

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: «Программное обеспечение средств вычислительной
техники и автоматизированных систем»

БАКАЛАВРСКАЯ РАБОТА

на тему

КЛИЕНТ-СЕРВЕРНОЕ ПРИЛОЖЕНИЕ
«ОНЛАЙН-МАГАЗИН ФУТБОЛЬНОЙ
ЭКИПИРОВКИ»

Студент

30.05.25 Щеглов Дмитрий Алексеевич
(подпись, дата)

Руководитель

30.05.25 Акифьев Илья Владимирович
(подпись, дата)

Нормоконтролёр

30.05.25 Бычков Андрей Станиславович
(подпись, дата)

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

Заведующий кафедрой

30.05.25
(подпись)

М.А. Митрохин
(фамилия, инициалы)

Работа защищена с отметкой хорошо (протокол заседания ГЭК от
16 июня 2025 № 39)

Секретарь ГЭК

30.05.25
(подпись)

Е.И. Гурин
(фамилия, инициалы)

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 » мая 2025г.

**ЗАДАНИЕ
НА БАКАЛАВРСКУЮ РАБОТУ**

1. Студент Щеглов Дмитрий Алексеевич гр.21ВВП1 факультета вычислительной техники

2. Тема работы: веб - приложение «Онлайн-магазин футбольной экипировки»

Тема утверждена приказом ПГУ № 601/о от "06" мая 2025 г.

3. Руководитель бакалаврской работы к.э.н., доцент кафедры ВТ Акифьев И.В.

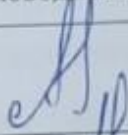
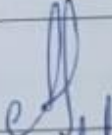
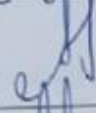
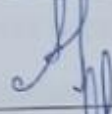
4. Задание на работу (назначение разработки, исходные данные и т.п.)

Разработать веб - приложение «Онлайн-магазин футбольной экипировки». Приложение должно обеспечивать пользователям удобный процесс выбора и покупки товаров. Предоставлять пользователям удобный каталог товаров, а администраторам инструменты для управления товарами и заказами. Приложение должно давать возможность выполнять регистрацию и авторизацию пользователя, изменять данные пользователя. Приложение должно быть построено по клиент-серверной архитектуре. Клиентская часть должна включать интуитивно понятный пользовательский интерфейс, функционал для взаимодействия пользователя с системой, адаптивный дизайн для понятного отображения на разных устройствах и механизмах отправки запросов к серверу. Серверная часть должна обеспечивать хранение и обработку данных в базе данных, их безопасное хранение, API для взаимодействия с клиентом и управлением доступом и правами пользователей.

5. Перечень подлежащих разработке вопросов

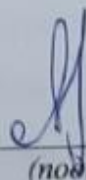
Требуется произвести анализ предметной области: изучить существующие аналоги, изучить способы проектирования и разработки веб - приложения. Также необходимо изучить технологий для разработки приложения, такие как Node.js, PostgreSQL, Express.js и т.д.

6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1.	Изучение предметной области	40	06.05.2025-10.05.2025	
2.	Анализ требований	16	11.05.2025-12.05.2025	
3.	Выбор технологий, применяемых при проектировании и разработке	24	13.05.2025-15.05.2025	
4.	Описание алгоритмов	24	16.05.2025-18.05.2025	
5.	Разработка веб-приложения «Онлайн-магазин футбольной экипировки»	80	19.05.2025-28.05.2025	
6.	Оформление пояснительной записки	16	29.05.2023-30.05.2025	

Дата выдачи задания "06" мая 2025 г.

Руководитель бакалаврской работы _____

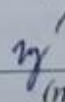


06.05.25

Акифьев И.В.

(подпись, дата)

Задание к исполнению принял студент _____



06.05.25

Щеглов Д.А.

(подпись, дата)

Работу к защите допустить
Декан факультета _____



16.06.2025

Фионова Л.Р.

Реферат

Бакалаврская работа содержит 102 листов, 25 рисунков, 3 таблицы, 7 источников и 3 приложения.

Клиент-серверное приложение интернет-магазин футбольной экипировки, Node.js, Express.js, PostgreSQL, Redis, Tailwind CSS, JWT, Nginx, HTML, CSS, JavaScript

Данная выпускная квалификационная работа посвящена разработке клиент-серверного приложения «Магазин футбольных принадлежностей», ориентированного на пользователей, желающих удобно и быстро выбирать товары для занятий футболом. Приложение предоставляет возможность просматривать каталог товаров, фильтровать их по категориям и цене, добавлять в корзину, оформлять заказы, а также управлять ассортиментом со стороны администратора.

Результатом разработки является проект, полностью соответствующий поставленной задаче. Было спроектировано, реализовано и протестировано клиент-серверное приложение «Магазин футбольных принадлежностей», предназначенное для удобного выбора и покупки товаров через интернет. Разработка велась с использованием современного стека технологий: язык программирования JavaScript (Node.js), фреймворк Express.js для организации серверной логики, система управления базами данных PostgreSQL с кэшированием данных через Redis, а также клиентская часть, реализованная с помощью HTML5, CSS3 и Vanilla JavaScript с применением утилитарного фреймворка Tailwind CSS. Для обеспечения безопасности были использованы JWT (JSON Web Token) и библиотека bcrypt, позволяющая хешировать пароли и разграничивать доступ между пользователями и администратором. Размещение приложения осуществлено с помощью веб-сервера Nginx, что

					ПГУ 1.090301.24.001 ПЗ				
Изм.	Лист	№ докум.	Подпись	Дата	Веб-приложение «Онлайн-магазин футбольной экипировки» Пояснительная записка	Лит.	Лист	Листов	
Разраб.		Щеглов Д.А.		30.05.25					
Провер.		Акифьев И.В.		30.05.25			2	102	
Н. Контр.		Бычков А. С.		30.05.25		ФВТ, каф. ВТ гр.21ВВП1			
Утверд.		Митрохин М.А.		30.05.25					

позволило организовать обработку статических файлов и проксирование запросов к API.

					<i>ПГУ 1.090301.24.001 ПЗ</i>	<i>Лист</i>
						5
<i>Изм.</i>	<i>Лист</i>	<i>№ докум.</i>	<i>Подпись</i>	<i>Дата</i>		

Содержание

Перечень сокращений и обозначений.....	8
Введение	10
1 Анализ предметной области	11
1.1 Актуальность темы.....	11
1.2 Клиент-серверная архитектура	11
1.3 Анализ аналогов интернет-магазинов	13
1.4 Постановка задачи.....	20
2 Выбор технологий и средств разработки.....	23
3 Архитектура онлайн магазина «FootballGear»	27
4 Реализация приложения	31
4.1 Описание работы приложения.....	31
4.2 Структура БД.....	33
4.3 Функционал приложения	33
4.4 Основные компоненты.....	36
4.4.1 База данных	37
4.4.2 Серверная часть	38
4.4.3 Клиентская часть	40
4.5 Основные алгоритмы взаимодействия клиента и сервера	43
4.5.1 Алгоритм регистрации и авторизации пользователя	43
4.5.2 Алгоритм просмотра каталога товаров и добавления в корзину.....	44
4.5.3 Алгоритм оформления заказа	45
4.5.4 Алгоритм управления товарами (для администратора)	45
4.5.5 Алгоритм работы с личным кабинетом	46
5 Тестирование приложения.....	48
5.1 Регистрация пользователя.....	48
5.2 Авторизация пользователя.....	50
5.3 Тестирование функционала взаимодействия с товарами.....	52
6 Безопасность жизнедеятельности.....	57

6.1 Требования к помещению и рабочему месту	57
6.2 Требования к освещению	58
6.3 Требования к режиму труда и отдыха.....	59
6.4 Рекомендации по использованию программного обеспечения	60
Заключение	62
Список использованных источников	63
Приложение А – UML-диаграммы.....	64
Приложение Б – Листинг программы	69
Приложение В – Презентация	100

Перечень сокращений и обозначений

В настоящей выпускной квалификационной работе применяют следующие сокращения и обозначения

БД – база данных

ПО – программное обеспечение

СУБД – система управления базой данных

API – (от англ. Application Programming Interface) – интерфейс взаимодействия между компонентами программного обеспечения

CSS – (от англ. Cascading Style Sheets) – язык описания внешнего вида документа, используемый для стилизации веб-страниц

Express.js – минимальный и гибкий веб-фреймворк для Node.js, предоставляющий набор инструментов для построения серверных и API-приложений

HTML – (от англ. HyperText Markup Language) – язык гипертекстовой разметки, применяемый для создания веб-страниц

HTTP – (от англ. Hypertext Transfer Protocol) – протокол прикладного уровня передачи гипертекстовых данных

HTTPS – (от англ. HyperText Transfer Protocol Secure) – защищённая версия протокола HTTP, обеспечивающая шифрование передаваемых данных

JWT – (от англ. JSON Web Token) – стандарт для передачи данных между сторонами в виде JSON-объекта, используется для аутентификации и авторизации

MVC – (от англ. Model-View-Controller) – шаблон программирования, разделяющий логику приложения на три модуля: модель, представление и контроллер

Node.js – программная платформа, позволяющая запускать JavaScript-приложения на сервере

Redis – (от англ. Remote Dictionary Server) – in-memory хранилище данных с открытым исходным кодом, поддерживающее работу с ключами и значениями

REST – (от англ. Representational State Transfer) – стиль проектирования веб-сервисов, основанный на использовании HTTP-методов

SQL – (от англ. Structured Query Language) – язык структурированных запросов, используемый для работы с реляционными базами данных

Tailwind CSS – утилитарный CSS-фреймворк, ориентированный на создание пользовательских интерфейсов непосредственно в HTML-разметке

UML – (от англ. Unified Modeling Language) – унифицированный язык моделирования, применяемый при проектировании программных систем

					<i>ПГУ 1.090301.24.001 ПЗ</i>	Лист
						9
Изм.	Лист	№ докум.	Подпись	Дата		

Введение

С каждым годом технологии всё глубже проникают в повседневную жизнь человека, и сфера торговли не остаётся в стороне. Онлайн-магазины становятся популярным способом совершения покупок, позволяя находить и приобретать товары, не выходя из дома. Особенно это актуально для спортивных товаров, где важны разнообразие ассортимента, удобство интерфейса и простота навигации.

