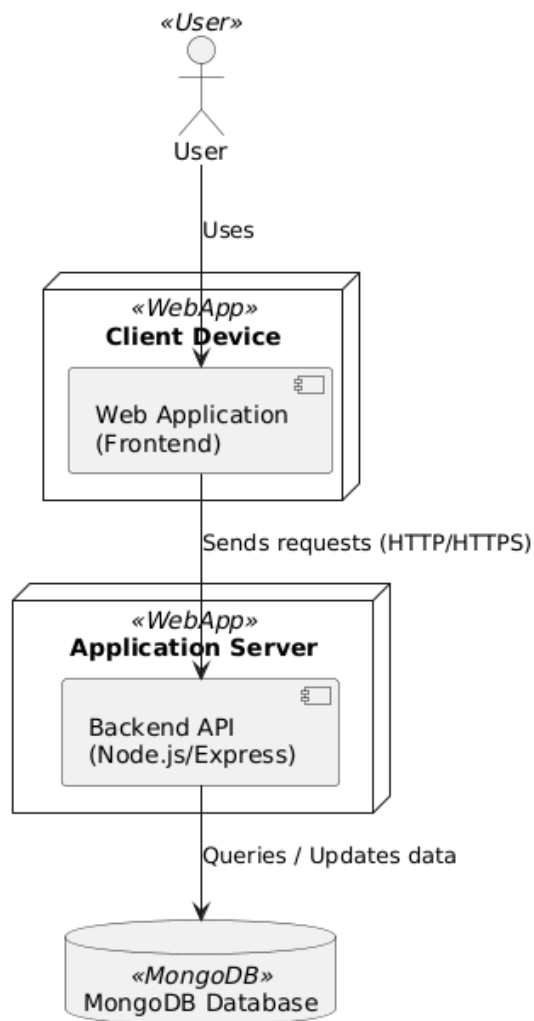


Рисунок 16 – UML–диаграмма развертывания приложения

UML–диаграмма последовательности добавления товара в корзину и его оформления представлена на рисунке 17.

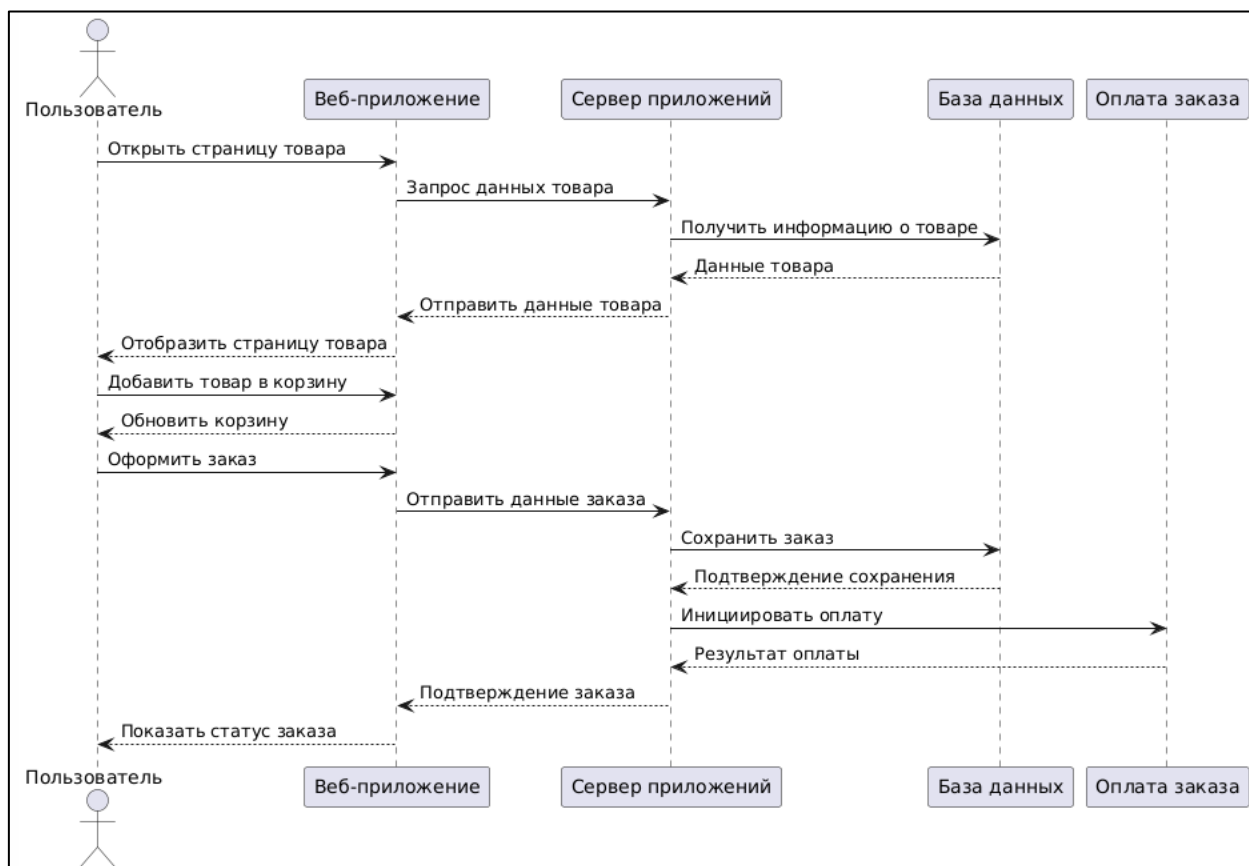


Рисунок 17 – UML–диаграмма последовательностей

UML–диаграмма деятельности веб–приложения представлена на рисунке 18.

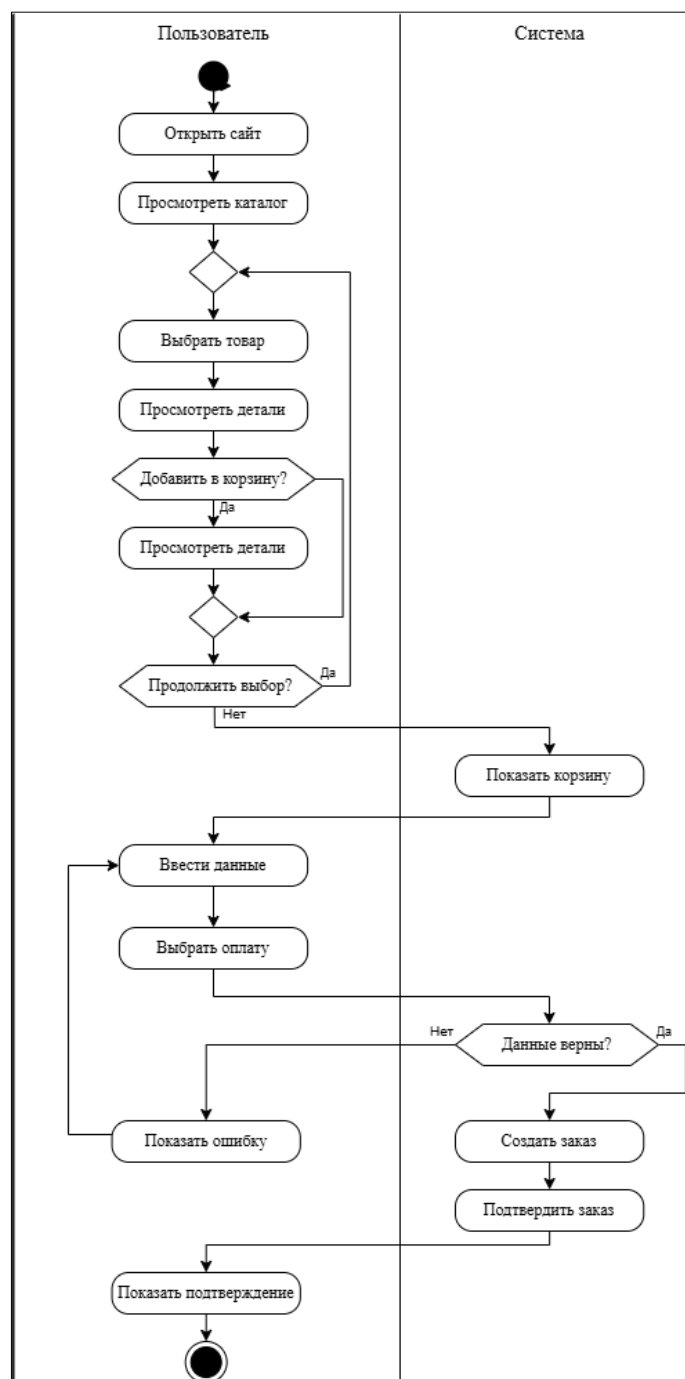


Рисунок 18 – UML–диаграмма деятельности веб–приложения

1.6.3 Проектирование базы данных

Для хранения информации необходимо реализовать базу данных с таблицей products, которая бы включала в себе всю информацию о товарах.

В таблице 3 представлены атрибуты таблицы products.

Изм.	Лист	№ докум.	Подпись	Дата

Таблица 3 – Атрибуты таблицы products

Имя столбца	Тип данных	Описание	Пример
id	INTEGER	Идентификатор товара	1
name	TEXT	Название товара	Iphone 16
description	TEXT	Краткое описание товара	Флагманский смартфон
fullDescription	TEXT	Полное описание товара	Honor Magic 4 Pro предлагает потрясающее качество на любой вкус
price	INTEGER	Цена товара	9789
imageUrl	TEXT	Ссылка на изображение товара	https://img.mvideo.ru/Pdb/400104496b.jpg
tags	TEXT	Теги для дальнейшего поиска товаров	телефоны
deliveryTime	INTEGER	Время доставки товара	3
brand	TEXT	Бренд товара	Philips
characteristics	TEXT	Характеристики товара	Процессор: AMD Zen 2 Графика: AMD RDNA

1.6.4 Проектирование графического интерфейса

На рисунке 19 представлена UML–диаграмма прецедентов.

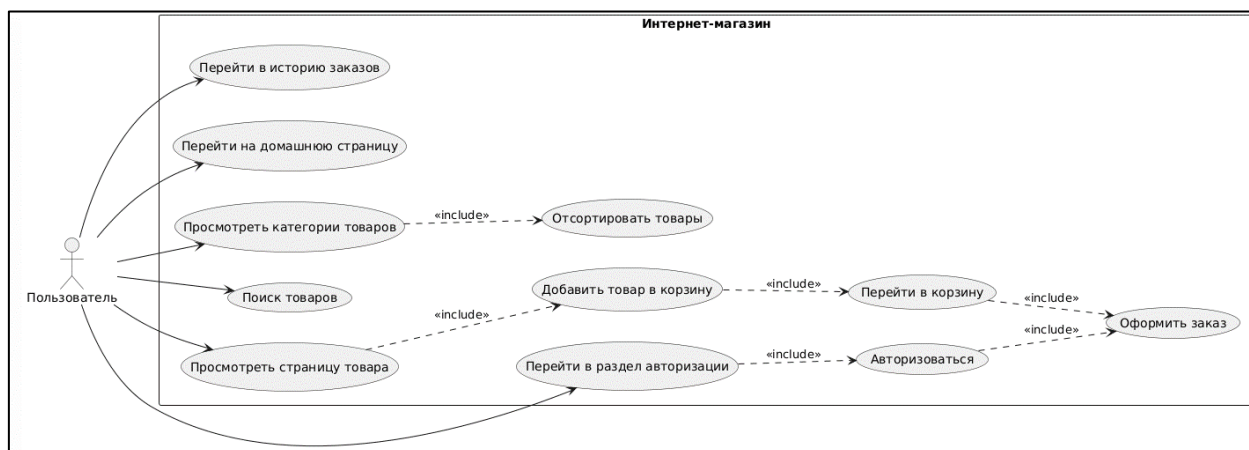


Рисунок 19 – UML – диаграмма прецедентов

При первом взаимодействии пользователя с представлением веб-приложения он может совершить несколько действий:

- осуществить поиск товаров;
- перейти в корзину, из которой он может оформить заказ;
- перейти в историю заказов;
- перейти на страницу товара, из которой можно добавить товар в корзину;
- перейти в раздел авторизации и авторизоваться;
- выбрать категорию интересующих товаров, а после отсортировать их;
- перейти на домашнюю страницу;
- добавить в корзину товар.

1.7 Реализация приложения

1.7.1 Создание стилей

Для стилей разных частей сайта мы используем CSS.

Перечень созданных css файлов:

- App.css устанавливает основные правила во всем приложении, чтобы был один вид;
- Cart.css отвечает за отображение корзины;
- CategoryPage.css делает заголовки сортировки;

Изм.	Лист	№ докум.	Подпись	Дата

- Checkout.css показывает, как выглядит страница для заказов;
- Product.css отвечает за корректное отображение товаров;
- ProductDetail.css делает страницу для каждого товара с подробной информацией;
- Loading.css – первый экран загрузки;
- ImageSlider.css отвечает за отображение слайдеров на главной странице;
- Header.css оформляет заголовок приложения.

1.7.2 Реализация сервера

Сервер реализован с помощью Node.js и Express.js.

Для начала мы импортируем все необходимые библиотеки для корректной работы сервера:

```
const express = require('express');
const mongoose = require('mongoose');
const cors = require('cors');
const bcrypt = require('bcrypt');
const jwt = require('jsonwebtoken');
```

Добавляется:

- express – фреймворк для создания веб-приложений на Node.js;
- mongoose – библиотеку для работы с MongoDB;
- cors – механизм, позволяющий API принимать запросы от иных доменов;
- bcrypt – библиотека, предназначенная для хеширования паролей;
- jsonwebtoken – библиотека для проверки и создания JSON WebTokens.

Далее мы устанавливаем порт 5000, на котором будет запущен сервер и подключаемся к базе данных MongoDB с использованием Mongoose:

```
const app = express();
```

```
const PORT = 5000;
const JWT_SECRET = 'jwt_secret_key';

mongoose.connect('mongodb://localhost:27017/db', {});
```

Создаем структуру пользователя, включающую в себя его имя и хеш пароля, а также структуру самого продукта со всеми данными о нем:

```
const User = mongoose.model('User', UserSchema);

const ProductSchema = new mongoose.Schema({
  id: Number,
  name: String,
  price: Number,
  description: String,
  imageUrl: String,
  category: String,
  fullDescription: String,
  tags: [String],
  deliveryTime: Number,
  brand: String,
  characteristics: String,
});
```

Создаем модель Product на основе ProductSchema, которая связывается с коллекцией в базе данных. Подключаем функции для обработки HTTP-запросов:

```
const Product = mongoose.model('Product', ProductSchema);

app.use(cors());
app.use(express.json());
```

Далее реализуется возможность регистрации пользователя в нашем веб-приложении. После ввода пользователем его имени и пароля в базу данных обновляется, а именно записывается его имя и хешированный пароль, если же такой пользователь уже существует, то возвращается ошибка:

```
app.post('/api/register', async (req, res) => {
  const { username, password } = req.body;
```

```

    const passwordHash = await bcrypt.hash(password, 10);
    try {
      const user = new User({ username, passwordHash });
      await user.save();
      res.status(201).json({ message: 'User created' });
    } catch (e) {
      res.status(400).json({ error: 'Username already exists' });
    }
  });

```

Пользователь имеет возможность авторизации, для этого он вводит свое имя и пароль, после идет проверка имени и пароля. Если при авторизации возникают проблемы, то будет выдана ошибка:

```

app.post('/api/login', async (req, res) => {
  const { username, password } = req.body;
  const user = await User.findOne({ username });
  if (!user) return res.status(400).json({ error: 'Invalid
  username or password' });

  const isValidPassword = await bcrypt.compare(password,
  user.passwordHash);
  if (!isValidPassword) return res.status(400).json({ error:
  'Invalid username or password' });

  const token = jwt.sign({ id: user._id, username:
  user.username }, JWT_SECRET, { expiresIn: '1h' });
  res.json({ token });
});

```

Получаем список всех продуктов из нашей базы данных. Запускаем сервер на указанном порту:

```

app.get('/api/products', async (req, res) => {
  const products = await Product.find();
  res.json(products);
});

app.listen(PORT, () => {
  console.log(`Server started on http://localhost:${PORT}`);
});

```

1.7.3 Реализация базы данных

В нашем веб-приложении реализована локальная база данных для интернет-магазина электроники с использованием MongoDB Compass–

					ПГУ 1.090301.8.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		35

инструмента для работы с MongoDB. Выбор нереляционной базы данных облегчает управление информацией и оптимизирует разработку приложения.

Для начала устанавливается npm и Node.js и создается директория приложения.

Далее оформляется структура данных в MongoDB Compass, вместо таблиц и строк создаются коллекции, в которых хранятся данные о товарах и пользователях. В коллекции товаров содержится свой уникальный номер, название, краткое и подробное описание, цену, ссылку на изображение, теги, информацию о времени доставки, бренд и технические характеристики. Для корректной работы базы данных задействованы функции чтения и записи информации.

Затем мы создаем API-сервер с использованием Express.js. В файлах App.js и server.js хранится вся логика обработки HTTP-запросов. Сервер начинает работу вместе с подключением к базе данных, это позволяет обновлять и получать данные в динамическом режиме.