Клиент-серверное приложение интернет-магазина футбольной атрибутики предоставляет пользователям возможность просматривать каталог, выбирать нужные товары, оформлять заказы и отслеживать их статус. Для владельцев бизнеса такая система упрощает управление товарами, заказами и клиентскими данными, что положительно сказывается на эффективности работы.

Поэтому создание подобного решения – задача, имеющая практическую ценность. Оно помогает автоматизировать ключевые процессы, повысить качество обслуживания покупателей и расширить географию продаж. Это делает разработку клиент-серверного приложения «Магазин футбольных принадлежностей» значимой и востребованной в современных условиях.

					<i>ПГУ 1.090301.24.001 ПЗ</i>	<i>Лист</i>
						10
<i>Изм.</i>	<i>Лист</i>	<i>№ докум.</i>	<i>Подпись</i>	<i>Дата</i>		

1 Анализ предметной области

1.1 Актуальность темы

Технологии сегодня активно влияют на образ жизни и привычки людей, особенно в вопросах совершения покупок. Если раньше человек отправлялся в магазин или на рынок, то теперь большинство решений принимается онлайн, через интернет-магазины, где можно найти любые товары – от одежды до бытовой техники и продуктов питания.

Популярность набирают онлайн-магазины, ориентированные на спорт, ведь физическая активность становится частью повседневной жизни многих. Особое место занимает футбол – один из самых массовых видов спорта в мире. Он объединяет болельщиков, формирует культуру, связанную с клубами, командами и символикой. Покупки спортивной одежды, обуви и аксессуаров позволяют людям чувствовать себя частью этой большой футбольной общности.

Создание интернет-магазина футбольных товаров представляет собой не просто интересный проект, но и полезное решение для бизнеса. Такое приложение позволяет продавать продукцию в любое время, охватывать широкую целевую аудиторию и предоставлять клиентам комфортный способ выбора товаров. Владелец получает инструмент для управления ассортиментом, заказами и взаимодействием с покупателями.

Развитие веб-технологий и появление надёжных фреймворков дают возможность строить такие системы с учётом требований безопасности, масштабируемости и удобства использования. Это открывает возможности для дальнейшего развития проекта: создания мобильных версий, интеграции с системами доставки и другими сервисами.

1.2 Клиент-серверная архитектура

Создание любого полноценного веб-приложения начинается с выбора архитектуры. В данном случае речь идёт о клиент-серверной модели, которая уже

					<i>ПГУ 1.090301.24.001 ПЗ</i>	Лист
						11
Изм.	Лист	№ докум.	Подпись	Дата		

давно стала стандартом для большинства веб-систем. Эта модель предполагает чёткое разделение ролей между клиентом (браузером пользователя) и сервером (удалённой машиной, которая обрабатывает запросы). Именно такое построение обеспечивает стабильность, производительность и простоту масштабирования.

Представим себе, как работает эта система. Допустим, пользователь заходит на сайт магазина, просматривает каталог, выбирает товар, добавляет его в корзину и оформляет заказ. Все эти действия отправляются на сервер, который обрабатывает запросы, обращается к базе данных, проверяет наличие товара, записывает информацию о заказе, а затем возвращает результат обратно клиенту в виде обновлённой страницы или сообщения.

Такой подход имеет ряд важных преимуществ:

- централизация данных: вся информация хранится на сервере, что исключает возможность случайного изменения или удаления со стороны клиента;
- разграничение прав доступа: можно легко организовать разные уровни доступа – например, обычный пользователь может просматривать товары и делать покупки, а администратор – добавлять, редактировать и удалять позиции;
- безопасность: благодаря тому, что логика обработки запросов сосредоточена на сервере, можно реализовать защиту от несанкционированного доступа, использовать шифрование паролей, токены и другие механизмы;
- масштабируемость: если потребуется увеличить количество функциональных возможностей или обслуживать больше пользователей, можно расширять серверную часть без значительного влияния на клиентскую;
- независимость от устройства: клиентская часть может быть реализована как веб-интерфейс, мобильное приложение или даже desktop-версия – всё это будет взаимодействовать с одной и той же серверной системой.

Таким образом, клиент-серверная архитектура – это не просто способ организации кода, а продуманная модель, которая позволяет строить сложные и надёжные системы, в том числе интернет-магазины, ориентированные на широкую аудиторию.

					<i>ПГУ 1.090301.24.001 ПЗ</i>	<i>Лист</i>
						12
<i>Изм.</i>	<i>Лист</i>	<i>№ докум.</i>	<i>Подпись</i>	<i>Дата</i>		

1.3 Анализ аналогов интернет-магазинов

С каждым годом на рынок выходят всё новые сервисы и приложения, которые предлагают пользователям более удобные способы совершать покупки, выбирать товары и управлять заказами. Одними из таких решений являются клиент-серверные приложения интернет-магазинов спортивного направления, которые позволяют пользователям просматривать каталог товаров, добавлять их в корзину, оформлять заказы и отслеживать статус доставки.

Для успешной разработки собственного решения необходимо провести анализ существующих аналогов. Это поможет выявить как положительные стороны уже реализованных систем, так и недостатки, которые можно устранить или улучшить. В данном пункте будут рассмотрены три популярных интернет-магазина, специализирующихся на продаже футбольной атрибутики:

- www.sportmaster.ru ;
- www.football-mania.ru ;
- www.decathlon.ru .