1.7.5 Реализация компонентов

Компонент App является главным компонентом приложения. Он отвечает за маршрутизацию, вся структура приложения описана в нем.

```
import { BrowserRouter, Routes, Route } from 'react-router-dom';

const App = () => {
  const [cart, setCart] = useState([]);
  const [products, setProducts] = useState([]);
  const [loading, setLoading] = useState(true);
  const [user, setUser] = useState(null);

  useEffect(() => {
    const fetchProducts = async () => {
      try {
        const response = await
axios.get('http://localhost:5000/api/products');
setProducts(response.data);
      } catch (error) {
console.error('Ошибка при загрузке продуктов:', error);
      } finally {
```

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

Лист

36

```

setLoading(false);
    }
};

```

```

fetchProducts();
    }, []);

```

Компонент Header отвечает за навигацию по приложению, в нем же содержатся ссылки на корзину, историю заказов, авторизацию и перехода на главную страницу.

```

const Header = ({ cartItems, products, user, setUser }) => {
  const [searchTerm, setSearchTerm] = useState('');
  const [filteredProducts, setFilteredProducts] = useState([]);
  const [isModalOpen, setIsModalOpen] = useState(false);
  const [username, setUsername] = useState('');
  const [password, setPassword] = useState('');
  const [isDropdownOpen, setIsDropdownOpen] = useState(false);
  const [loginError, setLoginError] = useState('');
  const navigate = useNavigate();

  const handleSearchChange = (event) => {
    const value = event.target.value;
    setSearchTerm(value);

    const filtered = products.filter(product => {
      const name = product.name ? product.name.toLowerCase() :
      '';
      const category = product.category ?
      product.category.toLowerCase() : '';
      const brand = product.brand ? product.brand.toLowerCase()
      : '';
      return name.includes(value.toLowerCase()) ||
      category.includes(value.toLowerCase()) ||
      brand.includes(value.toLowerCase());
    });
    setFilteredProducts(filtered);
    setIsDropdownOpen(value.length > 0 && filtered.length > 0);
  };

```

Компонент HomePage отвечает за главную страницу приложения, на которой отображается весь ассортимент товаров.

```

const HomePage = ({ products, onAddToCart }) => {
  return (
    <div className="home-page">

```

```

<ImageSlider />
<h2 className="section-title">Каталог товаров</h2>
<div className="category-icons">
<Link to="/category/чайники" className="category-icon">
<div className="icon-container">

<span>Чайники</span>
</div>
</Link>
...
</div>
<h2 className="section-title">Хиты продаж</h2>
<ProductList products={products} onAddToCart={onAddToCart} />
</div>
);
};

```

Компонент ProductList загружает список продуктов, получает данные о каждом продукте с сервера.

```

const ProductList = ({ products = [], onAddToCart }) => {
  const navigate = useNavigate();
  const handleProductClick = (productId) => {
    navigate(`/product/${productId}`);
  };
};

```

Компонент ProductDetail отвечает за отображение и наполнение отдельной страницы каждого товара. Эта страница отображает полное описание товара и его характеристики, помимо основных данных о товаре.

```

const ProductDetails = ({ products, onAddToCart }) => {
  const { id } = useParams();
  const navigate = useNavigate();
  const product = products.find(item => item.id ===
    parseInt(id));

  if (!product) {
    return <div>Товар не найден</div>;
  }

  const handleAddToCart = () => {
    onAddToCart(product);
  };
};

```

```
const handleBack = () => {
  navigate(-1);
};
```

Компонент Cart отвечает за отображение корзины приложения. Все товары, находящиеся в ней, могут быть удалены, а сама корзина очищена.

```
const Cart = ({ cartItems, onRemoveFromCart, onClearCart }) => {
  const navigate = useNavigate();
  const totalItems = cartItems.length;
  const totalPrice = cartItems.reduce((total, item) => total +
    item.price, 0);

  const [sortCriteria, setSortCriteria] = useState('name');
  const [pulseId, setPulseId] = useState(null);
  const [animateAdd, setAnimateAdd] = useState(false);
  const handleRemove = (productId) => {
    if (window.confirm('Вы уверены, что хотите удалить этот
товар из корзины?')) {
      onRemoveFromCart(productId);
    }
  };
};
```

1.7.6 Руководство пользователя

После запуска веб-приложения пользователь попадает на домашнюю страницу. Страница содержит в верхней части окно поиска товаров и кнопки навигации, такие как корзина, история заказов, возврат на домашнюю страницу и авторизация. Поля для поиска товаров и кнопки навигации представлены на рисунке 20. Главный экран приложения представлен на рисунке 21.



Рисунок 20 – Поля для поиска товаров и кнопки навигации

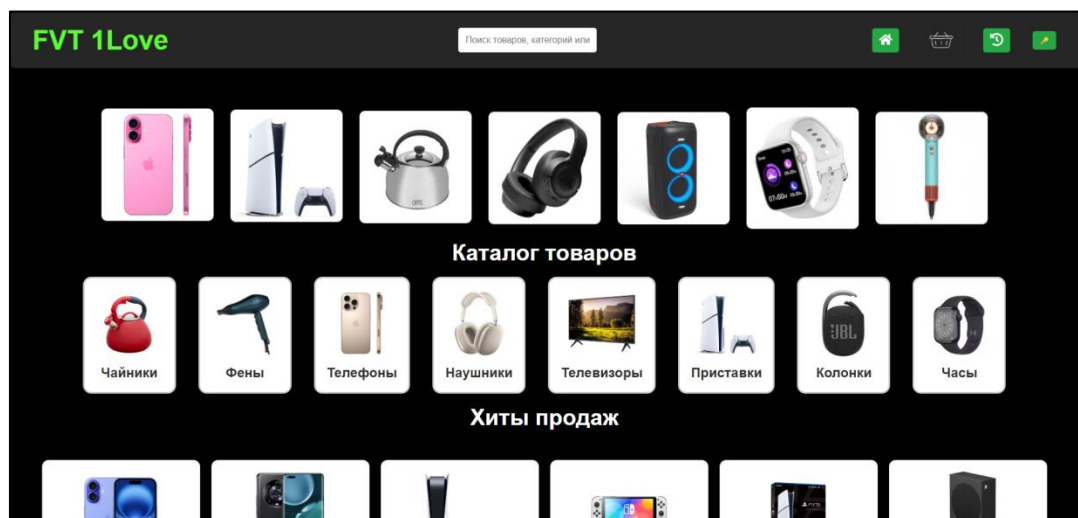


Рисунок 21 – Главный экран веб–приложения

Пользователю предлагается найти интересующий товар самому или ознакомиться с ассортиментом на главном экране. Поиск товаров представлен на рисунке 22. Ассортимент представлен на рисунке 23.

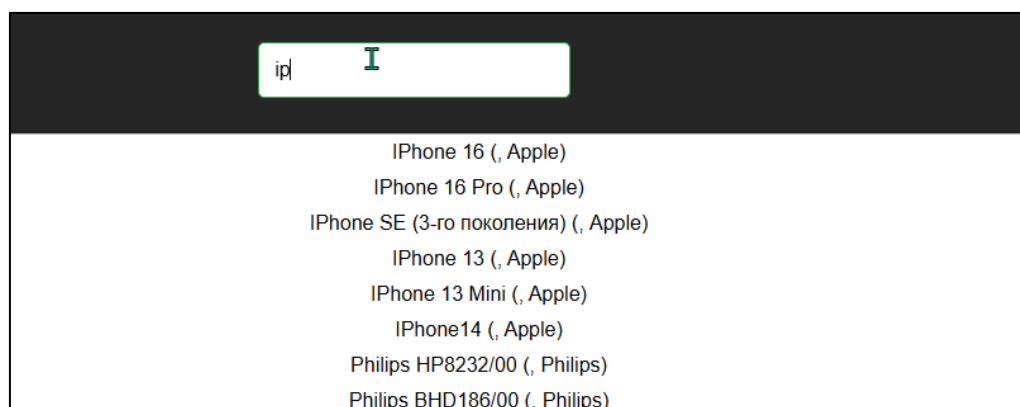


Рисунок 22 – Поиск товаров

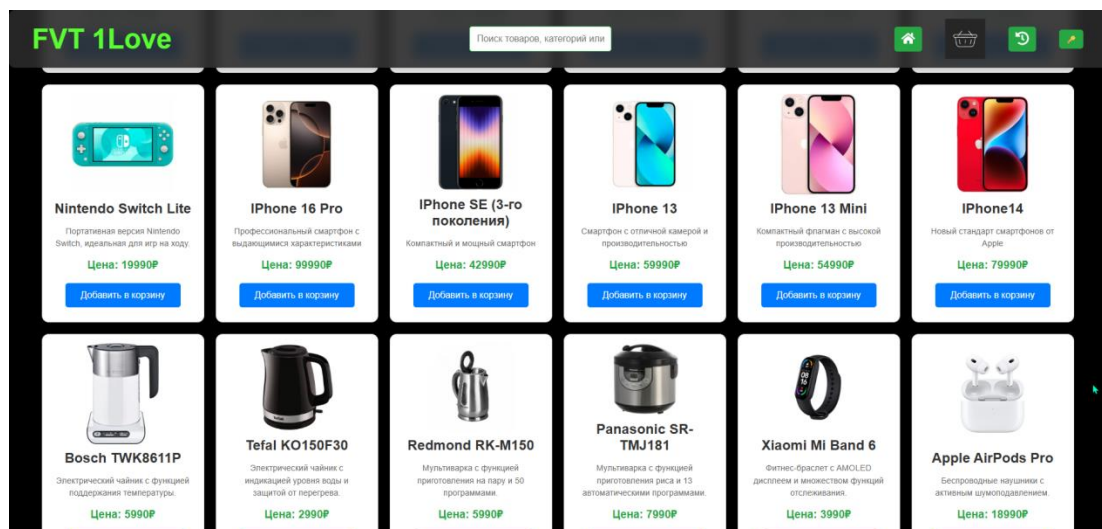


Рисунок 23 – Ассортимент товаров

Для того, чтобы посмотреть все товары одного типа пользователю предлагается выбрать конкретную категорию из каталога товаров. Каталог товаров представлен на рисунке 24. Товары категории «Телефоны» представлены на рисунке 25.



Рисунок 24 – Каталог товаров

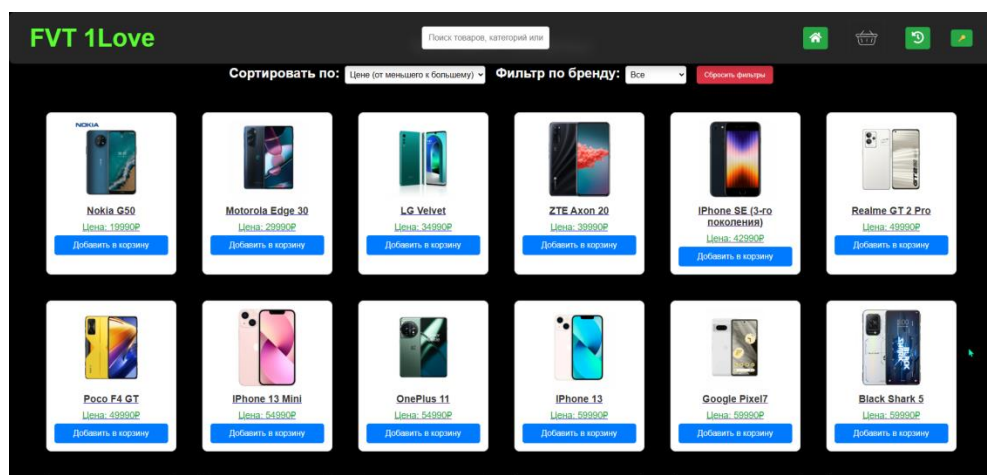


Рисунок 25 – Товары категории «Телефоны»

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

Лист
41

Для более удобного поиска по товарам пользователь может отсортировать товары по цене, времени доставки и бренду. Если же фильтры больше не нужны, то их всегда можно сбросить с помощью специально кнопки. Сортировка по цене и бренду представлена на рисунке 26. Кнопка сброса фильтров представлена на рисунке 27.

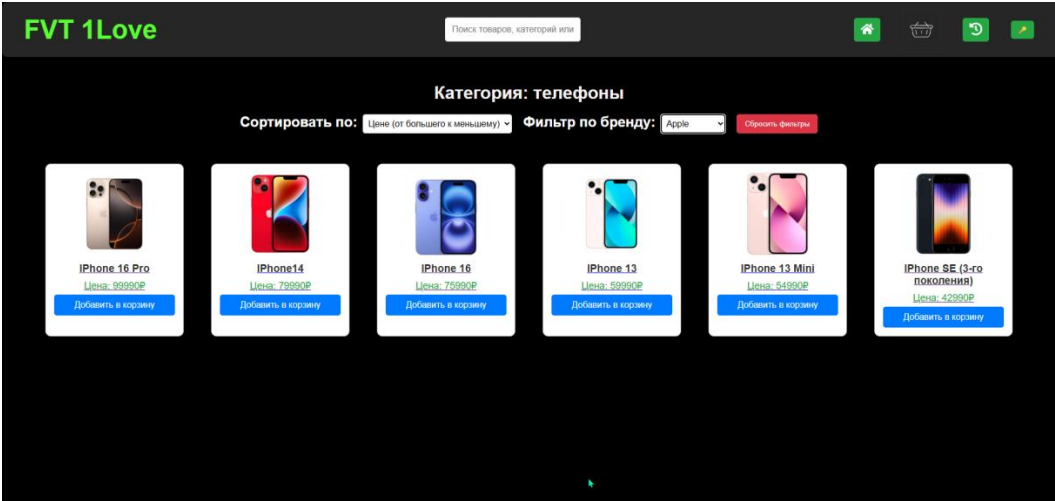


Рисунок 26 – Сортировка по цене и бренду

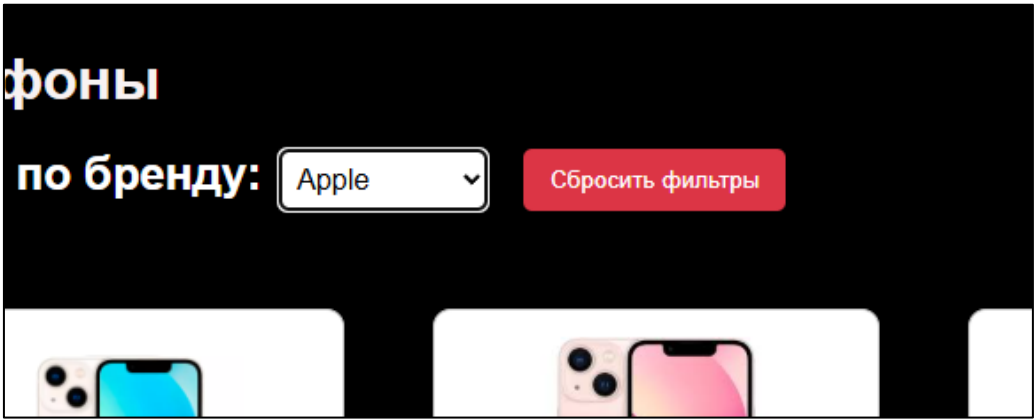


Рисунок 27 – Кнопка сброса фильтров

Для того, чтобы пользователь смог увидеть полную информацию о товаре, он должен нажать на иконку товара и перейти на страницу с полной информацией о нем. Если пользователь захочет вернуться, то он может нажать кнопку «Назад». Окно с полной информацией о товаре представлено на рисунке 26. Кнопка назад представлена на рисунке 27.

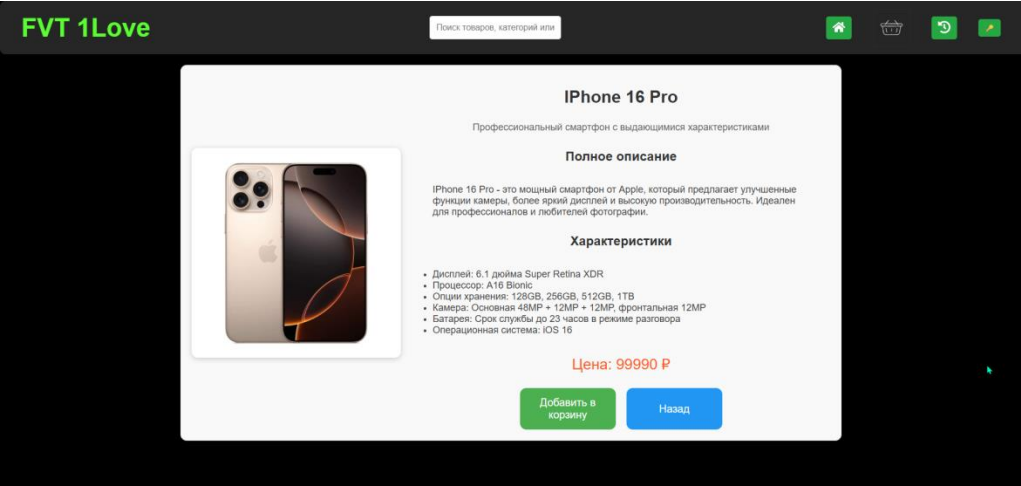


Рисунок 26 – Окно с полной информацией о товаре

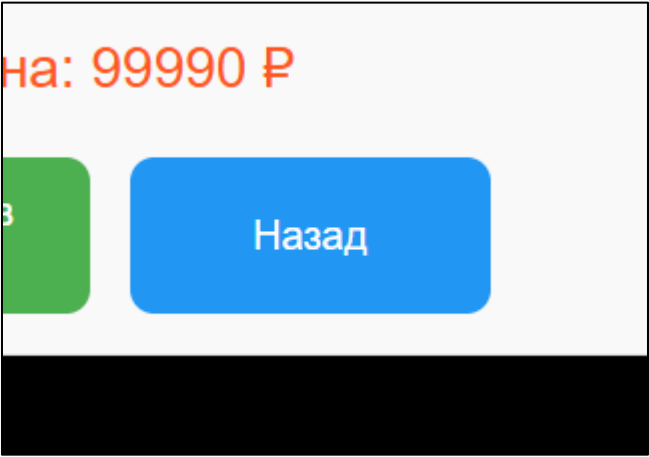


Рисунок 27 – Кнопка «Назад»

Пользователь имеет возможность добавить товар в корзину на главном экране приложения, из окна определенной категории товаров, на странице самого товара. Кнопки добавления в корзину представлены на рисунках 28,29,30.

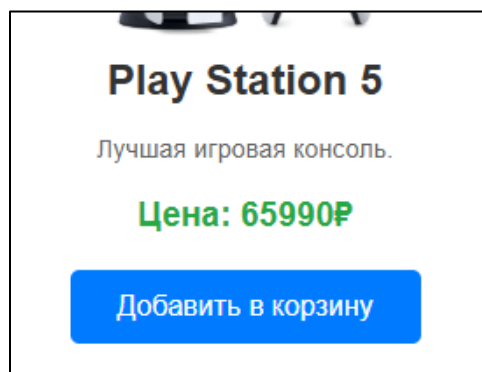


Рисунок 28 – Кнопка «Добавить в корзину»

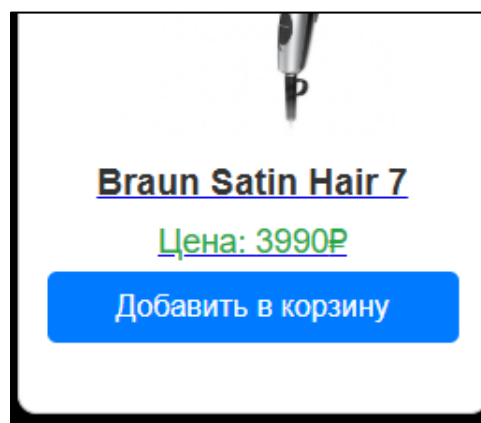


Рисунок 29 – Кнопка «Добавить в корзину»

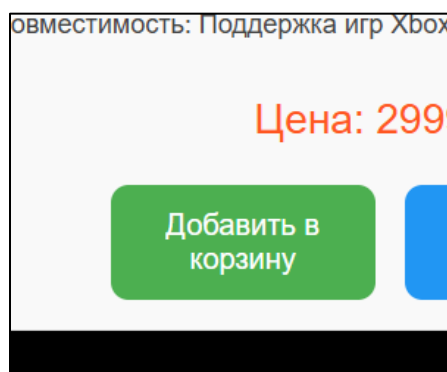


Рисунок 30 – Кнопка «Добавить в корзину»

Пользователь может посмотреть содержимое корзины после взаимодействия с кнопкой корзины на главном экране. В корзине он видит количество товаров и их итоговую цену. Он может отсортировать товары в корзине по имени и цене. Кнопки удалить и очистить корзину позволяют

пользователю удалить либо конкретный товар либо очистить корзину полностью. Окно корзины представлено на рисунке 31. Сортировка в корзине по имени представлена на рисунке 32. Удаление товара и очистка корзины представлены на рисунках 33,34.

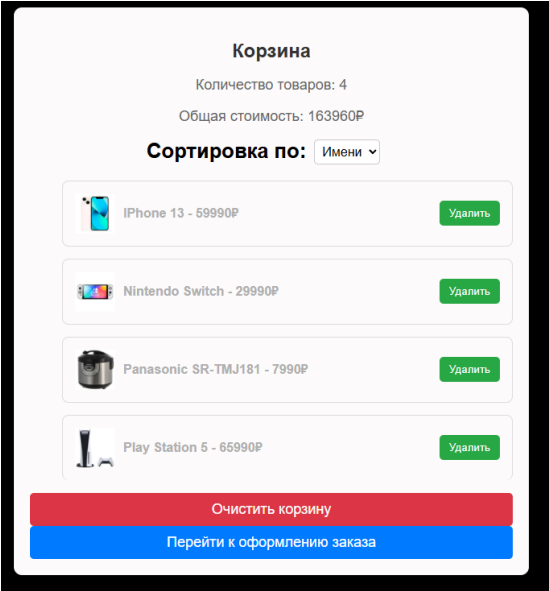


Рисунок 31 – Окно корзины

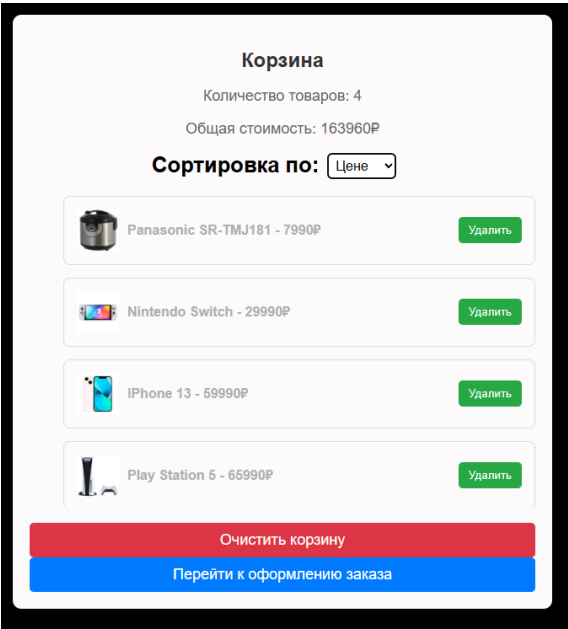


Рисунок 32 – Сортировка по цене в корзине

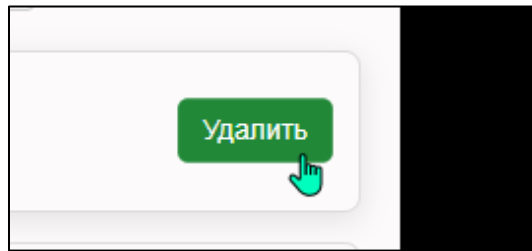


Рисунок 33 – Кнопка удаления товара из корзины

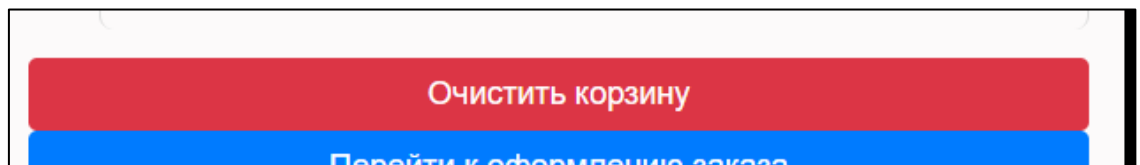


Рисунок 34 – Кнопка очистки корзины

Пользователь после выбора всего, что его интересовало может перейти к оформлению заказа, для этого он должен нажать на соответствующую кнопку. Кнопка перехода к оформлению заказа представлена на рисунке 35. Окно оформления заказа представлено на рисунке 36.

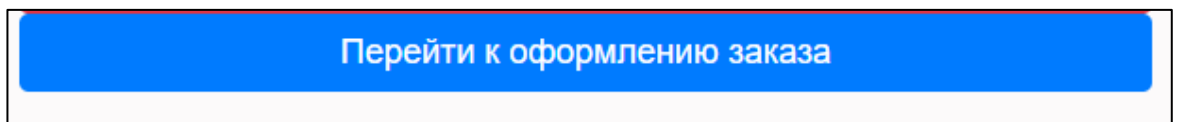


Рисунок 35 – Кнопка оформления заказа

Заполните следующие поля для оформления заказа

Город:

Почтовый индекс:

Адрес:

Почтовый ящик:

Номер телефона:

Способ оплаты:

☒
Наличными

☐
Картой

Оформить заказ

Рисунок 36 – Окно оформления заказа

Пользователю предлагается заполнить свои данные такие, как город, почтовый индекс, адрес, почтовый ящик, номер телефона и номер карты, в случае если выбран метод оплаты картой. После успешного заполнения вы можете оформить свой заказ нажав на соответствующую кнопку ниже. Поля ввода информации представлены на рисунке 37. Кнопка оформления заказа на рисунке 38.

Город:	Пенза
Почтовый индекс:	442315
Адрес:	Пр-кт Победы 16, кв-53
Почтовый ящик:	pgu1love@mail.com
Номер телефона:	8 (841) 266-64-19
Способ оплаты:	☐ Наличными ☒ Картой
Номер карты:	

Рисунок 37 – Поля ввода информации

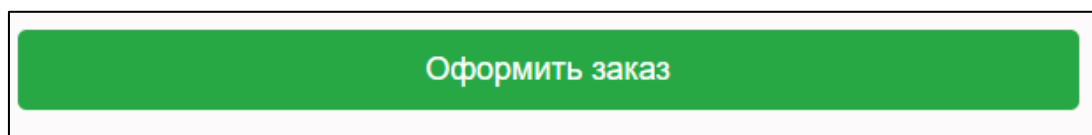


Рисунок 38 – Кнопка оформления заказа

Когда пользователь оформляет заказ ему приходит уведомление об этом и открывается окно истории заказов, где он может ознакомиться с номером заказа, ценой, временем доставки и заказанными товарами. Уведомление об успешном оформлении заказа представлено на рисунке 39. Окно истории заказов представлено на рисунке 40.

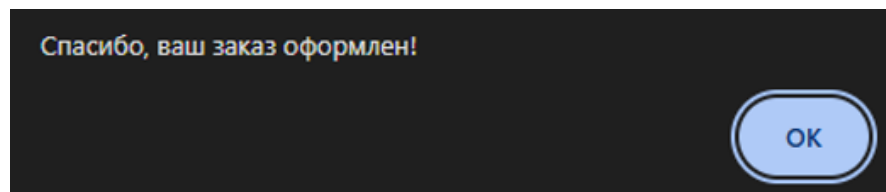


Рисунок 39 – Уведомление об успешном оформлении заказа



Рисунок 40 – Окно истории заказов

2 Охрана труда

Условия труда, рабочее место и трудовой процесс в силу ст. 25 Федерального закона от 30.03.1999 № 52–ФЗ «О санитарно–эпидемиологическом благополучии населения» не должны оказывать вредное воздействие на работников. Требования к обеспечению безопасных для них условий труда устанавливаются санитарными правилами и иными нормативными правовыми актами Российской Федерации. Юридические лица и индивидуальные предприниматели при этом обязаны осуществлять санитарно–противоэпидемические (профилактические) мероприятия по обеспечению безопасных для сотрудников условий труда и выполнению требований санитарных правил и иных нормативных правовых актов РФ, в частности к производственным процессам и технологическому оборудованию, организации рабочих мест в целях предупреждения травм, профессиональных заболеваний и заболеваний, связанных с условиями труда.

Условиями труда согласно ст. 209 ТК РФ является совокупность факторов производственной среды и трудового процесса, оказывающих влияние на работоспособность и здоровье работника. Вредным же производственным фактором в силу упомянутой ст. 209 ТК РФ признается фактор, воздействие которого на работника может привести к его заболеванию [13].

Использование персональных электронных вычислительных машин (ПЭВМ) в помещениях, лишённых естественного освещения, допускается только при наличии обоснованных расчетов, подтверждающих соответствие установленным нормам и гарантирующих безопасность для здоровья сотрудников.

Как естественное, так и искусственное освещение должны строго соответствовать требованиям действующих нормативных актов. Окна в

					ПГУ 1.090301.8.001 ПЗ	Лист
						50
Изм.	Лист	№ докум.	Подпись	Дата		

комнатах с вычислительной техникой предпочтительно ориентируют на север или северо–восток и оснащают регулируемыми шторами или жалюзи для контроля уровня света.

В связи с недостатком естественного освещения, в таких помещениях обычно устанавливают системы искусственного равномерного освещения общего типа. Особые параметры освещения рабочих мест с ПЭВМ регламентируются разделом VI Санитарных правил.

В помещениях производственного и административного назначения, где основная работа связана с документами, рекомендуется применять комбинированные системы освещения. Они включают общее освещение и дополнительные локальные светильники, направленные на рабочую поверхность с документами.

Уровень освещенности на рабочем столе должен находиться в диапазоне 300–500 люкс. Локальные светильники не должны быть чрезмерно яркими и обязаны исключать блики на экране — освещенность его поверхности не должна превышать 300 люкс. Кроме того, настольные лампы необходимо оснащать непрозрачными отражателями с защитным углом не менее 40 градусов для обеспечения комфортного и безопасного освещения.

Конструкция рабочего стола должна быть продумана таким образом, чтобы обеспечить оптимальное размещение всего необходимого оборудования с учётом его количества, особенностей конструкции и характера выполняемых задач. Допускается использование различных моделей рабочих столов, соответствующих современным эргономическим требованиям. Рекомендуемые размеры поверхности стола для работы с ПЭВМ составляют: ширина от 800 до 1400 мм и глубина 800–1000 мм, при фиксированной высоте 725 мм.

Монитор следует размещать на расстоянии 60–70 см от глаз пользователя, но не ближе 50 см, чтобы обеспечить комфортное восприятие текста и изображений. Клавиатура должна находиться на расстоянии 100–300

					ПГУ 1.090301.8.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		51

мм от края стола, обращённого к пользователю, или на специальной регулируемой по высоте поверхности, отделённой от основной столешницы.

Стул, используемый в рабочем месте, должен способствовать поддержанию правильной осанки и позволять изменять положение для снижения напряжения в мышцах спины и шейно–плечевой области. Оптимальным вариантом является подъемно–поворотное кресло с возможностью регулировки высоты и угла наклона спинки. Все параметры должны регулироваться независимо, легко и надежно фиксироваться.

Для повышения комфорта рабочего места пользователя ПЭВМ рекомендуется использовать подставку для ног. Она должна иметь ширину не менее 300 мм, глубину не менее 400 мм и возможность регулировки высоты до 150 мм, а также угла наклона опорной поверхности до 20 градусов. Желательно, чтобы подставка имела рифлёную поверхность и бортик высотой 10 мм по переднему краю (разделы IX и X Санитарных правил).

Офисное оборудование, включая компьютеры, относится к электрическим устройствам, поэтому работодатель обязан обеспечить электробезопасность сотрудников. Для предотвращения коротких замыканий, пожаров и электротравм помещения с рабочими местами, оснащёнными компьютерами, должны быть оборудованы защитным заземлением или занулением в соответствии с установленными техническими нормами эксплуатации.

Согласно статье 212 Трудового кодекса РФ, работодатель несёт ответственность за создание безопасных условий труда и охрану здоровья работников. В частности, он обязан разработать и утвердить правила и инструкции по охране труда с учётом мнения выборного органа первичной профсоюзной организации или другого уполномоченного работниками органа. Методические рекомендации по разработке таких инструкций, утверждённые Минтрудом России 13 мая 2004 года, предусматривают их

					ПГУ 1.090301.8.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		52

составление на основе межотраслевых или отраслевых типовых инструкций по охране труда (а при их отсутствии — на основе соответствующих правил), требований безопасности, отражённых в эксплуатационной и ремонтной документации производителей оборудования, а также в технологической документации организации с учётом специфики производства. Эти требования должны быть адаптированы к конкретной должности, профессии или виду выполняемой работы.

Работодатель организует проверку и обновление инструкций по охране труда для сотрудников, при этом пересмотр должен проводиться не реже одного раза в пять лет.

					ПГУ 1.090301.8.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		53

Заключение

В процессе разработки веб-приложения были учтены все ключевые требования, направленные на создание современного, удобного и безопасного сервиса для пользователей. Особое внимание было уделено обеспечению простого и интуитивно понятного взаимодействия: реализована логичная структура навигации, удобный поиск товаров и функциональная корзина, что позволяет пользователям быстро находить нужные позиции и оформлять заказы без лишних усилий.

Реализован механизм шифрования паролей для обеспечения безопасности пользователей. Процесс оформления заказа организован максимально просто и удобно: пользователям предлагаются различные способы оплаты, что делает процесс покупки гибким и комфортным.

Техническая реализация проекта основана на использовании актуальных веб-технологий. Клиентская часть разработана с применением HTML, CSS и JavaScript с использованием фреймворка ReactJS, что обеспечивает адаптивность интерфейса и высокую отзывчивость на различных устройствах. Серверная часть построена с использованием Node.js и Express.js, а взаимодействие между клиентом и сервером осуществляется через RESTful API. Для хранения информации о товарах используется нереляционная база данных, обеспечивающая масштабируемость и высокую производительность системы.

Таким образом, разработанное веб-приложение полностью соответствует поставленным задачам и требованиям: оно сочетает в себе удобство использования, надёжную защиту данных, современный технологический стек и готовность к дальнейшему развитию.