На рисунке 1 представлена шапка сайта sportmaster.ru

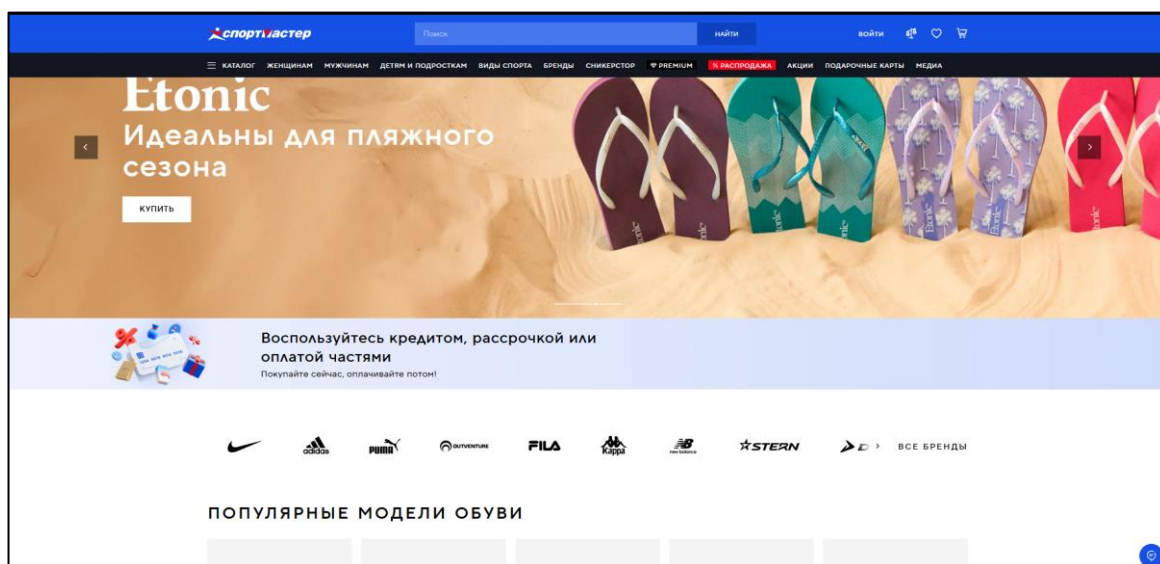


Рисунок 1 – Сайт sportmaster.ru

На рисунке 2 представлен каталог данного магазина

					ПГУ 1.090301.24.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		13

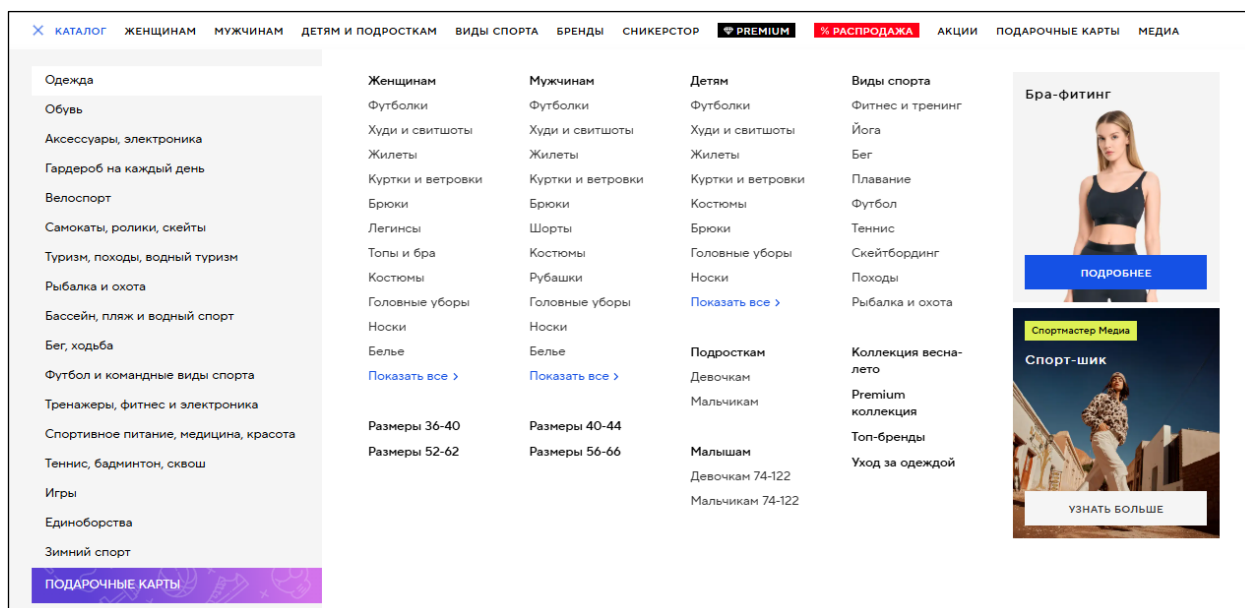


Рисунок 2 – Каталог магазина sportmaster

Интерфейс сайта представлен на рисунке 1 и 2. Особенностью данного магазина является широкий выбор спортивных товаров, включая профессиональную экипировку для футболистов. Сайт предлагает возможность выбора по брендам, фильтрации по размерам и наличию товара, а также мобильную версию сайта, которая делает процесс покупки доступным вне зависимости от устройства.

Интернет-магазин Sportmaster предоставляет своим пользователям следующие возможности:

- поиск товаров – пользователь может находить нужную продукцию по названию, категории, цене, бренду и другим параметрам;
- фильтрация – позволяет сужать выборку по таким критериям, как цена, бренд, наличие и другие;
- добавление товаров в корзину – удобный функционал, где можно менять количество и удалять товары;
- оформление заказа – интуитивно понятный интерфейс с возможностью указания способа доставки и оплаты;

- личный кабинет – здесь пользователь может просматривать историю своих заказов, изменять данные профиля и сохранять адреса доставки.

Однако, несмотря на хорошие технические показатели и широкую известность бренда, сайт имеет ряд недостатков:

- устаревший дизайн – внешний вид сайта не выглядит современным, особенно при сравнении с новыми цифровыми платформами;

- низкая скорость загрузки страниц – особенно это заметно при работе с большими каталогами;

- ограниченная история заказов – нет детального описания состава каждого заказа;

- отсутствие push-уведомлений – пользователь не получает автоматических сообщений о статусе заказа.

На рисунке 3 представлена шапка сайта football-mania.ru

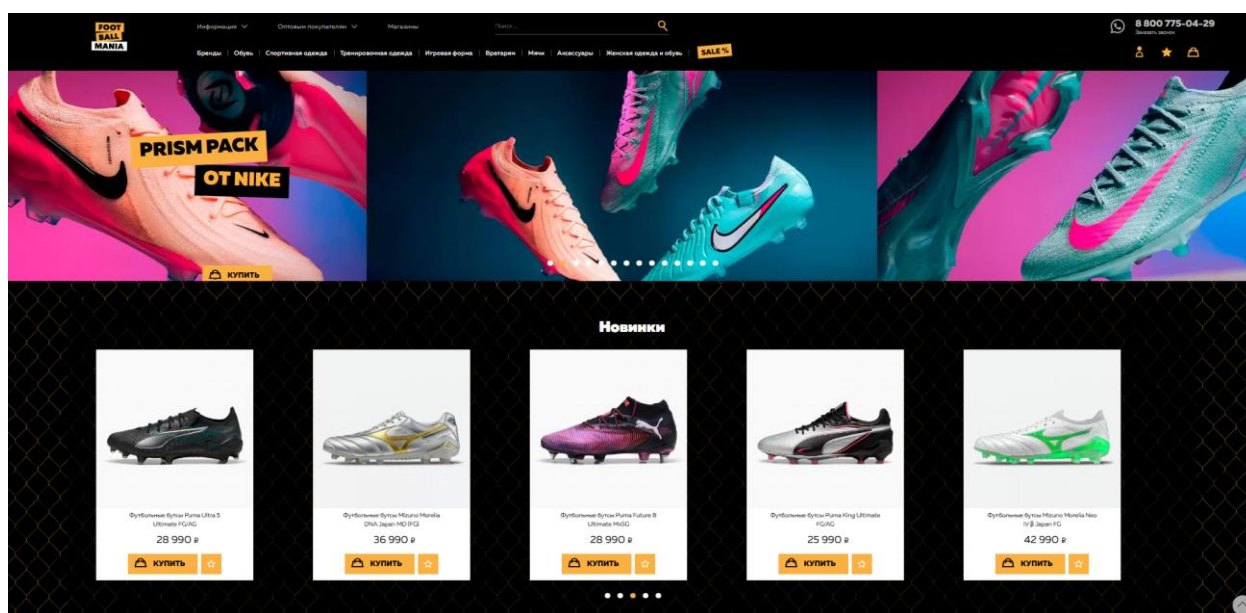


Рисунок 3 – Главная страница сайта www.football-mania.ru

На рисунке 4 представлена карточка товара

					ПГУ 1.090301.24.001 ПЗ	Лист
						15
Изм.	Лист	№ докум.	Подпись	Дата		

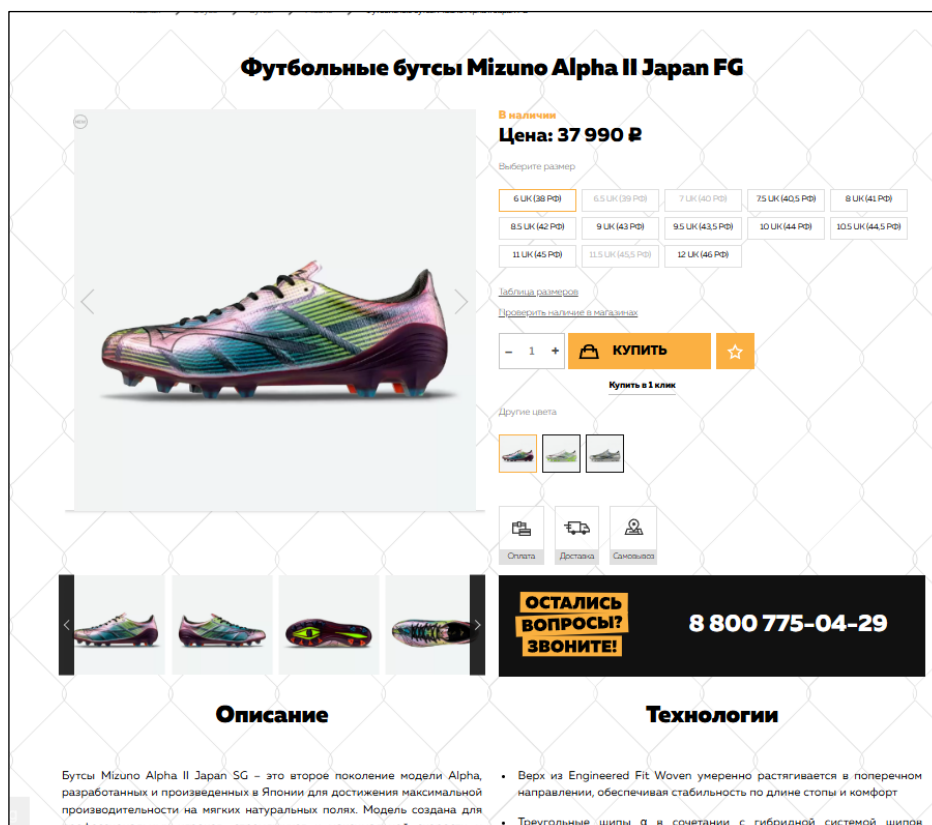


Рисунок 4 – Страница товара

Интерфейс онлайн-магазина представлен на рисунке 3 и 4. Отличительной особенностью этого сервиса является его специализация исключительно на футбольной продукции. Здесь можно найти форму команд, мячи, защитное снаряжение, гетры, перчатки и многое другое. Также стоит отметить наличие мобильного приложения, которое делает использование сайта ещё более удобным.