					ПГУ 1.090301.8.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		54

Список использованных источников

1. Официальная информация «DNS». [Электронный ресурс] / Режим доступа: <https://ru.wikipedia.org/wiki/DNS>, свободный (дата обращения 07.05.2025).
2. Официальная информация «М.Видео». [Электронный ресурс] / Режим доступа: <https://www.mvideo.ru/info/company>, свободный (дата обращения 07.05.2025).
3. Официальная информация «Ситилинк». [Электронный ресурс] / Режим доступа: <https://www.citilink.ru/about/citilink>, свободный (дата обращения 07.05.2025).
4. Официальная информация «Мегамаркет». [Электронный ресурс] / Режим доступа: <https://ru.wikipedia.org/wiki/Мегамаркет>, свободный (дата обращения 07.05.2025).
5. Официальная информация «Ozon». [Электронный ресурс] / Режим доступа: <https://ru.wikipedia.org/wiki/Ozon>, свободный (дата обращения 07.05.2025).
6. Официальная информация «Wildberries». [Электронный ресурс] / Режим доступа: <https://www.retail.ru/rbc/tradingnetworks/wildberries/>, свободный (дата обращения 07.05.2025).
7. Макконнелл, С. Совершенный код. – М.: Издательство "БХВ–Петербург", 2022. –896 с.
8. Флэнаган, Д. JavaScript. Полное руководство. – М.: Издательство "Диалектика-Вильямс", 2021. –720 с.
9. Дакетт, Д. HTML и CSS. Разработка и дизайн веб-сайтов – М.: Издательство "Эксмо", 2024. –480 с.
10. Официальная информация об интегрированных средах разработки. [Электронный ресурс] / Режим доступа: https://ru.wikipedia.org/wiki/Категория:Интегрированные_среды_разработки, свободный (дата обращения 11.05.2025).

11. Основные сведения MySQL. [Электронный ресурс] / Режим доступа: <https://servergate.ru/articles/mysql>, свободный (дата обращения 15.05.2025).

12. Описание шаблона проектирования MVC. [Электронный ресурс] / Режим доступа: https://pub.dev/documentation/mvc_pattern/latest/, свободный (дата обращения 13.05.2025).

13. Инструкция по охране труда при работе с ЭВМ. [Электронный ресурс] / Режим доступа: https://kpfu.ru/portal/docs/F_1086113514/1815.pdf/, свободный (дата обращения 13.05.25).

					<i>ПГУ 1.090301.8.001 ПЗ</i>	Лист
						56
Изм.	Лист	№ докум.	Подпись	Дата		

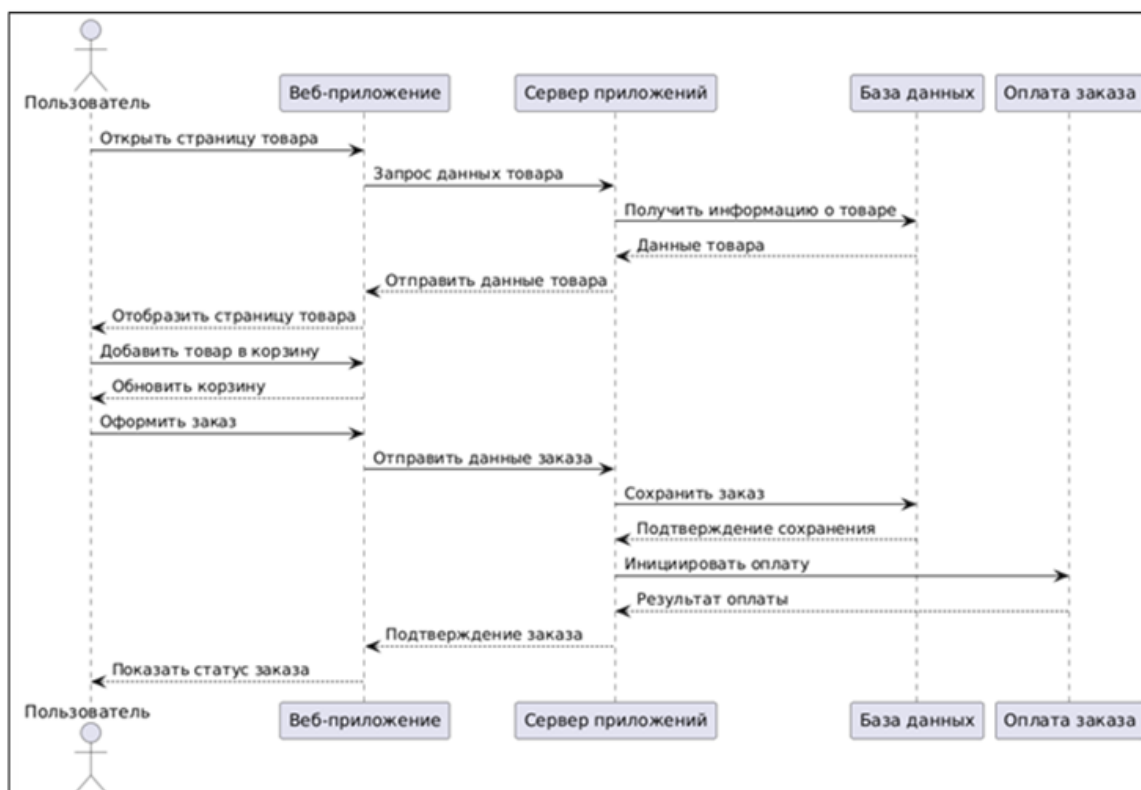
Приложение А

(Обязательное)

Интернет-магазин электроники

UML диаграммы

					<i>ПГУ 1.090301.8.001 ПЗ</i>	Лист
						57
Изм.	Лист	№ докум.	Подпись	Дата		



ПГУ 1.090301.9.005 Д14									
Вид	Тип	Инициатор	Получатель	Дата	Инициатор: Инициатор (ИМ) и получатель				
Инициатор	Инициатор	Инициатор	Инициатор	Инициатор	Инициатор: Инициатор (ИМ) и получатель				
Инициатор	Инициатор	Инициатор	Инициатор	Инициатор	Инициатор: Инициатор (ИМ) и получатель				
Инициатор	Инициатор	Инициатор	Инициатор	Инициатор	Инициатор: Инициатор (ИМ) и получатель				
Инициатор	Инициатор	Инициатор	Инициатор	Инициатор	Инициатор: Инициатор (ИМ) и получатель				
Инициатор	Инициатор	Инициатор	Инициатор	Инициатор	Инициатор: Инициатор (ИМ) и получатель				
Инициатор	Инициатор	Инициатор	Инициатор	Инициатор	Инициатор: Инициатор (ИМ) и получатель				
Инициатор	Инициатор	Инициатор	Инициатор	Инициатор	Инициатор: Инициатор (ИМ) и получатель				
Инициатор	Инициатор	Инициатор	Инициатор	Инициатор	Инициатор: Инициатор (ИМ) и получатель				

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

Лист
60

Приложение Б

(Обязательное)

Интернет-магазин электроники

Листинг программы

					ПГУ 1.090301.8.001 ПЗ	Лист
						63
Изм.	Лист	№ докум.	Подпись	Дата		

Файл Cart.js

```
import React, { useState } from 'react';
import { useNavigate } from 'react-router-dom';
import './Cart.css';

const Cart = ({ cartItems, onRemoveFromCart, onClearCart }) => {
  const navigate = useNavigate();
  const totalItems = cartItems.length;
  const totalPrice = cartItems.reduce((total, item) => total +
item.price, 0);

  const [sortCriteria, setSortCriteria] = useState('name');
  const [pulseId, setPulseId] = useState(null);
  const [animateAdd, setAnimateAdd] = useState(false);
  const handleRemove = (productId) => {
    if (window.confirm('Вы уверены, что хотите удалить этот
товар из корзины?')) {
      onRemoveFromCart(productId);
    }
  };

  const handleCheckout = () => {
    navigate('/checkout', { state: { orderDetails: { items:
cartItems } } });
  };

  const handleItemClick = (itemId) => {
    navigate(`/product/${itemId}`);
  };

  const handleRemoveClick = (itemId) => {
    handleRemove(itemId);
    setPulseId(itemId);
    setTimeout(() => {
      setPulseId(null);
    }, 600);
  };

  const handleClearCart = () => {
    if (window.confirm('Вы уверены, что хотите очистить
корзину?')) {
      onClearCart();
    }
  };

  const sortedItems = [...cartItems].sort((a, b) => {
    if (sortCriteria === 'name') {
      return a.name.localeCompare(b.name);
    } else if (sortCriteria === 'price') {
      return a.price - b.price;
    }
  });
};
```

Изм.	Лист	№ докум.	Подпись	Дата

```

    }
    return 0;
  });

  return (
    <div className="cart">
      <h2>Корзина</h2>
      <p>Количество товаров: {totalItems}</p>
      <p>Общая стоимость: {totalPrice}₽</p>

      <div className="sort-options">
        <label htmlFor="sort" style={{ color: 'black',
fontWeight: 'bold' }}>Сортировка по: </label>
        <select
          id="sort"
          value={sortCriteria}
          onChange={ (e) =>
setSortCriteria(e.target.value) }
        >
          <option value="name">Имени</option>
          <option value="price">Цене</option>
        </select>
      </div>

      {totalItems === 0 ? (
        <p>Корзина пуста</p>
      ) : (
        <>
          <ul>
            {sortedItems.map((item) => (
              <li key={item.id} className="cart-
item">
                <div className="cart-item-
content" onClick={() => handleItemClick(item.id)}>
                  <img src={item.imageUrl}
alt={item.name} className="cart-item-image" />
                  <span>{item.name} -
{item.price}₽</span>
                </div>
                <button
                  className={pulseId ===
item.id ? 'pulse' : ''}
                  onClick={() =>
handleRemoveClick(item.id) }
                >
                  Удалить
                </button>
              </li>
            ))}
          </ul>
        </>
      )}
    )
  );

```

Изм.	Лист	№ докум.	Подпись	Дата

```

        { /* Кнопка очистки корзины */ }
        <button className="clear-cart-button"
onClick={handleClearCart}>
            Очистить корзину
        </button>

        { /* Кнопка перехода к оформлению заказа */ }
        <button className="checkout-button"
onClick={handleCheckout}>
            Перейти к оформлению заказа
        </button>
    </>
    ) }
</div>
);
};

export default Cart;

```

Файл Cart.css

```

.cart {
    background: #fcfafa;
    border-radius: 10px;
    padding: 20px;
    box-shadow: 0 4px 20px rgba(0, 0, 0, 0.1);
    max-width: 600px;
    margin: 20px auto;
}

.cart h2 {
    color: #333;
    font-size: 24px;
    margin-bottom: 15px;
    text-align: center;
}

.cart p {
    font-size: 18px;
    color: #666;
    text-align: center;
}

.cart-item {
    display: flex;
    justify-content: space-between;
    align-items: center;
    margin: 15px 0;
    padding: 15px;
}

```

Изм.	Лист	№ докум.	Подпись	Дата

```

        border: 1px solid #ddd;
        border-radius: 8px;
        transition: box-shadow 0.3s;
    }

    .cart-item:hover {
        box-shadow: 0 4px 15px rgba(0, 0, 0, 0.1);
    }

    .cart-item:last-child {
        border-bottom: none;
    }

    .cart-item span {
        font-weight: bold;
    }

    button {
        background: #28a745;
        color: rgb(255, 255, 255);
        border: none;
        border-radius: 5px;
        padding: 8px 12px;
        cursor: pointer;
        transition: background 0.3s, transform 0.2s;
    }

    button:hover {
        background: #218838;
        transform: scale(1.05);
    }

    .checkout-button {
        width: 100%;
        padding: 10px;
        font-size: 18px;
        background: #007bff;
        color: white;
        border: none;
        border-radius: 5px;
        cursor: pointer;
        transition: background 0.3s, transform 0.2s;
    }

    .checkout-button:hover {
        background: #0056b3;
        transform: scale(1.05);
    }

```

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

Лист

67

```

}

.sort-label {
  color: black;
  font-weight: bold;
}

.cart-item-content {
  display: flex;
  align-items: center;
  cursor: pointer;
}

.cart-item-image {
  width: 50px;
  height: 50px;
  margin-right: 10px;
}

.clear-cart-button {
  width: 100%;
  padding: 10px;
  font-size: 18px;
  background: #dc3545;

  color: white;
  border: none;
  border-radius: 5px;
  cursor: pointer;
  transition: background 0.3s, transform 0.2s;
}

@ .clear-cart-button:hover {
  background: #c82333;

  transform: scale(1.05);
}

.add-to-cart-button {
  transition: transform 0.3s ease;
}

.add-to-cart-button.animate {
  animation: pulse 0.6s;
}

@keyframes pulse {

```

```

0% {
  transform: scale(1);
}

50% {
  transform: scale(1.1);
}

100% {
  transform: scale(1);
}
}

```

Файл CategoryList.js

```

import React from 'react';
import { Link } from 'react-router-dom';

const categories = [
  {
    name: "Телефоны",
    imageUrl: "https://example.com/phone.jpg",
    tag: "phones"
  },
  {
    name: "Чайники",
    imageUrl: "https://example.com/kettle.jpg",
    tag: "kettles"
  },
  {
    name: "Фены",
    imageUrl: "https://example.com/hairstyler.jpg",
    tag: "hairstyler"
  }
];

const CategoryList = () => {
  return (
    <div className="category-list">
      {categories.map(category => (
        <div key={category.tag} className="category-
item">
          <Link to={`/${category.tag}`}>
            <img src={category.imageUrl}
alt={category.name} />
            <h2>{category.name}</h2>
          </Link>
        </div>
      ))}
    </div>
  );
};

```

```
};

export default CategoryList;
```

Файл CategoryPage.js

```
import React, { useState } from 'react';
import { useParams, Link } from 'react-router-dom';
import './CategoryPage.css';

const CategoryPage = ({ products, onAddToCart }) => {
  const { tag } = useParams();
  const filteredProducts = products.filter(product =>
product.tags.includes(tag));
  const [sortCriteria, setSortCriteria] =
useState('priceAsc');
  const [selectedBrand, setSelectedBrand] = useState('all');
  const handleBrandChange = (event) => {
    setSelectedBrand(event.target.value);
  };

  const sortProducts = (products) => {
    return products
      .filter(product => selectedBrand === 'all' ||
product.brand === selectedBrand)
      .sort((a, b) => {
        if (sortCriteria === 'priceAsc') {
          return a.price - b.price;
        } else if (sortCriteria === 'priceDesc') {
          return b.price - a.price;
        } else if (sortCriteria === 'deliveryTime') {
          return a.deliveryTime - b.deliveryTime;
        } else if (sortCriteria === 'name') {
          return a.name.localeCompare(b.name);
        }
        return 0;
      });
  };

  const sortedProducts = sortProducts(filteredProducts);
  const uniqueBrands = Array.from(new
Set(filteredProducts.map(product => product.brand)));
  const resetFilters = () => {
    setSelectedBrand('all');
    setSortCriteria('priceAsc');
  };

  return (
    <div className="category-page">
```

```

        <h2 className="section-title1">Категория: {tag}</h2>

        <div className="sort-options">
            <label htmlFor="sortBy" className="sort-label">Сортировать по:</label>
            <select id="sortBy" onChange={ (e) =>
                setSortCriteria(e.target.value) }>
                <option value="priceAsc">Цене (от меньшего к
                большему)</option>
                <option value="priceDesc">Цене (от большего
                к меньшему)</option>
                <option value="deliveryTime">Времени
                доставки</option>
                <option value="name">Имени</option>
            </select>

            <label htmlFor="brandFilter" className="sort-label">Фильтр по бренду:</label>
            <select id="brandFilter"
                onChange={handleBrandChange}>
                <option value="all">Все</option>
                {uniqueBrands.map(brand => (
                    <option key={brand}
                    value={brand}>{brand}</option>
                ))}
            </select>

            { /* Кнопка сброса фильтров */ }
            <button className="reset-filters-btn"
                onClick={resetFilters}>Сбросить фильтры</button>
        </div>

        <div className="product-list">
            {sortedProducts.length === 0 ? (
                <p>Товары не найдены.</p>
            ) : (
                sortedProducts.map((product) => (
                    <div key={product.id}
                    className="product-card">
                        <Link to={` /product/${product.id}`}>
                            <img src={product.imageUrl}
                            alt={product.name} className="product-image" />
                            <h3 className="product-name">{product.name}</h3>
                            <p className="product-price">Цена: {product.price}Р</p>
                            </Link>
                            <button className="add-to-cart-btn"
                                onClick={ () => onAddToCart(product) }>Добавить в корзину</button>
                        </div>
                    ) )
            )
        }
    </div>

```

```

        })
      </div>
    </div>
  );
};

export default CategoryPage;

```

Файл CategoryPage.css

```

.category-page {
  text-align: center;
}

.section-title1 {
  font-size: 2em;
  font-weight: bold;
  margin-bottom: 20px;
  color: #f8f2f2;
}

.sort-options {
  display: flex;
  justify-content: center;
  margin-bottom: 20px;
}

.sort-options label {
  margin-right: 10px;
  color: white;
  font-weight: bold;
  font-size: 1.6em;
}

.sort-options select {
  margin-right: 20px;
  padding: 5px;
  border: 1px solid #ccc;
  border-radius: 5px;
  font-size: 1em;
}

.product-list {
  display: flex;
  flex-wrap: wrap;
  justify-content: center;
}

.product-card {
  background-color: white;
  border: 1px solid #ccc;
}