Футбольный магазин FootballMania предоставляет своим пользователям следующий функционал:

- современный дизайн – выполнение в футбольной тематике делает сайт визуально привлекательным;
- разнообразные варианты оплаты – банковские карты, Apple Pay, Google Pay, оплата частями;
- поддержка мобильных устройств – есть отдельное приложение для iOS и Android;

- персонализация – возможность нанесения имён, номеров и логотипов на форму;

- каталог с подробными описаниями – каждый товар содержит информацию о материале, размерах, отзывах и рейтинге.

Тем не менее, были выявлены и некоторые минусы:

- сложности с навигацией – поиск требует дополнительных действий, чтобы добраться до нужной позиции;

- высокая стоимость некоторых товаров – цены зачастую выше среднерыночных;

- отсутствие системы рекомендаций – нельзя получить предложения на основе предыдущих покупок;

- ограниченная фильтрация – нельзя фильтровать товары по лигам, сезонам или позициям игроков.

На рисунке 5 представлена главная страница сайта desport.ru

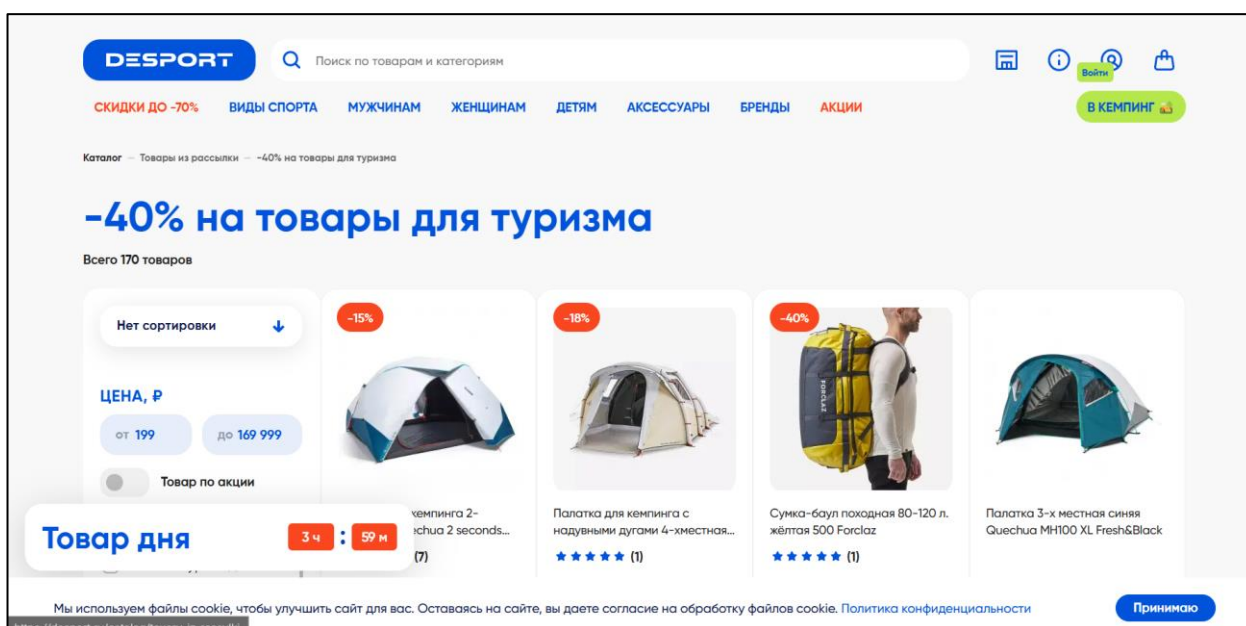


Рисунок 5 – Главная страница сервиса www.desport.ru

На рисунке 6 представлена карточка товара

					ПГУ 1.090301.24.001 ПЗ	Лист
						17
Изм.	Лист	№ докум.	Подпись	Дата		

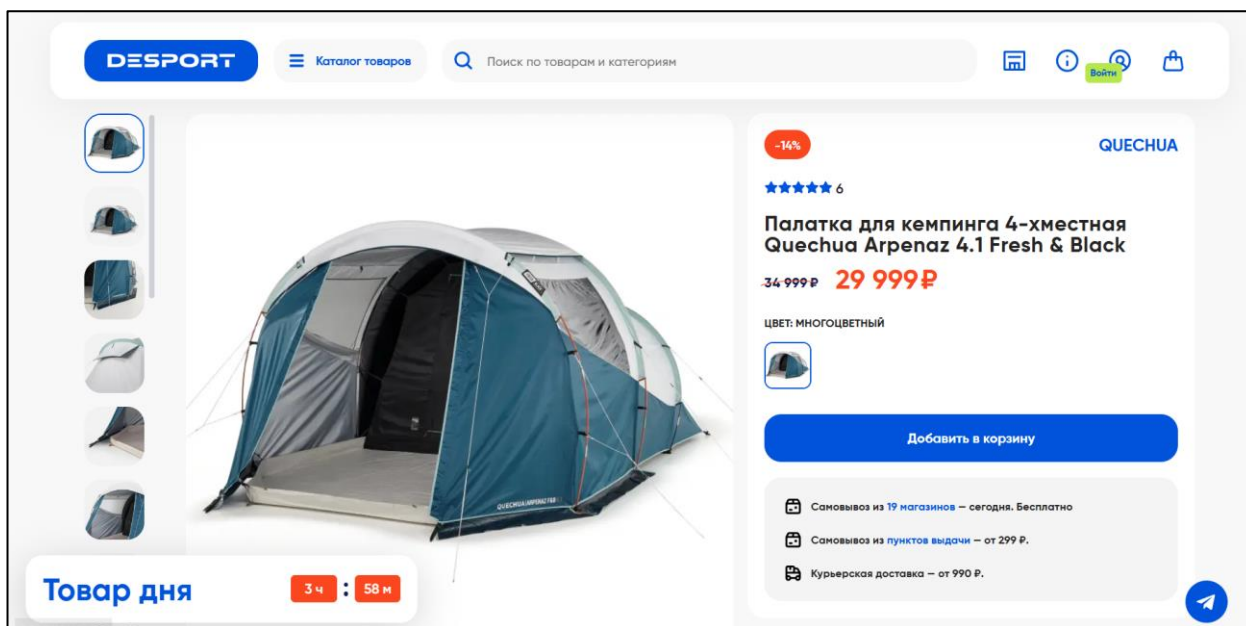


Рисунок 6 – Просмотр товара

Интерфейс сайта представлен на рисунке 5 и 6. Decathlon – международная сеть спортивных товаров, которая активно развивается в России. Сайт отличается простым, но эффективным подходом к электронной коммерции, предлагая большой выбор товаров по доступным ценам.

Decathlon предоставляет следующие возможности:

- широкий выбор товаров – от формы до тренировочного оборудования;
- прозрачная система акций и скидок – регулярные распродажи и бонусы;
- пункты самовывоза – возможность выбрать ближайший пункт получения;
- детальное описание товаров – информация о материале, размерах, условиях эксплуатации;
- обзоры и отзывы – покупатели могут оставлять свои комментарии и оценки.

Однако, как и другие сайты, Decathlon имеет свои ограничения:

- не очень выраженная футбольная направленность – хотя товары есть, они не выделены в отдельный раздел;
- минимальная персонализация – нет рекомендаций, нет возможности создать список желаемого;

- ограниченная аналитика – нет возможности отследить популярные товары по лигам или сезонам.

Результаты анализа существующих решений можно представить в виде таблицы, в которой собраны основные функции и особенности рассмотренных сайтов:

Таблица 1–Достоинства и недостатки онлайн-магазинов

Функционал	sportsmaster.ru	football-mania.ru	decathlon.ru
Тематическая направленность	Общий спортивный магазин	Узкоспециализированный (футбол)	Общий спортивный магазин
Ассортимент	Широкий	Средний, целевой	Очень широкий
Мобильная версия / приложение	Только мобильная версия	Мобильная версия + отдельное приложение	Только мобильная версия
Дизайн	Более консервативный	Современный, футбольный стиль	Простой, нейтральный
Поиск и фильтрация	Хорошая	Средняя, требуется доработка	Отличная
Цены	Средние	Выше среднего	Бюджетные
Личный кабинет	Ограниченный функционал	Расширенный функционал	Расширенный функционал

Функционал	sportsmaster.ru	football-mania.ru	decathlon.ru
Поддержка пользователей	Чат, email, телефон	Email, телефон	Email, телефон, чат

На основе проведённого анализа было принято решение разработать клиент-серверное приложение «Магазин футбольных принадлежностей», которое объединит лучшие черты всех трёх аналогов:

- профессиональный подход к подбору товаров (как у Sportsmaster);
- современный и тематический интерфейс (как у FootballMania);
- удобство использования и бюджетный подход (как у Decathlon).

При этом в проекте будут реализованы такие элементы, которых не хватает у конкурентов:

- гибкая фильтрация;
- авторизация через JWT и безопасное хранение данных;
- возможность просмотра истории заказов с полной информацией;
- модуль управления корзиной через Redis для повышения скорости работы;
- админ-панель с аналитикой продаж и управлением товарами.

Таким образом, разработка клиент-серверного приложения «Магазин футбольных принадлежностей» актуальна и перспективна, поскольку позволит предложить пользователям более точный и ориентированный на интересы болельщиков и спортсменов интерфейс, а также обеспечит высокий уровень надёжности и безопасности при совершении покупок.

1.4 Постановка задачи

Для успешной реализации проекта необходимо четко определить функциональные требования и требования к аппаратной части.

Функциональные требования:

					ПГУ 1.090301.24.001 ПЗ	Лист
						20
Изм.	Лист	№ докум.	Подпись	Дата		

регистрация/авторизация пользователей:

1) реализация системы аутентификации на основе JWT (JSON Web Tokens);

2) хеширование паролей с использованием bcrypt для защиты данных пользователей;

3) разграничение прав доступа между обычными пользователями и администраторами;

просмотр каталога товаров:

1) отображение товаров с возможностью фильтрации по категориям (форма, обувь, аксессуары) и сортировки (по цене, популярности);

2) информативные карточки товаров с фотографиями, описанием и техническими характеристиками;

поиск и фильтрация товаров:

1) поиск товаров по названию или описанию;

2) фильтры по цене, категории и популярности;

корзина и оформление заказа:

1) возможность добавления товаров в корзину;

2) управление количеством товаров в корзине;

3) оформление заказа с указанием адреса доставки и способа оплаты;

админ-панель:

1) управление товарами (CRUD-операции);

2) просмотр статистики продаж;

3) настройка системы (например, добавление новых категорий товаров).

Требования к аппаратной части:

Для обеспечения стабильной работы системы необходимо соблюдение следующих технических требований:

- процессор: минимум 4 ядра (рекомендуется 8 ядер для обработки большого количества запросов);

- ОЗУ: от 8 ГБ (рекомендуется 16 ГБ для серверной части);

					ПГУ 1.090301.24.001 ПЗ	Лист
						21
Изм.	Лист	№ докум.	Подпись	Дата		

- дисковое пространство: от 50 ГБ (для хранения базы данных и файлов);
- сетевое подключение: минимум 100 Мбит/с (рекомендуется 1 Гбит/с для высокой скорости отклика);
- операционная система: Linux (Ubuntu или CentOS) для серверной части;
- СУБД: PostgreSQL 17 для хранения структурированных данных;
- кэширование: Redis для ускорения работы часто запрашиваемых данных.

					ПГУ 1.090301.24.001 ПЗ	Лист
						22
Изм.	Лист	№ докум.	Подпись	Дата		

2 Выбор технологий и средств разработки

В этом разделе будут рассмотрены технологии, язык программирования и среда разработки, которые были выбраны для написания клиент-серверного приложения «Магазин футбольных принадлежностей».

Технологии и средства разработки:

- Node.js;
- Express.js;
- PostgreSQL;
- pgAdmin4;
- Redis;
- HTML5 / CSS3 / Tailwind CSS;
- Vanilla JavaScript;
- JWT (JSON Web Token);
- Bcrypt;
- Nginx;
- Visual Studio Code.

Для реализации серверной части проекта была выбрана платформа Node.js – среда выполнения кода на базе движка V8, позволяющая использовать язык JavaScript вне браузера. Это дало возможность применить единый язык программирования как для клиентской, так и для серверной частей, что упрощает взаимодействие между ними и делает процесс разработки более согласованным.

Одной из ключевых особенностей Node.js является его асинхронная, событийная модель, которая позволяет эффективно обрабатывать множество параллельных запросов. Такая способность особенно важна в случае интернет-магазина, где одновременно может быть зарегистрировано большое количество пользователей. Благодаря огромному количеству готовых модулей в репозитории npm, можно быстро добавлять новые функциональные возможности без необходимости писать всё с нуля.

					ПГУ 1.090301.24.001 ПЗ	Лист
						23
Изм.	Лист	№ докум.	Подпись	Дата		

Для построения серверной логики был выбран фреймворк Express.js – легковесный и гибкий инструмент, работающий поверх Node.js. Он предоставляет минимальный, но достаточный набор возможностей для создания веб-приложений и API, что делает его одним из самых популярных решений среди разработчиков.

Express.js используется для обработки HTTP-запросов (GET, POST, PUT, DELETE), маршрутизации URL и управления потоком данных. Также он поддерживает использование middleware – промежуточных функций, необходимых для реализации проверки данных, аутентификации, логирования и других задач. Простота и высокая скорость работы позволили сосредоточиться именно на бизнес-логике, не тратя время на настройку самого сервера.

Для хранения информации о пользователях, товарах, заказах и других сущностях была выбрана система управления базами данных PostgreSQL – мощная и надёжная СУБД с открытым исходным кодом. Она предоставляет широкие возможности для работы с данными, включая полную поддержку SQL, механизм транзакций, работу с JSON и другие современные функции.

Использование PostgreSQL дало возможность организовать централизованное хранение данных и эффективно управлять ими. Эта система хорошо зарекомендовала себя в проектах с высокими требованиями к безопасности и целостности информации. Благодаря своей гибкости и мощному функционалу, PostgreSQL стала отличным выбором для хранения данных интернет-магазина.

Для администрирования базы данных и работы с таблицами была использована графическая оболочка PgAdmin4 – удобный и функциональный инструмент для управления PostgreSQL.

PgAdmin4 предоставляет интерфейс, который позволяет создавать, редактировать и удалять таблицы, просматривать данные, выполнять SQL-запросы и анализировать производительность. Этот инструмент стал

незаменимым помощником при проектировании и тестировании структуры базы данных, а также при последующем управлении содержимым таблиц.

Для повышения скорости работы с часто запрашиваемыми данными было внедрено Redis – in-memory хранилище данных, поддерживающее работу с ключ-значение. Redis применялся для кэширования часто используемых данных, таких как категории товаров или популярные предложения.

Использование Redis позволило значительно ускорить обработку запросов, уменьшить нагрузку на основную базу данных и повысить отзывчивость системы. Его высокая производительность и простота использования сделали Redis отличным решением для оптимизации работы интернет-магазина.

- Для реализации клиентской части был выбран стек:HTML5 – для структуры страниц;
- CSS3 – для стилового оформления;
- Tailwind CSS – утилитарный CSS-фреймворк, обеспечивающий быструю верстку без написания собственного CSS.

Tailwind CSS позволил создавать адаптивный дизайн, корректно отображающийся на устройствах любого размера, а также сохранять чистоту и структурированность стилей за счёт использования готовых классов. Это сделало процесс верстки более быстрым и менее трудозатратным.

Интерактивность страниц реализована с помощью Vanilla JavaScript (ES6+) – чистый JavaScript без использования крупных фреймворков, таких как React или Vue. Такой подход позволил сохранить легковесность клиентской части и упростить её поддержку.

С его помощью были реализованы формы регистрации и авторизации, работа с корзиной, динамическое обновление данных без перезагрузки страницы, а также анимация элементов интерфейса. Это дало возможность быстро протестировать функционал и вносить изменения без необходимости разбираться в сложных механизмах фреймворков.

Для обеспечения безопасности и контроля доступа к различным частям приложения была внедрена технология JWT (JSON Web Token) – стандарт, позволяющий безопасно передавать данные между клиентом и сервером в виде закодированного токена.

JWT позволяет ограничивать доступ к защищённым маршрутам, передавать данные о пользователе и отказаться от необходимости хранить сессии на сервере. После успешной авторизации выдаётся токен, который должен передаваться вместе с каждым последующим запросом.

Для защиты паролей пользователей применялась библиотека bcrypt, реализующая алгоритм хеширования с солью. Пароли никогда не хранятся в открытом виде, а преобразуются в хеш с использованием алгоритма bcrypt, что делает их практически невозможными для расшифровки даже в случае утечки базы данных.

Для размещения приложения на удалённом сервере был выбран Nginx – высокопроизводительный веб-сервер и обратный прокси. Nginx выполнял несколько важных функций:

- обслуживание статических файлов, таких как HTML, CSS, JS и изображения;
- проксирование запросов к серверному API;
- настройка SSL/TLS для безопасного HTTPS-соединения.

Работа с Nginx позволила сделать размещение приложения надёжным, производительным и безопасным. Также он обеспечил возможность масштабирования приложения в будущем, если потребуется увеличить количество пользователей или добавить дополнительные модули.

В качестве основного редактора кода был выбран Visual Studio Code – мощная и гибкая среда разработки с поддержкой множества языков программирования и плагинов.

VS Code обеспечил комфортную работу с кодом: автодополнение, подсветка синтаксиса, встроенная отладка, поддержка Git и терминал. Все эти

функции положительно повлияли на скорость и качество разработки. Благодаря своей открытости и широким возможностям, VS Code стал идеальным выбором для реализации проекта.

3 Архитектура онлайн магазина «FootballGear»

Веб-приложение «Магазин футбольных принадлежностей» было разработано с использованием современных подходов к созданию веб-приложений, основанных на разделении клиентской и серверной частей. Обмен данными между ними осуществляется через чётко определённые интерфейсы, что обеспечивает гибкость, простоту поддержки и возможность расширения системы.