```

```

border-radius: 10px;
margin: 15px;
padding: 15px;
text-align: center;
width: 220px;
box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
transition: transform 0.3s, box-shadow 0.3s;
}

.product-card:hover {
  transform: translateY(-5px);
  box-shadow: 0 6px 15px rgba(0, 0, 0, 0.2);
}

.product-image {
  width: 100%;
  height: auto;
  max-height: 150px;
  object-fit: contain;
  border-radius: 10px;
}

.product-name {
  font-size: 1.2em;
  margin: 10px 0;
  color: #333;
  text-decoration: none;
}

.product-price {
  font-size: 1.1em;
  color: #28a745;
  margin: 5px 0;
  text-decoration: none;
}

.add-to-cart-btn {
  background-color: #007bff;
  color: white;
  border: none;
  border-radius: 5px;
  padding: 10px;
  cursor: pointer;
  transition: background-color 0.3s, transform 0.2s;
  width: 100%;
  font-size: 1em;
}

.add-to-cart-btn:hover {
  background-color: #0056b3;
  transform: scale(1.05);
}

```

Изм.	Лист	№ докум.	Подпись	Дата

```

}

.reset-filters-btn {
  background-color: #dc3545;
  color: white;
  border: none;
  border-radius: 5px;
  padding: 10px 15px;
  cursor: pointer;
  transition: background-color 0.3s, transform 0.2s;
}

.reset-filters-btn:hover {
  background-color: #23b5c8;
  transform: scale(1.05);
}

```

Файл Checkout.js

```

import React, { useState } from 'react';
import { useNavigate, useLocation } from 'react-router-dom';
import './Checkout.css';

const Checkout = ({ onClearCart }) => {
  const location = useLocation();
  const cartItems = location.state?.orderDetails.items || [];

  const [city, setCity] = useState('');
  const [postalCode, setPostalCode] = useState('');
  const [address, setAddress] = useState('');
  const [email, setEmail] = useState('');
  const [phoneNumber, setPhoneNumber] = useState('');
  const [paymentMethod, setPaymentMethod] = useState('cash');
  const [cardNumber, setCardNumber] = useState('');

  const navigate = useNavigate();

  const handleOrder = () => {
    const confirmOrder = window.confirm('Вы уверены, что хотите оформить заказ?');
    if (confirmOrder) {
      const newOrderDetails = {
        items: cartItems,
        city,
        postalCode,
        address,
        email,
        phoneNumber,
        paymentMethod,
      };
    }
  };
};

```

```

        cardNumber: paymentMethod === 'card' ?
cardNumber : null,
        };

        alert('Спасибо, ваш заказ оформлен!');
        onClearCart();

        localStorage.setItem('orderDetails',
JSON.stringify(newOrderDetails));

        navigate('/order-history', { state: { orderDetails:
newOrderDetails } });
    }
};

return (
    <div className="checkout">
        <h2>Заполните следующие поля для оформления
заказа</h2>

        <div className="form-group">
            <label>Город:</label>
            <input
                type="text"
                value={city}
                onChange={(e) => setCity(e.target.value)}
                required
            />
        </div>

        <div className="form-group">
            <label>Почтовый индекс:</label>
            <input
                type="text"
                value={postalCode}
                onChange={(e) =>
setPostalCode(e.target.value)}
                required
            />
        </div>

        <div className="form-group">
            <label>Адрес:</label>
            <input
                type="text"
                value={address}
                onChange={(e) => setAddress(e.target.value)}
                required
            />
        </div>
    </div>
);

```

```

<div className="form-group">
  <label>Почтовый ящик:</label>
  <input
    type="email"
    value={email}
    onChange={(e) => setEmail(e.target.value)}
    required
  />
</div>

<div className="form-group">
  <label>Номер телефона:</label>
  <input
    type="tel"
    value={phoneNumber}
    onChange={(e) =>
setPhoneNumber(e.target.value)}
    required
  />
</div>

<div className="form-group">
  <label>Способ оплаты:</label>
  <div>
    <input
      type="radio"
      value="cash"
      checked={paymentMethod === 'cash'}
      onChange={(e) =>
setPaymentMethod(e.target.value)}
    />
    <label>Наличными</label>
  </div>
  <div>
    <input
      type="radio"
      value="card"
      checked={paymentMethod === 'card'}
      onChange={(e) =>
setPaymentMethod(e.target.value)}
    />
    <label>Картой</label>
  </div>
</div>

{paymentMethod === 'card' && (
  <div className="form-group">
    <label>Номер карты:</label>
    <input
      type="password"
      value={cardNumber}

```

Изм.	Лист	№ докум.	Подпись	Дата

```

                                onChange={ (e) =>
setCardNumber (e.target.value) }
                                required
                                />
                                </div>
                                ) }

                                <button onClick={handleOrder} className="order-
button">
                                Оформить заказ
                                </button>
                                </div>
                                );
};

export default Checkout;

```

Файл Checkout.css

```

.checkout {
  background: #fcfafa;
  border-radius: 10px;
  padding: 30px;
  box-shadow: 0 4px 20px rgba(0, 0, 0, 0.1);
  max-width: 600px;
  margin: 20px auto;
  font-family: 'Arial', sans-serif;
}

.checkout h2 {
  color: #333;
  font-size: 28px;
  margin-bottom: 20px;
  text-align: center;
}

input[type="text"]:focus,
input[type="tel"]:focus,
input[type="email"]:focus {
  border-color: #28a745;
  outline: none;
}

.order-button {
  width: 100%;
  padding: 12px;
  font-size: 18px;
  background: #28a745;
  color: white;
  border: none;
}

```

```

        border-radius: 5px;
        cursor: pointer;
        transition: background 0.3s, transform 0.2s, box-shadow
0.2s;
    }

    .order-button:hover {
        background: #218838;
        transform: scale(1.05);
        box-shadow: 0 4px 15px rgba(0, 0, 0, 0.2);
    }

    .error-message {
        color: red;
        font-size: 14px;
    }

    .success-message {
        color: green;
        font-size: 14px;
    }

```

Файл Header.js

```

import React, { useState } from 'react';
import { Link, useNavigate } from 'react-router-dom';
import Modal from 'react-modal';
import './Header.css';
import { FaHome, FaHistory } from 'react-icons/fa';

Modal.setAppElement('#root');

const Header = ({ cartItems, products }) => {
    const [searchTerm, setSearchTerm] = useState('');
    const [filteredProducts, setFilteredProducts] =
useState([]);
    const [isModalOpen, setIsModalOpen] = useState(false);
    const [username, setUsername] = useState('');
    const [password, setPassword] = useState('');
    const [isDropdownOpen, setIsDropdownOpen] = useState(false);
    const navigate = useNavigate();

    const handleSearchChange = (event) => {
        const value = event.target.value;
        setSearchTerm(value);

        const filtered = products.filter(product => {
            const name = product.name ?
product.name.toLowerCase() : '';
            const category = product.category ?
product.category.toLowerCase() : '';

```

```

        const brand = product.brand ?
product.brand.toLowerCase() : '';
        return name.includes(value.toLowerCase()) ||
        category.includes(value.toLowerCase()) ||
        brand.includes(value.toLowerCase());
    });
    setFilteredProducts(filtered);
    setIsDropdownOpen(value.length > 0 && filtered.length >
0);
    };

    const handleSearchSubmit = (event) => {
        event.preventDefault();
        if (filteredProducts.length > 0) {
            window.location.href = '/?search=' +
encodeURIComponent(searchTerm);
        }
    };

    const handleLogin = () => {
        console.log("Логин:", username);
        console.log("Пароль:", password);
        alert('Авторизация успешна!');
        closeModal();
        navigate('/');
    };

    const openModal = () => {
        setIsModalOpen(true);
    };

    const closeModal = () => {
        setIsModalOpen(false);
        setUsername('');
        setPassword('');
    };

    const handleHomeClick = () => {
        navigate('/');
    };

    const handleOrderHistoryClick = () => {
        navigate('/order-history');
    };

    const handleDropdownClick = () => {
        setIsDropdownOpen(!isDropdownOpen);
    };

    return (
        <header className="header">

```

```

1Love</div>
    <div className="logo" onClick={handleHomeClick}>FVT
1Love</div>
    <div className="search-container">
        <form onSubmit={handleSearchSubmit}
className="search-form">
            <input
                type="text"
                placeholder="Поиск товаров, категорий
или брендов..."
                value={searchTerm}
                onChange={handleSearchChange}
                className="search-input"
                onClick={handleDropdownClick}
            />
            {isDropdownOpen && (
                <div className="dropdown">
                    {filteredProducts.map((product,
index) => (
                        <Link
                            key={index}
                            to={`\product/\${product.id}`}
                            className="dropdown-item"
                            onClick={() => {
                                setSearchTerm('');
                                setIsDropdownOpen(false);
                            }}
                        >
                            {product.name}
                            ({product.category}, {product.brand})
                        </Link>
                    )}}
                </div>
            )}
        </form>
    </div>
    <div className="navbar">
        <button onClick={handleHomeClick}
className="home-button">
            <FaHome size={24} />
        </button>
        <Link to="/cart" className="cart-icon">
            
            {cartItems > 0 && (

```

```

        <div className="cart-count">
            {cartItems}
        </div>
    )}
</Link>
<button onClick={handleOrderHistoryClick}
className="history-button">
    <FaHistory size={24} />
</button>
<button onClick={openModal} className="auth-
button">🔑</button>
</div>

{/* Модальное окно авторизации */}
<Modal
    isOpen={isModalOpen}
    onRequestClose={closeModal}
    contentLabel="Авторизация"
    className="modal"
    overlayClassName="modal-overlay"
>
    <div className="modal-header">

        <h2>Авторизация</h2>
        <span className="close"
onClick={closeModal}>&times;</span>
    </div>
    <div className="modal-body">
        <input
            type="text"
            placeholder="Логин"
            value={username}
            onChange={(e) =>
setUsername(e.target.value)}
            required
            className="modal-input"
        />
        <input
            type="password"
            placeholder="Пароль"
            value={password}
            onChange={(e) =>
setPassword(e.target.value)}
            required
            className="modal-input"
        />
    </div>
    <div className="modal-footer">
        <button onClick={handleLogin}
className="modal-button">Войти</button>

```

```

        <button onClick={closeModal}
className="modal-button">Заккрыть</button>
      </div>
    </Modal>
  </header>
);
};

export default Header;

```

Файл Header.css

```

.header {
  position: fixed;
  top: 0;
  left: 0;
  width: 100%;
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100px;
  padding: 0 20px;
  background-color: rgba(42, 42, 42, 0.9);
  color: white;
  box-shadow: 0 4px 20px rgba(0, 0, 0, 0.3);
  border-radius: 10px;
  z-index: 1000;
  backdrop-filter: blur(10px);
}

.logo {
  font-size: 48px;
  font-weight: bold;
  color: #FFA500;
  text-align: left;
  margin-left: 20px;
  cursor: pointer;
  transition: transform 0.3s;
}

.logo:hover {
  transform: scale(1.1);
  color: #FFD700;
}

.navbar {
  display: flex;
  gap: 40px;
  margin-right: 60px;
  align-items: center;
}

```

Изм.	Лист	№ докум.	Подпись	Дата

```

}

.cart-icon {
  position: relative;
  display: flex;
  align-items: center;
  color: #FFFFFFF;
  text-decoration: none;
}

.cart-icon img {
  width: 70px;
  height: auto;
  cursor: pointer;
}

.cart-count {
  position: absolute;
  top: -10px;
  right: -10px;
  background-color: #FFA500;
  color: white;
  border-radius: 50%;
  padding: 2px 6px;
  font-size: 14px;
}

.search-icon {
  position: relative;
  cursor: pointer;
}

.search-button {
  background-color: #FFA500;
  border: none;
  padding: 10px;
  border-radius: 5px;
  cursor: pointer;
  transition: background-color 0.3s;
}

.search-button:hover {
  background-color: #FFD700;
}

body {
  margin: 0;
  padding: 0;
  background-color: #1E1E1E;
  color: #B0B0B0;
  padding-top: 120px;

```

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

Лист

83

```

        background-image: linear-gradient(to right, #000000,
#000000);
    }

    @keyframes pulse {

        0%,
        100% {
            transform: scale(1);
        }

        50% {
            transform: scale(1.05);
        }
    }

    .dropdown {
        position: absolute;
        top: 100%;
        left: 0;
        background-color: white;
        border: 1px solid #ccc;
        border-radius: 5px;
        z-index: 1000;
        width: 100%;
    }

    .dropdown-item {
        padding: 5px;

        text-decoration: none;
        color: black;
        display: block;
    }

    .dropdown-item:hover {
        background-color: #f0f0f0;
    }

    .modal {
        display: block;
        position: fixed;
        z-index: 1001;
        left: 0;
        top: 0;
        width: 100%;
        height: 100%;
        overflow: auto;
        background-color: rgba(0, 0, 0, 0.7);
    }

```

```

.modal-content {
    background-color: #fefefe;
    margin: 15% auto;
    padding: 20px;
    border: 1px solid transparent;
    border-radius: 10px;
    display: flex;
    flex-direction: column;
    align-items: center;
    position: relative;
}

.modal-content::before {
    content: "";
    position: absolute;
    top: -5px;
    left: -5px;
    right: -5px;
    bottom: -5px;
    background: linear-gradient(45deg, #ccc, #888);
    border-radius: 10px;
    z-index: -1;
}

.modal-header {
    display: flex;
    justify-content: center;
    align-items: center;
    width: 100%;
    color: white;
    font-size: 24px;
}

.modal-submit {
    background-color: #FFA500;
    border: none;
    color: white;
    padding: 10px 20px;
    border-radius: 5px;
    cursor: pointer;
    transition: background-color 0.3s;
    flex: 1;
    margin: 0 5px;
}

.modal-submit:hover {
    background-color: #FFD700;
}

.modal-close {
    background: none;

```

```

border: none;
color: #FF0000;
font-size: 20px;
cursor: pointer;
margin-top: 10px;
}

.modal-close:hover {
color: #FF4500;
}

.modal-footer a {
color: #FFA500;
text-decoration: none;
}

.modal-footer a:hover {
text-decoration: underline;
}

.cart-message {
color: white;
font-size: 18px;
margin-left: 20px;

display: flex;
align-items: center;
}

.logo {
font-size: 48px;
font-weight: bold;
color: #5bfd33;
text-align: left;
margin-right: auto;

cursor: pointer;
transition: transform 0.3s;
}

.navbar {
display: flex;
gap: 40px;
margin-left: auto;

align-items: center;
}

.modal-body {
display: flex;
flex-direction: column;

```

```

        align-items: center;
        margin: 10px 0;
    }

    .modal-input {
        width: 100%;
        max-width: 300px;
        padding: 10px;
        margin: 10px 0;
        border: 1px solid #ccc;
        border-radius: 5px;
        font-size: 16px;
    }

    .modal-footer {
        display: flex;
        justify-content: center;
        margin-top: 20px;
    }

    .modal-button {
        max-width: 120px;
        min-width: 80px;
        margin: 0 10px;
        background-color: #FFA500;
        border: none;
        color: white;
        padding: 10px;
        border-radius: 5px;
        cursor: pointer;
        transition: background-color 0.3s;
    }

    .modal-button:hover {
        background-color: #FFD700;
    }

```

Файл HomePage.js

```

import React from 'react';
import ImageSlider from './ImageSlider';
import ProductList from './ProductList';
import { Link } from 'react-router-dom';
import './HomePage.css';

const HomePage = ({ products, onAddToCart }) => {
    return (
        <div className="home-page">
            <ImageSlider />
            <h2 className="section-title">Каталог товаров</h2>
            <div className="category-icons">

```

```

        <Link to="/category/чайники"
className="category-icon">
            <div className="icon-container">
                
                <span>Чайники</span>
            </div>
        </Link>
        <Link to="/category/фены" className="category-
icon">
            <div className="icon-container">
                
                <span>Фены</span>
            </div>
        </Link>
        <Link to="/category/телефоны"
className="category-icon">
            <div className="icon-container">
                
                <span>Телефоны</span>
            </div>
        </Link>
        <Link to="/category/наушники"
className="category-icon">
            <div className="icon-container">
                
                <span>Наушники</span>
            </div>
        </Link>
        <Link to="/category/телевизоры"
className="category-icon">
            <div className="icon-container">
                
                <span>Телевизоры</span>
            </div>
        </Link>
        <Link to="/category/приставки"
className="category-icon">
            <div className="icon-container">
                
        <span>Приставки</span>
    </div>
</Link>
<Link to="/category/колонки"
className="category-icon">
    <div className="icon-container">
        
        <span>Колонки</span>
    </div>
</Link>
<Link to="/category/часы" className="category-
icon">
    <div className="icon-container">
        
        <span>Часы</span>
    </div>
</Link>
</div>
<h2 className="section-title">Хиты продаж</h2>
<ProductList products={products}
onAddToCart={onAddToCart} />
</div>
);
};

export default HomePage;

```

Файл HomePage.css

```

.home-page {
    text-align: center;
}

.section-title {
    color: #FFFFFFF;
    font-size: 2.5em;
    margin: 20px 0;
    animation: slideIn 1s ease-in-out;
    text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.5);
}

.category-icons {
    display: flex;
    justify-content: center;
    margin: 20px 0;
}

```

```

}

.category-icon {
  margin: 0 20px;
  text-decoration: none;
}

.icon-container span {
  display: block;
  margin-top: 10px;
  color: #333;
  font-weight: bold;
  font-size: 1.5em;
}

@keyframes slideIn {
  0% {
    transform: translateY(-20px);
    opacity: 0;
  }

  100% {
    transform: translateY(0);
    opacity: 1;
  }
}