При построении архитектуры была взята за основу модель MVC (Model – View – Controller) , несмотря на то, что клиентская часть представлена одним HTML-документом – index.html. Такой подход позволил выделить логику отображения, управления и данных, сохранив при этом масштабируемость решения и удобство дальнейшей модификации.

Серверная часть была реализована с использованием платформы Node.js и фреймворка Express.js , что дало возможность организовать REST API для обмена данными между клиентом и сервером. Сервер выполняет функции контроллера и модели одновременно: он принимает запросы, маршрутизирует их и работает с базой данных. Для хранения информации о пользователях, товарах и заказах применялась система управления реляционными базами данных PostgreSQL , которая обеспечивает целостность, безопасность и поддержку сложных SQL-запросов.

Для ускорения работы с часто запрашиваемыми данными была внедрена технология Redis – инструмент, используемый в качестве кэширующего хранилища. Redis позволяет временно хранить популярные категории товаров и снижает нагрузку на основную базу данных, повышая общую отзывчивость системы.

					ПГУ 1.090301.24.001 ПЗ	Лист
						27
Изм.	Лист	№ докум.	Подпись	Дата		

Клиентская часть выполнена в виде одного файла index.html, где с помощью библиотек Tailwind CSS и Vanilla JavaScript динамически формируется содержимое страницы. Такой подход позволил реализовать поведение, напоминающее одностраничное приложение (SPA), без необходимости использования тяжёлых фреймворков. Интерфейс состоит из нескольких разделов: главная страница, каталог товаров, корзина, профиль пользователя и административная панель. Переход между разделами осуществляется без перезагрузки страницы, благодаря чему пользователь получает более плавный и комфортный опыт.

Общение между клиентом и сервером происходит через HTTP-запросы, соответствующие принципам REST API. Например, при добавлении товара в корзину клиент отправляет POST-запрос на маршрут /api/cart/add с указанием ID товара и количества. Сервер принимает этот запрос, проверяет наличие JWT-токена для авторизации, обновляет данные о корзине в Redis или БД и возвращает клиенту актуальное состояние корзины. На основе полученного ответа JavaScript обновляет интерфейс, информируя пользователя о том, что товар успешно добавлен.

На сервере обработка запросов построена по принципам MVC: контроллеры принимают входящие HTTP-запросы, передают их сервисному слою, который взаимодействует с моделью данных. Модели представляют собой объекты, соответствующие таблицам в базе данных. Все эти элементы находятся в одном серверном приложении, что упрощает разработку и делает её более управляемой.

Для обеспечения безопасности были применены технологии JWT (JSON Web Token) и bcrypt. JWT используется для передачи данных о пользователе между клиентом и сервером, а также для ограничения доступа к защищённым маршрутам. После успешного входа в систему пользователю выдаётся токен, который должен быть отправлен с каждым последующим запросом. Bcrypt применяется для безопасного хранения паролей – они хешируются с

					ПГУ 1.090301.24.001 ПЗ	Лист
						28
Изм.	Лист	№ докум.	Подпись	Дата		

использованием алгоритма с солью, что делает их практически невозможными к восстановлению даже в случае утечки.

Проект был развёрнут на удалённом сервере под ОС Ubuntu 24.04 LTS , где в качестве обратного прокси использовался Nginx . Он отвечал за обслуживание статических файлов, маршрутизацию запросов к API и обеспечение безопасного HTTPS-соединения с помощью сертификатов Let's Encrypt . Это позволило добиться высокой производительности, надёжности и безопасности при работе приложения в реальных условиях.

Данные о пользователях, товарах и заказах хранятся в PostgreSQL , структура которой была спроектирована таким образом, чтобы максимально точно отражать бизнес-процессы интернет-магазина. Основные таблицы включают:

- users – информация о зарегистрированных пользователях;
- products – данные о товарах (название, цена, категория, описание);
- cart_items – временные записи о товарах в корзине;
- orders – оформленные заказы с деталями доставки и текущим статусом.

Эти таблицы связаны между собой через внешние ключи, что позволяет эффективно управлять данными и выполнять сложные выборки при необходимости. Для управления структурой базы данных и анализа данных использовалась графическая среда pgAdmin4 , предоставившая удобный интерфейс для создания, изменения и просмотра таблиц.

Клиентская часть включает все необходимые компоненты для полноценной работы интернет-магазина. Пользователь может просматривать товары, фильтровать их по цене, категории и популярности, добавлять в корзину, оформлять заказы и просматривать историю покупок. Администратор имеет дополнительные возможности: добавление, редактирование и удаление товаров, просмотр статистики продаж и управление аккаунтами пользователей.

Работа всей системы организована следующим образом: пользователь взаимодействует с интерфейсом, JavaScript формирует HTTP-запрос и отправляет

					ПГУ 1.090301.24.001 ПЗ	Лист
						29
Изм.	Лист	№ докум.	Подпись	Дата		

его на сервер. Сервер обрабатывает запрос, обращается к базе данных или кэшу, формирует JSON-ответ и отправляет его обратно клиенту. Полученные данные используются для обновления состояния интерфейса, что создаёт эффект мгновенной реакции на действия пользователя.

Таким образом, выбранная архитектура обеспечивает надёжную и безопасную работу интернет-магазина, позволяет масштабировать систему и легко внедрять новые функции. Все компоненты – от клиентской части до сервера и базы данных – работают согласованно.

					<p><i>ПГУ 1.090301.24.001 ПЗ</i></p>	Лист
						30
Изм.	Лист	№ докум.	Подпись	Дата		

4 Реализация приложения

Реализация клиент-серверного приложения «Магазин футбольных принадлежностей» включает в себя описание работы системы, структуры базы данных, функциональных возможностей, а также основных компонентов, обеспечивающих взаимодействие между клиентской и серверной частями. В рамках данной работы была разработана система, позволяющая пользователям просматривать каталог товаров, добавлять их в корзину, оформлять заказы, а администраторам – управлять ассортиментом магазина и отслеживать статистику продаж. Основные этапы реализации включали:

- разработку клиентской части на основе HTML5, CSS3 и Vanilla JavaScript;
- создание серверной части на Node.js с использованием Express.js;
- проектирование и настройку базы данных PostgreSQL;
- реализацию функционала регистрации, авторизации, добавления товаров и управления ими;
- тестирование всех модулей системы для обеспечения надёжности и безопасности.

4.1 Описание работы приложения

Приложение представляет собой интернет-магазин, в котором пользователи могут зарегистрироваться, войти в систему, выбрать товары, добавить их в корзину и оформить заказ. Клиентская часть реализована в одном HTML-файле (index.html) с использованием библиотеки Tailwind CSS для стилизации и чистого JavaScript для динамического обновления интерфейса без перезагрузки страницы.

					ПГУ 1.090301.24.001 ПЗ	Лист
						31
Изм.	Лист	№ докум.	Подпись	Дата		

Серверная часть приложения построена на базе Node.js и Express.js, где реализован REST API для обработки запросов от клиента. Все данные хранятся в PostgreSQL – мощной реляционной СУБД, которая обеспечивает целостность и безопасность информации. Для повышения производительности используется Redis, в котором хранятся популярные категории и товары.

Взаимодействие между клиентом и сервером происходит через HTTP-запросы, которые обрабатываются сервером, проверяются на соответствие прав доступа и выполняются в зависимости от роли пользователя. Например, обычный пользователь может просматривать каталог, добавлять товары в корзину и оформлять заказы, тогда как администратор имеет возможность управлять товарами, категориями и пользователями. На рисунке 7 представлена UML-диаграмма последовательности функционирования приложения

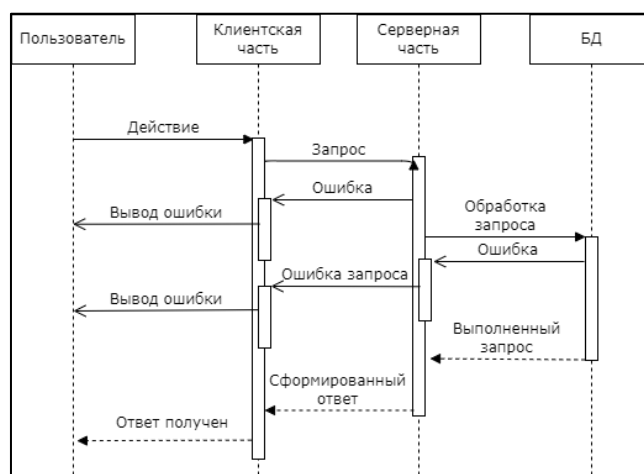
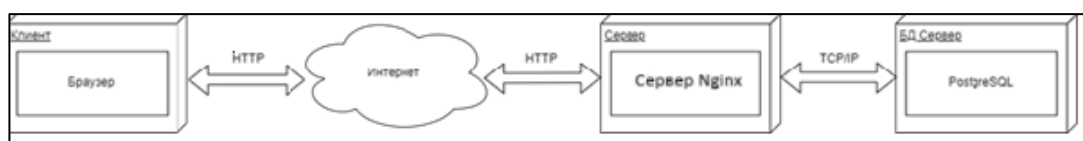