```

Файл Product.js

```

import React from 'react';
import { useNavigate } from 'react-router-dom';
import './ProductList.css';

const Product = ({ product, onAddToCart }) => {
  const navigate = useNavigate();

  const handleProductClick = () => {
    navigate(`/product/${product.id}`);
  };

  return (
    <div className="product-card"
      onClick={handleProductClick}>
      <img src={product.imageUrl} alt={product.name}
        className="product-image" />
      <h3 className="product-name">{product.name}</h3>
      <p className="product-
description">{product.description}</p>
      <p className="product-price">{product.price}₽</p>
      <button
        className="add-to-cart-button"

```

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

Лист

90

```

        onClick={ (e) => {
            e.stopPropagation();
            onAddToCart(product);
        }}
    >
        Добавить в корзину
    </button>
</div>
);
};

export default Product;

```

Файл Product.css

```

.product {
    background-color: #ffffff;
    border: 1px solid #ddd;
    border-radius: 8px;
    padding: 15px;
    text-align: center;
    transition: box-shadow 0.3s;
}

.product:hover {
    box-shadow: 0 4px 15px rgba(0, 0, 0, 0.2);
}

.product-image {
    max-width: 100%;
    height: auto;
    border-radius: 5px;
}

.product-name {
    font-size: 20px;
    margin: 10px 0;
    cursor: pointer;
}

.product-price {
    color: #28a745;
    font-weight: bold;
    font-size: 1.2em;
}

```

Файл ProductDetail.js

```

import React from 'react';
import { useParams, useNavigate } from 'react-router-dom';

```

					ПГУ 1.090301.8.001 ПЗ	Лист
						91
Изм.	Лист	№ докум.	Подпись	Дата		

```

import './ProductDetail.css';

const ProductDetails = ({ products, onAddToCart }) => {
  const { id } = useParams();
  const navigate = useNavigate();
  const product = products.find(item => item.id ===
parseInt(id));

  if (!product) {
    return <div>Товар не найден</div>;
  }

  const handleAddToCart = () => {
    onAddToCart(product);
  };

  const handleBack = () => {
    navigate(-1);
  };

  const characteristicsArray = product.characteristics
    .split('-')
    .map(item => item.trim())
    .filter(item => item);

  return (
    <div className="product-detail-container">
      <div className="product-detail-image-container">
        <img src={product.imageUrl} alt={product.name}
className="product-detail-image" />
      </div>
      <div className="product-detail-info">
        <h1 className="product-detail-
title">{product.name}</h1>
        <p className="product-detail-
description">{product.description}</p>

        {/* Новые элементы для отображения полной
информации о продукте */}
        <h2 className="full-description-title">Полное
описание</h2>
        <p className="full-
description">{product.fullDescription}</p>

        <h2 className="characteristics-
title">Характеристики</h2>
        <ul className="characteristics">
          {characteristicsArray.map((item, index) => (
            <li key={index}>{item}</li>
          ))}
        </ul>
      </div>
    </div>
  );
};

```

```

        <p className="product-detail-price">Цена:
{product.price} ₸</p>
        <div className="button-container">
            <button className="add-to-cart-button1"
onClick={handleAddToCart}>
                Добавить в корзину
            </button>
            <button className="back-button"
onClick={handleBack}>
                Назад
            </button>
        </div>
    </div>
</div>
);
};

export default ProductDetails;

```

Файл ProductDetail.css

```

.product-detail-container {
    display: flex;
    padding: 20px;
    max-width: 1200px;
    margin: auto;
    border: 1px solid #e0e0e0;
    border-radius: 12px;
    box-shadow: 0 4px 20px rgba(0, 0, 0, 0.1);
    background-color: #f9f9f9;
}

.product-detail-image-container {
    flex: 1;
    display: flex;
    justify-content: center;
    align-items: center;
}

.product-detail-image {
    width: 100%;
    max-width: 450px;
    height: auto;
    border-radius: 12px;
    box-shadow: 0 2px 10px rgba(0, 0, 0, 0.2);
}

.product-detail-info {
    flex: 2;
    padding-left: 20px;
}

```

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

Лист

93

```

        display: flex;
        flex-direction: column;
    }

    .product-detail-title {
        text-align: center;
        font-size: 32px;
        margin-bottom: 10px;
        color: #333;
    }

    .product-detail-description {
        margin-bottom: 10px;
        font-size: 18px;
        color: #666;
        text-align: center;
    }

    .full-description-title,
    .characteristics-title {
        font-size: 24px;
        margin-top: 20px;
        color: #333;
        text-align: center;
        font-weight: bold;
    }

    .full-description,
    .characteristics {
        font-size: 18px;
        color: #444;
        margin-bottom: 10px;
        padding-left: 5%;
        padding-right: 5%;
    }

    .product-detail-price {
        font-size: 28px;
        color: #FF5722;
        text-align: center;
    }

    .button-container {
        display: flex;
        justify-content: center;
        margin-top: auto;
    }

    .add-to-cart-button1,
    .back-button {

```

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

Лист

94

```

padding: 16px 24px;
border: none;
border-radius: 12px;
cursor: pointer;
width: 180px;
font-size: 20px;
transition: transform .2s ease-in-out, box-shadow .2s ease-
in-out;
margin: 0 10px;
}

.add-to-cart-button1 {
background-color: #4CAF50;
color: white;
}

.add-to-cart-button1:hover {
background-color: #45a049;
transform: scale(1.05);
box-shadow: 0 4px 15px rgba(0, 0, 0, .3);
}

.back-button {
background-color: #2196F3;
color: white;
}

.back-button:hover {
background-color: #1976D2;
}

```

Файл ProductList.js

```

import React from 'react';
import { useNavigate } from 'react-router-dom';
import './ProductList.css';

const ProductList = ({ products = [], onAddToCart }) => {
  const navigate = useNavigate();
  const handleProductClick = (productId) => {
    navigate(`/product/${productId}`);
  };

  return (
    <div>
      <h2 className="product-list-title"></h2>
      <div className="product-list">
        {products.length === 0 ? (
          <p>Нет доступных товаров</p>
        ) : (
          products.map(product => (
            <div

```

```

        key={product.id}
        className="product-list-card"
        onClick={() =>
handleProductClick(product.id)}
      >
        <div className="image-container">
          <img src={product.imageUrl}
alt={product.name} className="product-list-image" />
        </div>
        <h3
          className="product-list-name"
          onClick={(e) => {
            e.stopPropagation();

handleProductClick(product.id);
          }}
        >
          {product.name}
        </h3>
        <p className="product-list-
description">{product.description}</p>
        <p className="product-list-
price">Цена: {product.price}Р</p> { /* Добавлено слово "Цена:"
*/}
        <button
          className="add-to-cart-button"
          onClick={(e) => {
            e.stopPropagation();
            onAddToCart(product);
          }}
        >
          Добавить в корзину
        </button>
      </div>
    ))
  )}
</div>
</div>
);
};

export default ProductList;

```

Файл ProductList.css

```

.product-list-title {
  text-align: center;
  font-size: 2.5em;
  margin-bottom: 30px;
  color: #333;
  font-weight: bold;

```

```

}

.product-list {
  display: flex;
  flex-wrap: wrap;
  justify-content: center;
  gap: 20px;
  padding: 20px;
}

.product-list-card {
  border: 1px solid #ddd;
  border-radius: 12px;
  padding: 15px;
  text-align: center;
  background-color: #fff;
  transition: transform 0.3s ease, box-shadow 0.3s ease;
  display: flex;
  flex-direction: column;
  justify-content: space-between;
  width: 250px;
  height: 380px;
  box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);
}

.product-list-card:hover {
  transform: scale(1.05);
  box-shadow: 0 4px 20px rgba(0, 0, 0, 0.2);
}

.image-container {
  display: flex;
  justify-content: center;
  align-items: center;
  height: 200px;
  overflow: hidden;
}

.product-list-image {
  max-width: 100%;
  max-height: 100%;
  object-fit: cover;
  border-radius: 8px;
  transition: transform 0.3s ease;
}

.product-list-name {
  margin: 10px 0;
  font-size: 1.5em;
  color: #333;
}

```

```

.product-list-description {
  margin: 5px;
  color: #666;
  font-size: 0.9em;
  line-height: 1.5;
}

.product-list-price {
  font-size: 1.2em;
  font-weight: bold;
  color: #28a745;
  margin: 10px 0;
}

.product-list-price-label {
  font-weight: normal;
  color: #999;
}

.add-to-cart-button {
  padding: 12px 20px;
  background-color: #007bff;
  color: white;
  border: none;
  border-radius: 5px;
  cursor: pointer;
  transition: background-color 0.3s ease, transform 0.3s ease;
  margin: 10px auto;
  font-size: 1em;
  width: 80%;
}

.add-to-cart-button:hover {
  background-color: #0056b3;
  transform: translateY(-2px);
}

.add-to-cart-button:active {
  transform: translateY(0);
}

```

Файл App.js

```

import React, { useState, useEffect } from 'react';
import { BrowserRouter as Router, Route, Routes } from 'react-router-dom';
import './App.css';
import Header from './components/Header';
import ProductList from './components/ProductList';
import ProductDetail from './components/ProductDetail';

```

					ПГУ 1.090301.8.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		98

```

import Cart from './components/Cart';
import Checkout from './components/Checkout';
import HomePage from './components/HomePage';
import CategoryPage from './components/CategoryPage';
import OrderHistory from './OrderHistory';
import Loading from './components/Loading';
import axios from 'axios';

const App = () => {
  const [cart, setCart] = useState([]);
  const [products, setProducts] = useState([]);
  const [loading, setLoading] = useState(true);
  const [user, setUser] = useState(null); // состояние
пользователя

  useEffect(() => {
    const fetchProducts = async () => {
      try {
        const response = await
axios.get('http://localhost:5000/api/products');
        setProducts(response.data);
      } catch (error) {
        console.error('Ошибка при загрузке продуктов:', error);
      } finally {
        setLoading(false);
      }
    };

    fetchProducts();
  }, []);

  const onAddToCart = (product) => {
    setCart(prevCart => [...prevCart, product]);
  };

  const onRemoveFromCart = (productId) => {
    setCart(prevCart => {
      const updatedCart = [...prevCart];
      const productIndex = updatedCart.findIndex(item => item.id
=== productId);

      if (productIndex !== -1) {
        if (updatedCart[productIndex].quantity > 1) {
          updatedCart[productIndex].quantity -= 1;
        } else {
          updatedCart.splice(productIndex, 1);
        }
      }

      return updatedCart;
    });
  };

```

```

    };

    const clearCart = () => {
      setCart([]);
    };

    if (loading) {
      return <Loading />;
    }

    return (
      <Router>
        <Header cartItems={cart.length} products={products}
        user={user} setUser ={setUser } />
        <Routes>
          <Route path="/" element={<HomePage products={products}
            onAddToCart={onAddToCart} />} />
          <Route path="/cart" element={<Cart cartItems={cart}
            onRemoveFromCart={onRemoveFromCart} onClearCart={clearCart} />}
            />
          <Route path="/checkout" element={<Checkout
            onClearCart={clearCart} />} />
          <Route path="/product/:id" element={<ProductDetail
            products={products} onAddToCart={onAddToCart} />} />
          <Route path="/category/:tag" element={<CategoryPage
            products={products} onAddToCart={onAddToCart} />} />
          <Route path="/order-history" element={<OrderHistory />}
            />
        </Routes>
      </Router>
    );
  };
};

export default App;

```

Файл App.css

```

body {
  margin: 0;
  font-family: Arial, sans-serif;
  background: linear-gradient(to bottom, #f0f0f0, #ffffff);
  color: #333;
}

.container {
  max-width: 1200px;
  margin: 0 auto;
  padding: 20px;
}

.header {
  background: #333;

```

```

    color: white;
    padding: 15px;
    text-align: center;
}

.cart {
    background: #fff;
    border-radius: 8px;
    padding: 20px;
    box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);
}

.cart-item {
    display: flex;
    justify-content: space-between;
    align-items: center;
    margin: 10px 0;
    padding: 10px;
    border-bottom: 1px solid #ddd;
}

.cart-item:last-child {
    border-bottom: none;
}

.checkout-button {
    background: #28a745;
    color: white;
    padding: 10px 15px;
    border: none;
    border-radius: 5px;
    cursor: pointer;
    transition: background 0.3s;
}

.checkout-button:hover {
    background: #218838;
}

.gradient-border {
    border-left: 10px solid transparent;
    border-right: 10px solid transparent;
    background: linear-gradient(to right, #000, #333);
    padding: 10px;
    border-radius: 8px;
}

```

Файл server.js

```

const express = require('express');
const mongoose = require('mongoose');

```

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

Лист

101

```

const cors = require('cors');
const bcrypt = require('bcrypt');
const jwt = require('jsonwebtoken');

const app = express();
const PORT = 5000;
const JWT_SECRET = 'jwt_secret_key';

mongoose.connect('mongodb://localhost:27017/db', {});

const UserSchema = new mongoose.Schema({
  username: { type: String, unique: true },
  passwordHash: String,
});

const User = mongoose.model('User', UserSchema);

const ProductSchema = new mongoose.Schema({
  id: Number,
  name: String,
  price: Number,
  description: String,
  imageUrl: String,
  category: String,
  fullDescription: String,
  tags: [String],
  deliveryTime: Number,
  brand: String,
  characteristics: String,
});

const Product = mongoose.model('Product', ProductSchema);

app.use(cors());
app.use(express.json());

app.post('/api/register', async (req, res) => {
  const { username, password } = req.body;
  const passwordHash = await bcrypt.hash(password, 10);
  try {
    const user = new User({ username, passwordHash });
    await user.save();
    res.status(201).json({ message: 'User created' });
  } catch (e) {
    res.status(400).json({ error: 'Username already exists' });
  }
});

app.post('/api/login', async (req, res) => {
  const { username, password } = req.body;
  const user = await User.findOne({ username });

```

```

    if (!user) return res.status(400).json({ error: 'Invalid
username or password' });

    const isPasswordValid = await bcrypt.compare(password,
user.passwordHash);
    if (!isPasswordValid) return res.status(400).json({ error:
'Invalid username or password' });

    const token = jwt.sign({ id: user._id, username: user.username
}, JWT_SECRET, { expiresIn: '1h' });
    res.json({ token });
  });

app.get('/api/products', async (req, res) => {
  const products = await Product.find();
  res.json(products);
});

app.listen(PORT, () => {
  console.log(`Server started on http://localhost:${PORT}`);
});

```

Файл products.json

```

{
  "id": 265,
  "name": "iPhone 16",
  "description": "Современный флагман компании Apple",
  "fullDescription": "iPhone 16 – это флагманский смартфон
от Apple с улучшенной камерой, мощным процессором и стильным
дизайном. Он поддерживает 5G и обладает высокой
производительностью, что делает его идеальным для игр и
работы.",
  "price": 75990,
  "imageUrl": "https://i-
ray.ru/upload/iblock/1c7/r2b1jy4cnh0zmjqzvsz63hp2lkd63h27.png",
  "tags": ["телефоны"],
  "deliveryTime": 4,
  "brand": "Apple",
  "characteristics": "- Дисплей: 6.1 дюйма Super Retina
XDR\n- Процессор: A16 Bionic\n- Опции хранения: 128GB, 256GB,
512GB\n- Камера: Основная 12MP + 12MP, фронтальная 12MP\n-
Батарея: Срок службы до 20 часов в режиме разговора\n-
Операционная система: iOS 16"
},
{
  "id": 280,
  "name": "Honor Magic 4 Pro",
  "description": "Флагманский смартфон с мощным
процессором и отличной камерой.",
  "fullDescription": "Honor Magic 4 Pro предлагает
потрясающее качество изображения благодаря своей камере и

```

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

Лист

103

```

мощному процессору. Он также поддерживает быструю зарядку и
беспроводную зарядку.",
    "price": 69990,
    "imageUrl": "https://img.mvideo.ru/Pdb/400104496b.jpg",
    "tags": ["телефоны"],
    "deliveryTime": 6,
    "brand": "Honor",
    "characteristics": "- Дисплей: 6.81 дюйма OLED\n-
Процессор: Qualcomm Snapdragon 8 Gen 1\n- Опции хранения: 256GB,
512GB\n- Камера: Основная 50MP + 50MP + 64MP, фронтальная
12MP\n- Батарея: 4600 mAh\n- Операционная система: Android 12"
  },
  {
    "id": 2355,
    "name": "Play Station 5",
    "description": "Лучшая игровая консоль.",
    "fullDescription": "PlayStation 5 предлагает невероятную
графику и быструю загрузку благодаря SSD. Идеальная для игр
нового поколения с поддержкой 4K и 120fps.",
    "price": 65990,
    "imageUrl": "https://static.insales-
cdn.com/r/yM_NpclEgCc/rs:fit:700:700:1/plain/images/products/1/6
509/468277613/sony_playatation_5.png@png",
    "tags": ["приставки"],
    "deliveryTime": 4,
    "brand": "Sony",
    "characteristics": "- Процессор: AMD Zen 2\n- Графика:
AMD RDNA 2\n- Хранение: 825GB SSD\n- Память: 16GB GDDR6\n-
Разрешение: 4K 120fps\n- Обратная совместимость: Поддержка PS4
игр"
  },
  ...
}

```

Файл users.json

```

{
  Vladislav:
    '$2b$10$EixZaYVK1fsbw1ZfbX3OXePaWxn96p36E/9V7Zp6q6u9aFqWcQ1eW'
  , // password123

  Alex:
    '$2b$10$7Qjv6R9XyR6v2lXQx9v0sOa6dVxqz5u9vYtWwT8J1B2C3D4E5F6G'
  , // mypassword

  Gora:
    '$2b$10$1a2b3c4d5e6f7g8h9i0jklmnopqrstuvwxyz1234567890abcdef
g', // securepass
};

```

Приложение В

(Обязательное)

Интернет-магазин электроники

Презентация

					<i>ПГУ 1.090301.8.001 ПЗ</i>	Лист
						105
Изм.	Лист	№ докум.	Подпись	Дата		

Выпускная бакалаврская работ «Интернет-магазин электроники»

Автор работы: студ
группы 21ВВВ1 Каг
Научный руководи
к.т.н., доцент кафе
Карамышева Н.С.