Рисунок 7 –UMLдиаграмма последовательности функционирования приложения

На рисунке 8 представлена UML-диаграмма развёртывания системы



4.2 Структура БД

Для хранения данных приложения была разработана реляционная модель базы данных, включающая следующие основные таблицы:

Таблица 2 - users (Пользователи)

Поле	Тип данных	Описание	Пример
id	SERIAL	Уникальный идентификатор	1
email	VARCHAR(100)	Электронная почта	user@example.com
first_name	VARCHAR(50)	Имя	Иван
last_name	VARCHAR(50)	Фамилия	Иванов
password	VARCHAR(255)	Хэшированный пароль	
created_at	TIMESTAMP	Дата регистрации	2023-01-01 12:00

Таблица 3 - products (Товары)

Поле	Тип данных	Описание	Пример
id	SERIAL	Уникальный идентификатор	1
name	VARCHAR(100)	Название товара	Футбольный мяч
category	VARCHAR(50)	Категория	Мячи
price	DECIMAL(10,2)	Цена	1999.99
image	VARCHAR(255)	Ссылка на изображение	/images/ball.jpg
description	TEXT	Описание товара	Официальный мяч
popular	BOOLEAN	Признак популярного товара	true

Эта структура позволяет гибко управлять ассортиментом, а также выполнять сложные запросы для фильтрации товаров по цене, категории или популярности. Для администрирования базы данных использовалась графическая оболочка pgAdmin4, предоставляющая удобный интерфейс для создания, изменения и анализа таблиц.

4.3 Функционал приложения

Приложение "FootballGear" предоставляет дифференцированный набор возможностей в зависимости от статуса пользователя. Рассмотрим подробно каждый уровень функциональности.

Для гостей сайта (неавторизованных пользователей) доступен базовый

функционал знакомства с ассортиментом магазина. Основная возможность – это просмотр каталога товаров с применением фильтров по категориям (футбольная форма, обувь, аксессуары) и сортировки по цене или популярности. Каждая карточка товара содержит фотографии, подробное описание и технические характеристики. Гости могут зарегистрироваться в системе или авторизоваться, если уже имеют аккаунт.

На рисунке 9 представлена UML диаграмма вариантов использования неавторизованного пользователя.

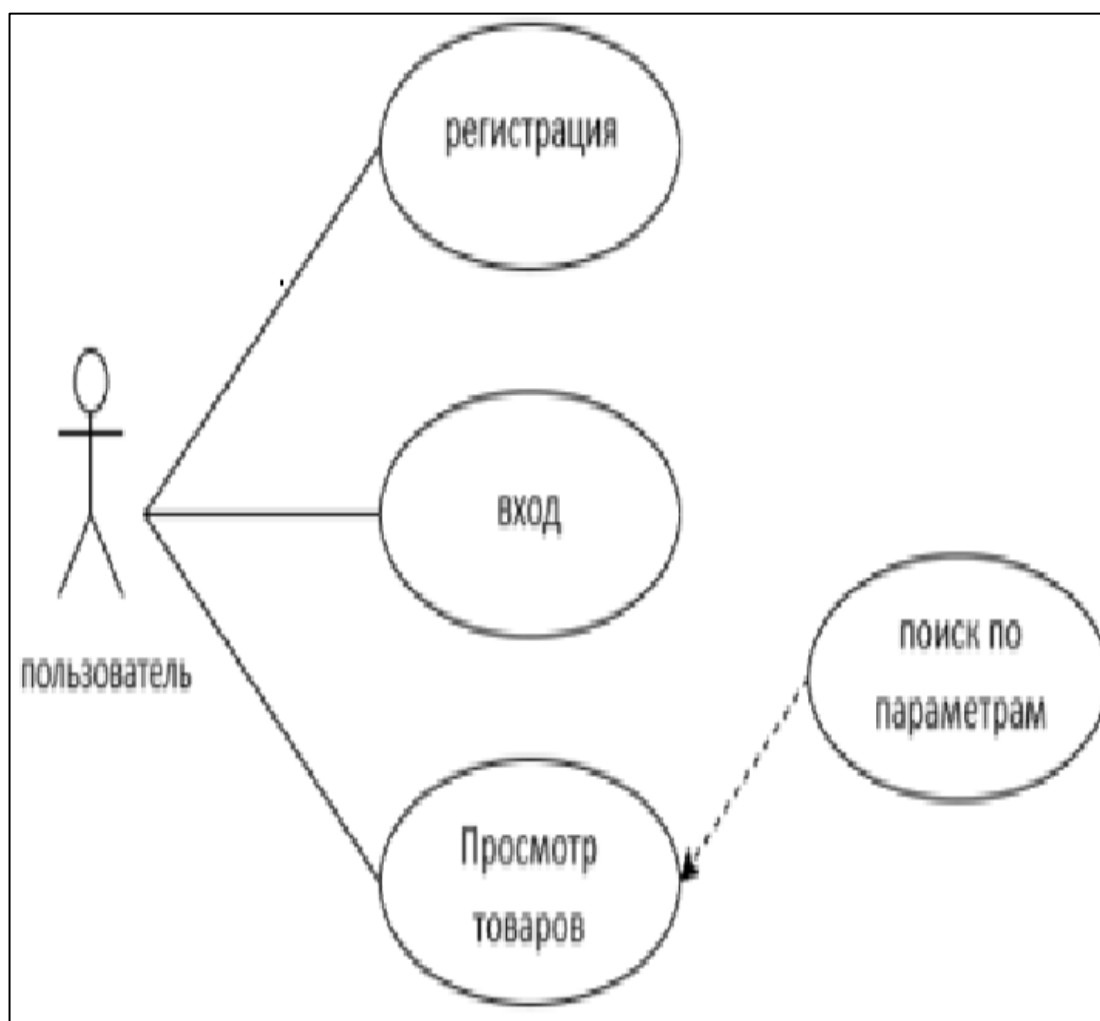


Рисунок 9 – UML диаграмма вариантов использования неавторизованного пользователя

После успешной авторизации пользователь получает расширенный набор

функций. Центральным элементом становится интерактивная корзина, где можно формировать заказ, изменять количество товаров или удалять позиции.

Пользователи могут:

- обновлять контактную информацию;
- добавлять несколько адресов доставки;
- изменять пароль;
- просматривать историю действий.

На рисунке 10 представлена UML диаграмма вариантов использования администратора

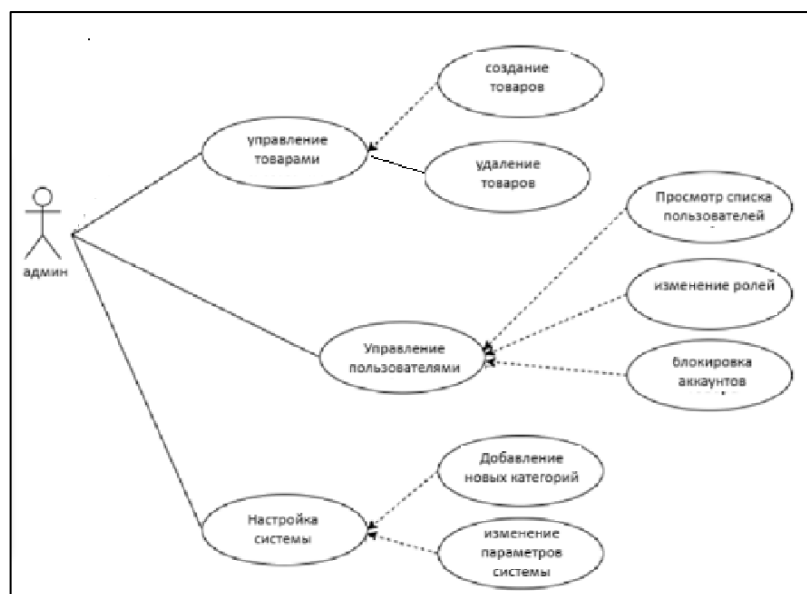


Рисунок 10 – UML диаграмма вариантов использования администратора

Диаграмма отображает основные функции администратора:

- управление товарами (CRUD операции);
- создание новых товаров;
- редактирование существующих товаров;
- удаление товаров из каталога;
- управление пользователями;
- просмотр списка пользователей;
- изменение ролей пользователей (например, назначение администратора);

- блокировка аккаунтов при необходимости;
- настройка системы;
- добавление новых категорий товаров;
- изменение параметров работы системы.

На рисунке 11 представлена UML диаграмма вариантов использования авторизованного пользователя.