Изм.	Лист	№ докум.	Подпись	Дата

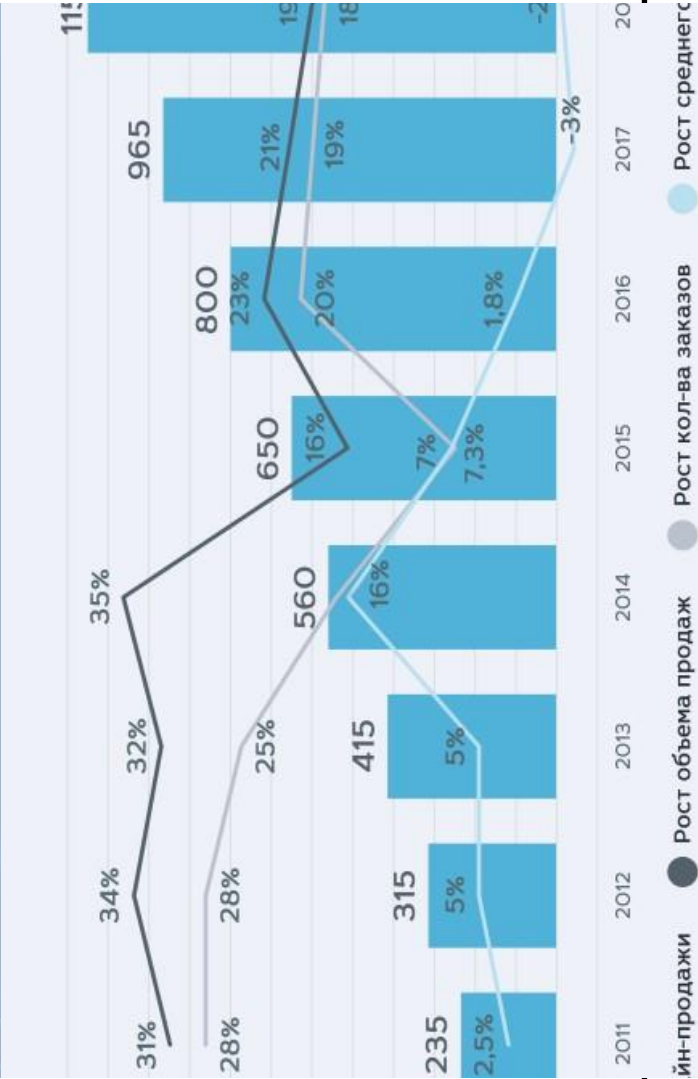
ПГУ 1.090301.8.001 ПЗ

Лист
106

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

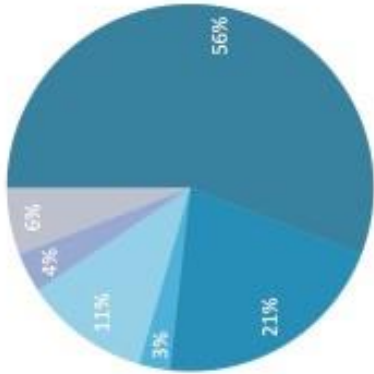
Актуальность темы



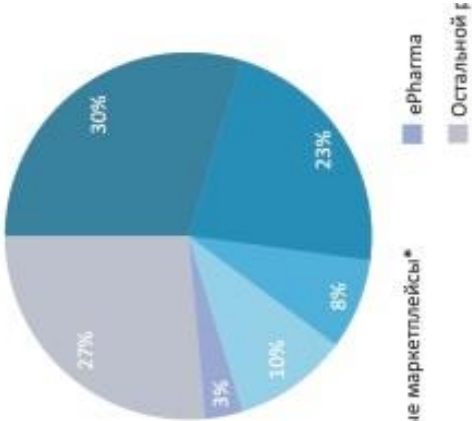
Актуальность темы

Оzon лидируют в российском eCommerce: на них приходится 77% всех заказов и 53% объема продаж.

Количество заказов, млн шт.



Объем продаж, млрд руб.



* Wildberries, Ozon, MegaMarket, Marnit Market, AliExpress (только российские продавцы)
** Ozon Fresh
Источник: Statista, данные за 2023-2024 гг.
Источник: Statista, данные за 2023-2024 гг.

проекта и постановка за

-приложение для онлайн -заказа товаров, которое обеспечит пользователю безопасную работу с интерфейсом, а также упростит процесс оформления заказа. Приложение будет кроссплатформенным, поддерживаться на современных устройствах и соответствовать следующим требованиям:

действия:

1. Приложение должно быть интуитивно понятным, чтобы даже неопытные пользователи могли легко ориентироваться в приложении.

2. Приложение должно обеспечивать удобную навигацию по каталогу товаров, возможность поиска товаров по категориям, а также функционал корзины для управления выбранными товарами.

3. Приложение должно обеспечивать защиту персональных данных пользователей.

требования к функционалу:

1. Пользователь должен иметь возможность регистрироваться и авторизовываться в приложении. Регистрация должна занимать минимальное количество шагов и быть простой.

2. Пользователь должен иметь возможность выбора одного из нескольких вариантов товара.

3. Пользователь должен иметь возможность добавления товара в корзину.

проекта и постановка за

еские требования:

ая система: Windows (7,8, 10 и 11)),

ли Linux (Ubuntu, Debian и Redhat);

,6 ГГц или выше;

ОЗУ;

одного дискового пространства;

T Framework 4.5.2.



Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

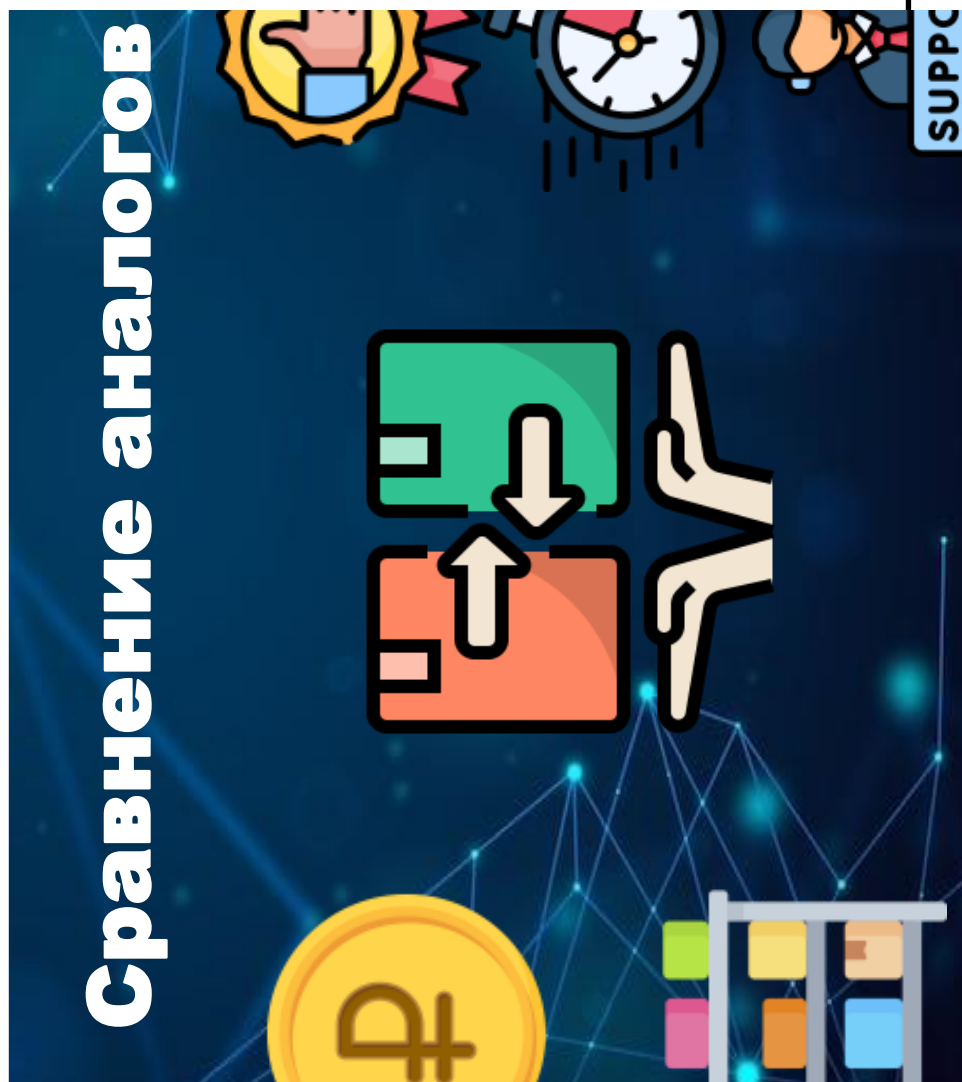


Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

Лист
111

Сравнение аналогов



Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

Лист
112

Диаграмма вариантов использования

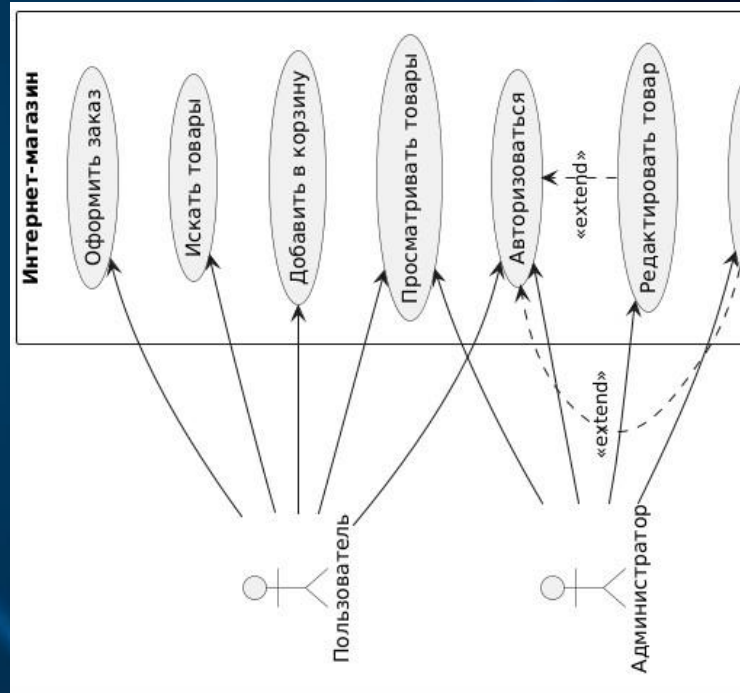
Интернет-магазин



Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

Диаграмма вариантов использования



Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

Лист
114

Стек технологий

- JavaScript



- ReactJS

- HTML



- MongoDB

- CSS



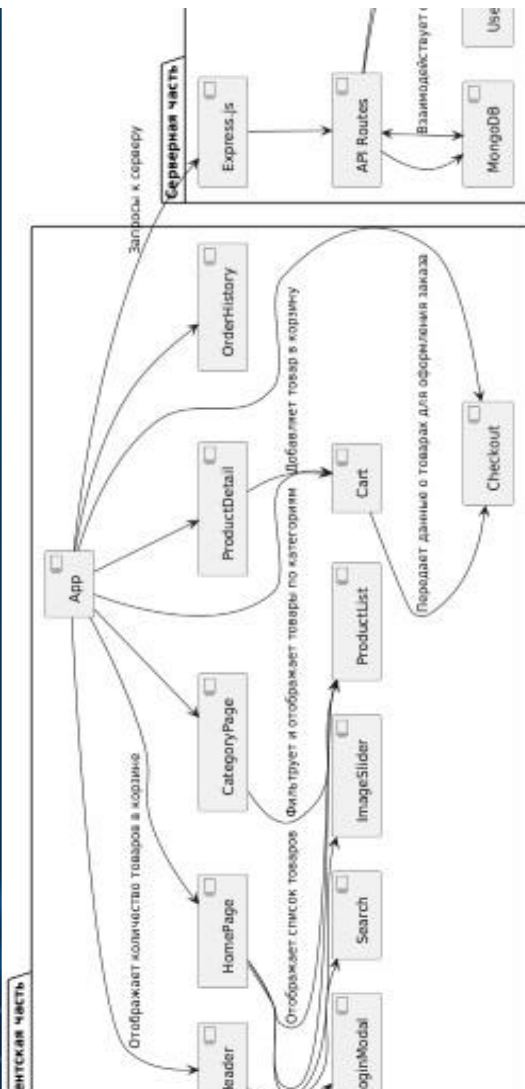
- Visual S

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

Лист
115

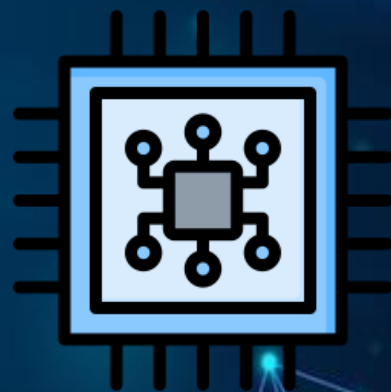
Диаграмма компонентов



Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

Компоненты программы



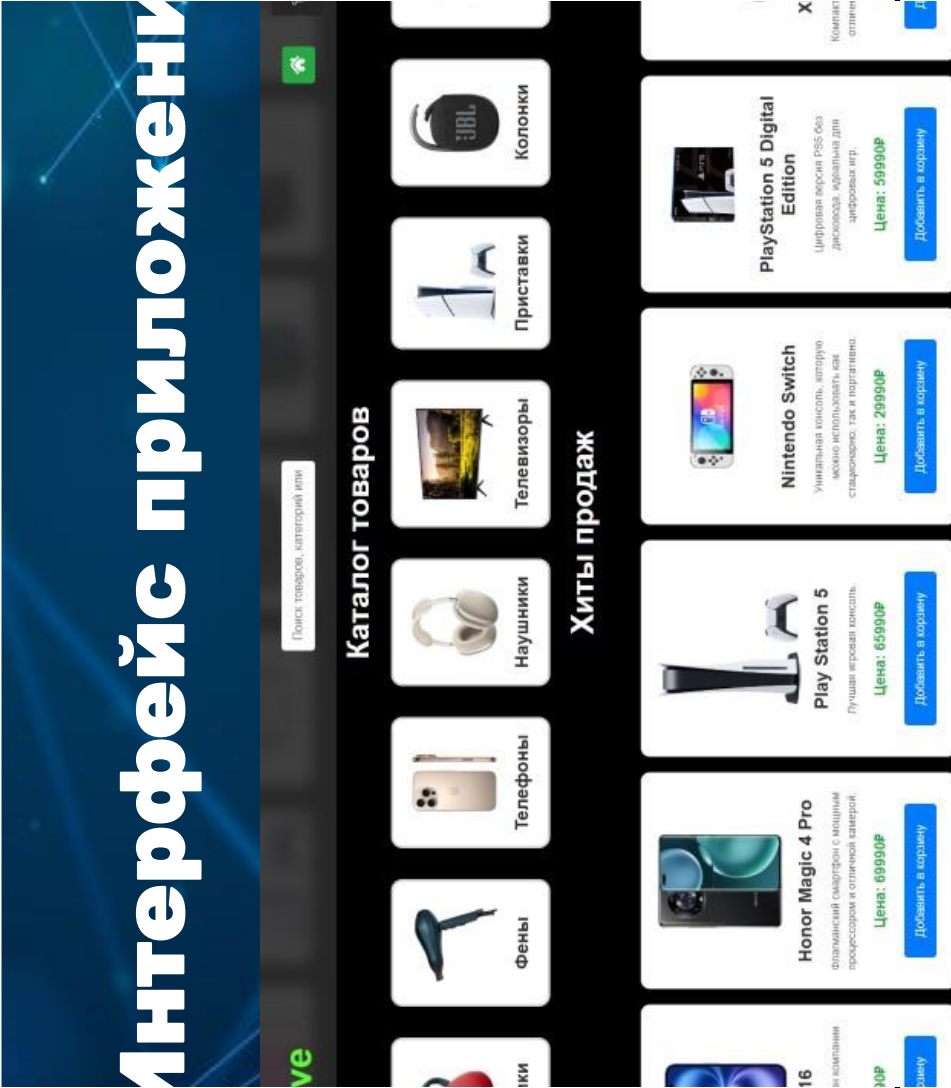
Изм.	Лист	№ докум.	Подпись	Дата

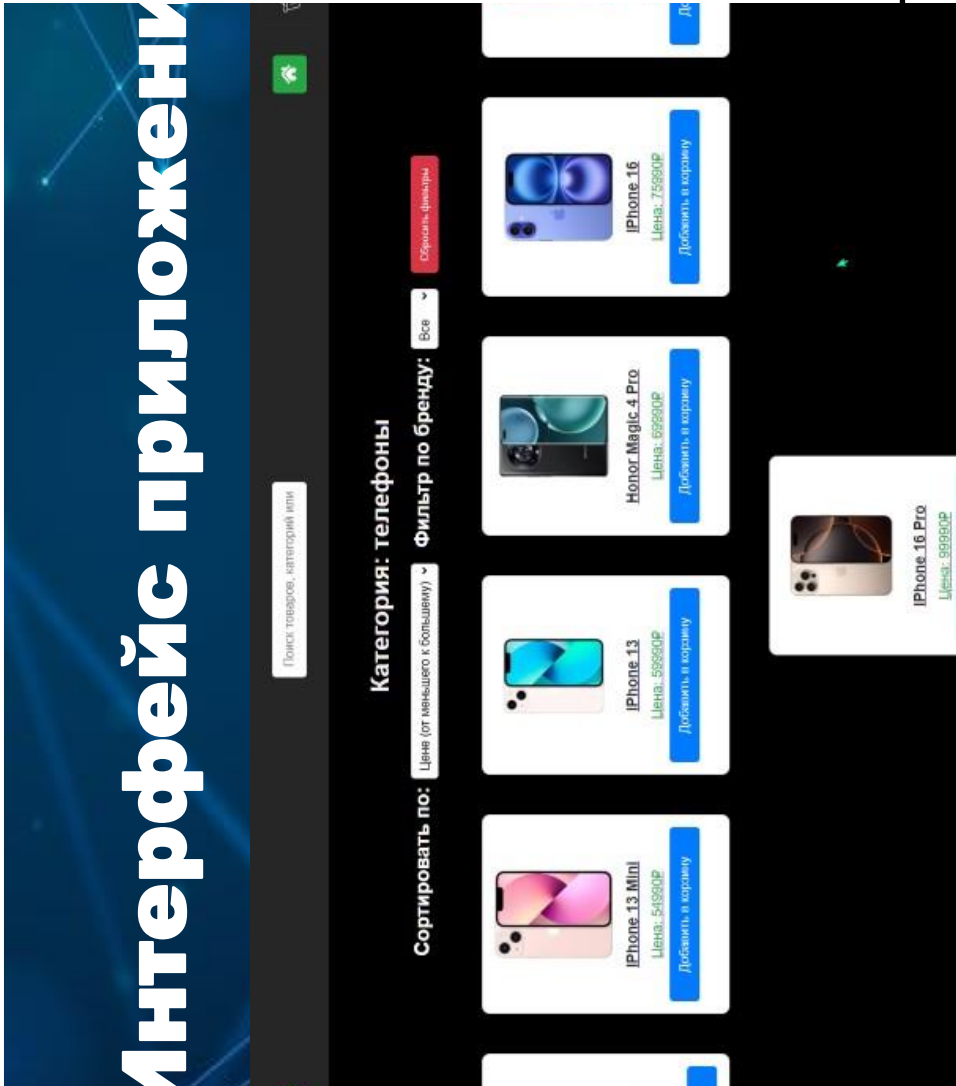
ПГУ 1.090301.8.001 ПЗ

Лист
117

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ







Tefal KO150F30

Электрический чайник с индикацией уровня воды и защитой от перегрева.

Полное описание

Tefal KO150F30 – это стильный и функциональный электронный чайник, который обеспечивает быструю и безопасную кипячение воды с индикацией уровня и защитой от перегрева.

Характеристики

- Объем: 1,7 литра
- Мощность: 2400 Вт
- Индикатор уровня воды: Да
- Материал: Пластик
- Автоотключение: Да
- Защита от перегрева: Да

Цена: 2990 ₮

Добавить в корзину

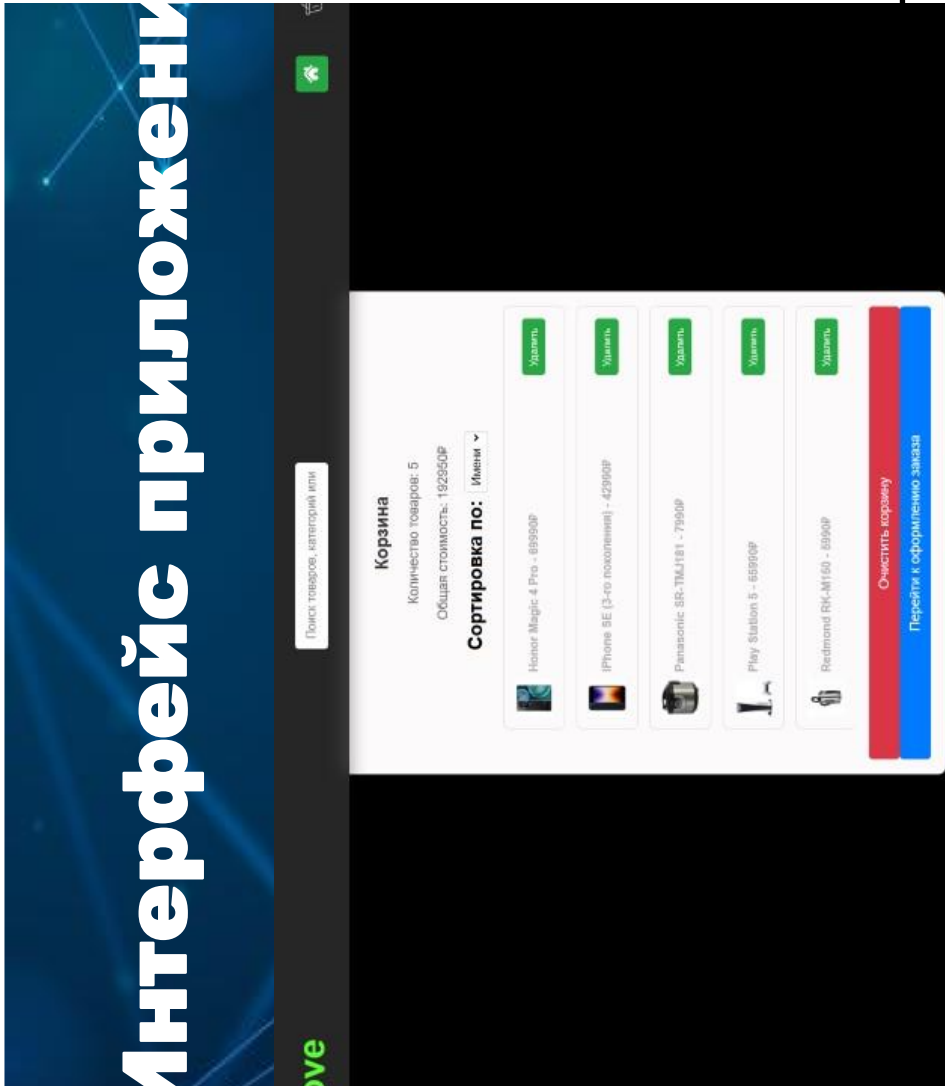
Назад

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

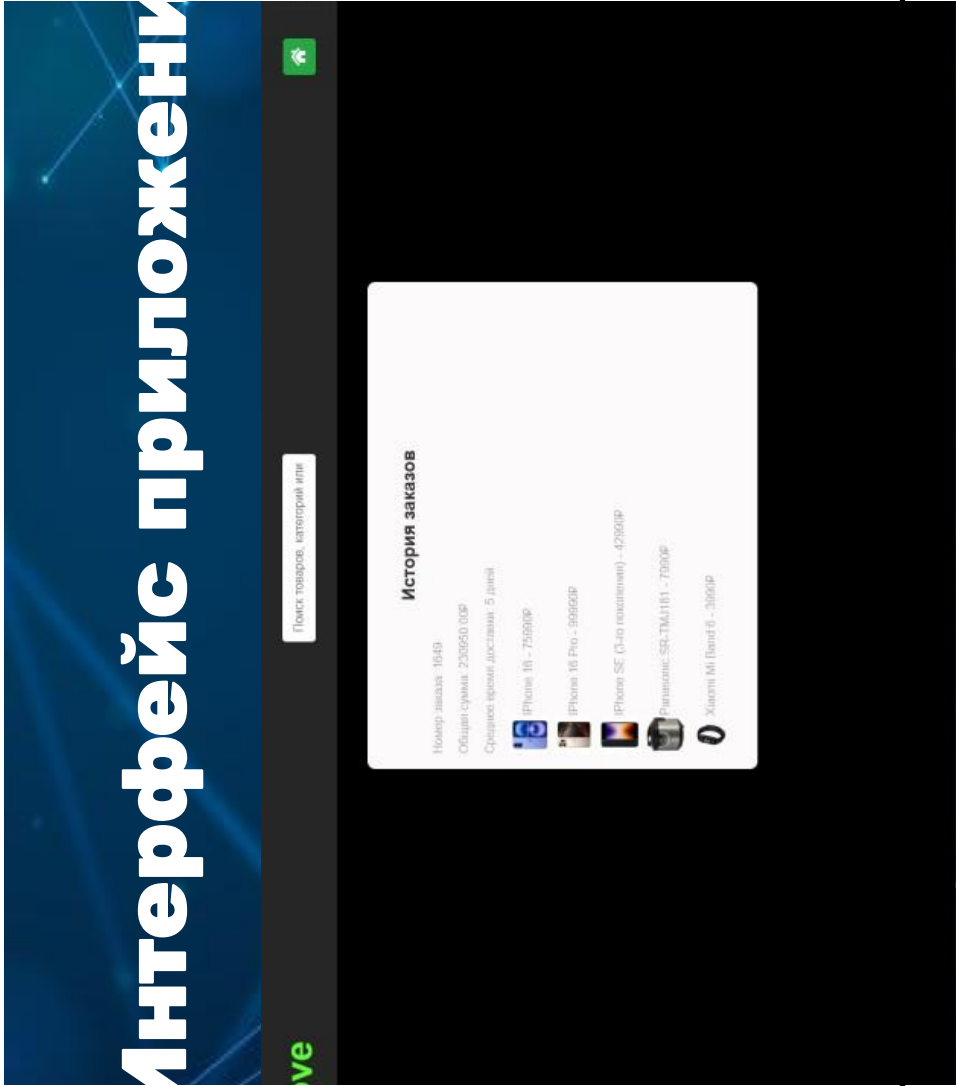
Лист

120

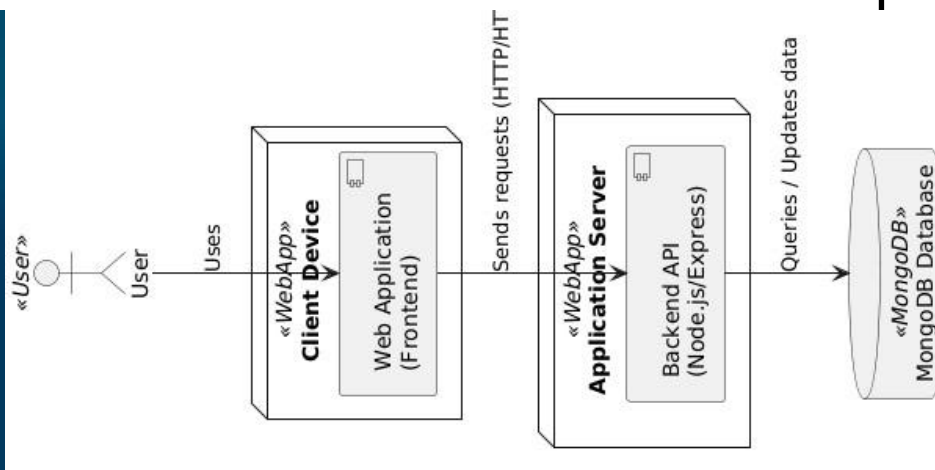


Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ



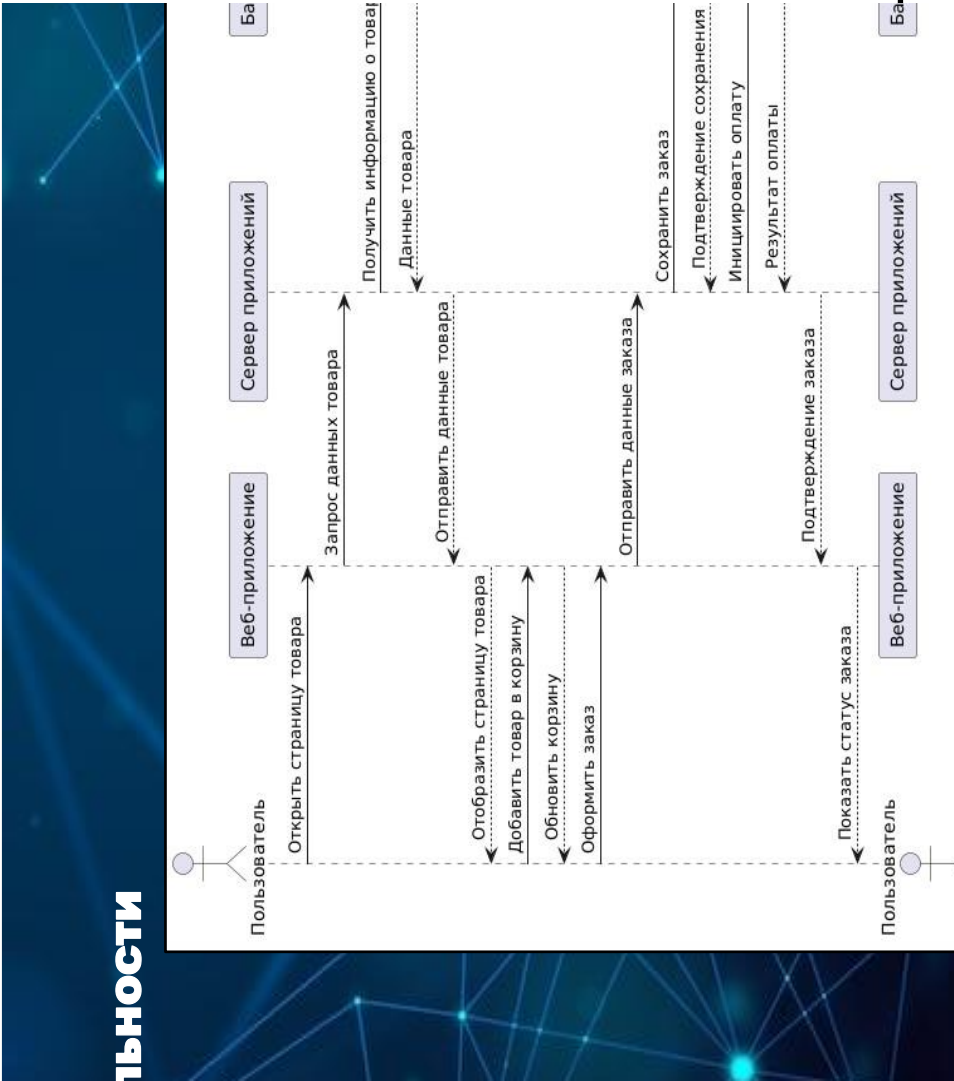
Смма тивания



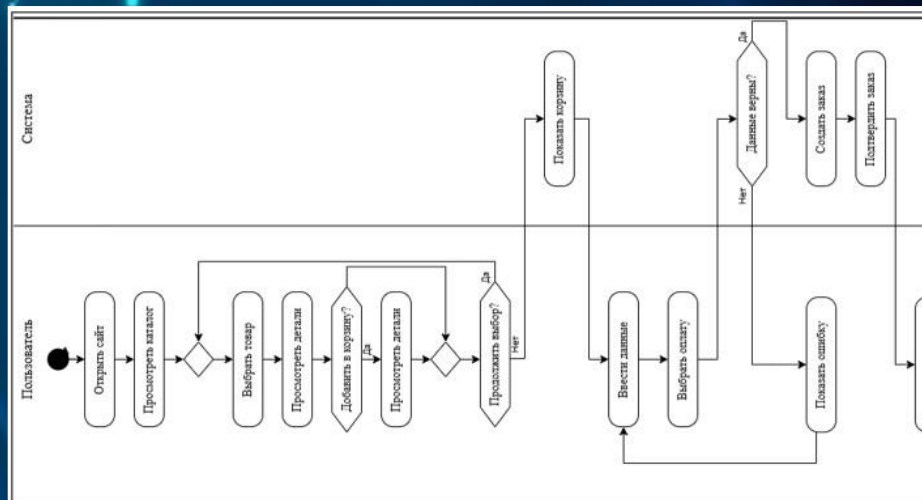
Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

Лист
123



Математическая модель



Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

Лист
125

Заключение

работки веб -приложения были реализованы все ключевые направления на создание удобного безопасного сервиса. Обеспечена интуитивная навигация, ск и простое оформление заказов. Парольных реализована путем хеширования паролей ей.

часть создана с использованием HTML, CSS, JavaScript что обеспечивает адаптивность интерфейса реализована на Node.js и Express.js с применением Для хранения данных используется нереляционная высокая производительность и масштабируемость соответствует современным требованиям, удобно и безопасно и готово к дальнейшему развитию.

Изм.	Лист	№ докум.	Подпись	Дата



Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.8.001 ПЗ

Лист
127