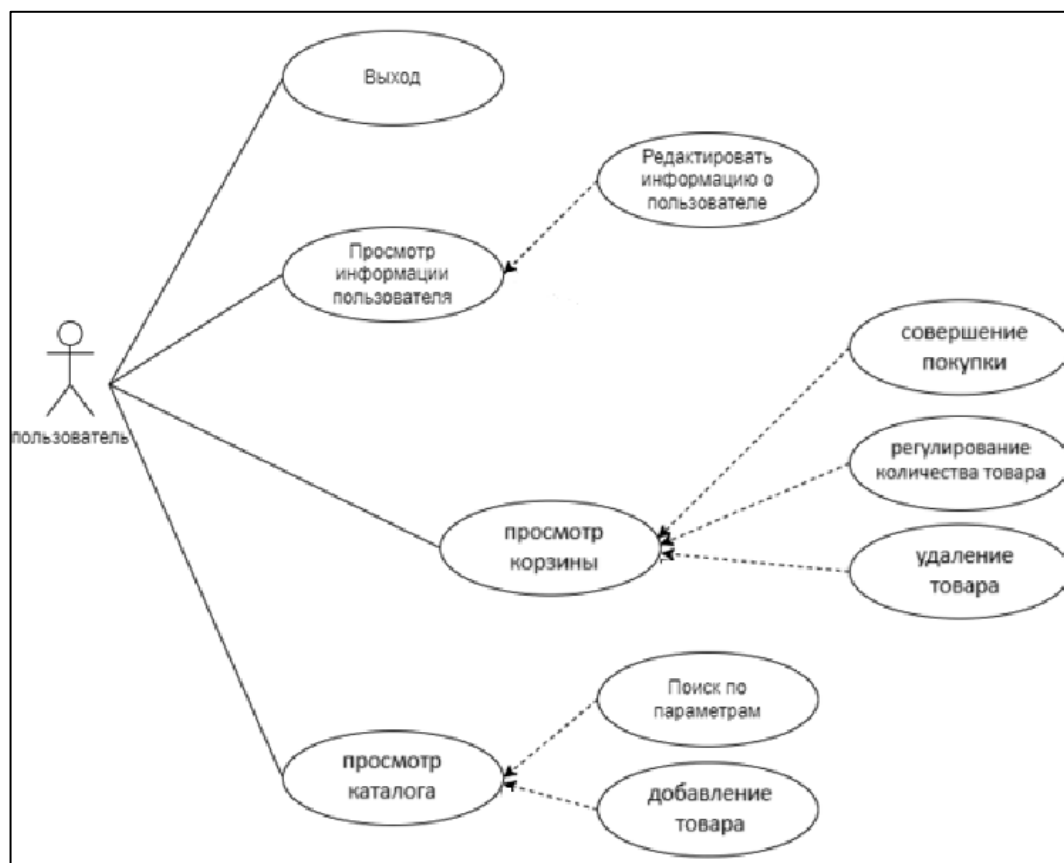


Рисунок 11 – UML диаграмма вариантов использования авторизованного пользователя

4.4 Основные компоненты

Сайт "FootballGear" представляет собой интернет-магазин футбольной экипировки и аксессуаров.

Он состоит из двух основных частей:

- серверная часть: реализована на Node.js с использованием Express и PostgreSQL;

- клиентская часть: написана на HTML, CSS и JavaScript;

система включает следующие ключевые компоненты:

- авторизация и регистрация пользователей;
- пользователи могут регистрироваться и входить в систему;
- для защиты маршрутов используется JWT-аутентификация.

админ-панель:

- доступна только для администраторов;
- позволяет управлять товарами и пользователями.

каталог товаров:

- отображает список доступных товаров;
- пользователи могут добавлять товары в корзину.

корзина:

- хранит выбранные товары;
- позволяет оформить заказ после авторизации.

4.4.1 База данных

База данных построена на PostgreSQL и включает две основные таблицы;

Далее представлена таблица пользователей.

```
CREATE TABLE users (  
    id SERIAL PRIMARY KEY,  
    name VARCHAR(255) NOT NULL,  
    email VARCHAR(255) UNIQUE NOT NULL,  
    password VARCHAR(255) NOT NULL,  
    role VARCHAR(50) DEFAULT 'user'  
);
```

Поля:

- id: уникальный идентификатор пользователя;
- name: имя пользователя;
- email: электронная почта (уникальное значение);

- password: хэшированный пароль;
- role: роль пользователя (user или admin).

Таблица товаров.

```
CREATE TABLE products (
  id SERIAL PRIMARY KEY,
  name VARCHAR(255) NOT NULL,
  category VARCHAR(255) NOT NULL,
  price DECIMAL(10, 2) NOT NULL,
  image_url TEXT NOT NULL,
  description TEXT,
  popular BOOLEAN DEFAULT FALSE
);
```

Поля:

- id: уникальный идентификатор товара;
- name: название товара;
- category: категория товара;
- price: цена товара;
- image_url: ссылка на изображение товара;
- description: описание товара;
- popular: флаг популярности товара.

4.4.2 Серверная часть

Серверная часть реализована на Node.js с использованием Express. Она обрабатывает запросы от клиентской части и взаимодействует с базой данных.

Маршрут для регистрации пользователя представлен ниже.

```
app.post('/api/register', async (req, res) => {
  try {
    const { name, email, password } = req.body;
    const hashedPassword = await bcrypt.hash(password, 10);
    const query = 'INSERT INTO users (name, email, password, role)
VALUES ($1, $2, $3, $4) RETURNING *';
    const values = [name, email, hashedPassword, 'user'];
    const { rows } = await pool.query(query, values);
```

```

        res.json({ message: 'Пользователь успешно зарегистрирован', user:
rows[0] });
    } catch (err) {
        console.error(err);
        res.status(500).json({ error: 'Ошибка сервера' });
    }
});

```

Далее показан маршрут для входа пользователя.

```

app.post('/api/login', async (req, res) => {
    try {
        const { email, password } = req.body;
        const user = await pool.query('SELECT * FROM users WHERE email =
$1', [email]);
        if (user.rows.length === 0) {
            return res.status(400).json({ error: 'Пользователь не найден'
});
        }
        const userRow = user.rows[0];

        // Проверка пароля
        const isPasswordValid = await bcrypt.compare(password,
userRow.password);
        if (!isPasswordValid) {
            return res.status(401).json({ error: 'Неверный пароль' });
        }
        const token = jwt.sign({ id: userRow.id, role: userRow.role },
JWT_SECRET, { expiresIn: '1h' });
        res.json({
            message: 'Вход выполнен успешно',
            token,
            user: {
                id: userRow.id,
                name: userRow.name,
                email: userRow.email,
                role: userRow.role
            }
        });
    }
});

```

```

} catch (err) {
    console.error(err);
    res.status(500).json({ error: 'Ошибка сервера' });
}
});

```

Middleware для аутентификации представлен ниже.

```

const authenticateToken = (req, res, next) => {
    const authHeader = req.headers['authorization'];
    const token = authHeader && authHeader.split(' ')[1];
    if (!token) return res.status(401).json({ error: 'Токен не предоставлен' });

    jwt.verify(token, JWT_SECRET, (err, user) => {
        if (err) return res.status(403).json({ error: 'Неверный токен' });
        req.user = user;
        next();
    });
};

```

4.4.3 Клиентская часть

Клиентская часть реализована с использованием HTML, CSS и JavaScript.

Фрагмент реализации шапки сайта.

```

<header class="bg-white shadow-sm sticky top-0 z-10">
    <div class="container mx-auto px-4 py-3 flex justify-between items-center">
        <div class="flex items-center space-x-2">
            <i class="fas fa-futbol text-blue-500 text-2xl"></i>
            <h1 class="text-xl font-bold text-gray-800">FootballGear</h1>
        </div>
        <nav class="hidden md:flex space-x-6">
            <a href="#" class="tab-link py-2 px-1 border-b-2 border-transparent hover:text-blue-500 transition" data-tab="home">Главная</a>
            <a href="#" class="tab-link py-2 px-1 border-b-2 border-transparent hover:text-blue-500 transition" data-tab="catalog">Каталог</a>

```

```

        <a href="#" class="tab-link py-2 px-1 border-b-2 border-
transparent hover:text-blue-500 transition" data-tab="account">Личный
кабинет</a>

        <a href="#" class="tab-link py-2 px-1 border-b-2 border-
transparent hover:text-blue-500 transition" data-tab="admin">Админ-
панель</a>

    </nav>

</div>

</header>

```

Форма добавления товара приведена ниже.

```

<div id="add-product-form" class="hidden">
    <div class="mb-4">
        <label for="product-name">Название</label>
        <input type="text" id="product-name" class="w-full px-3 py-2 border
rounded-md">
    </div>
    <button id="save-product" class="w-full bg-blue-500 hover:bg-blue-600
text-white py-2 rounded-md transition">
        Сохранить товар
    </button>
</div>

```

Обработчик сохранения товара представлена далее.

```

document.getElementById('save-product').addEventListener('click', async ()
=> {
    try {
        const token = localStorage.getItem('token');
        if (!token) {
            alert('Требуется авторизация');
            return;
        }
        const product = {
            name: document.getElementById('product-name').value,
            category: document.getElementById('product-category').value,
            price:
                parseFloat(document.getElementById('product-
price').value),
            image_url: document.getElementById('product-image').value,

```

```

        description: document.getElementById('product-
description').value,
        popular: document.getElementById('product-popular').checked
    };
    if (!product.name || !product.category || !product.price || !prod-
uct.image_url) {
        alert('Заполните все обязательные поля');
        return;
    }

    const response = await fetch('http://localhost:5000/api/products',
{
    method: 'POST',
    headers: {
        'Content-Type': 'application/json',
        'Authorization': `Bearer ${token}`
    },
    body: JSON.stringify(product)
});
    if (response.ok) {
        alert('Товар успешно добавлен');
        document.getElementById('add-product-
form').classList.add('hidden');
        renderAdminProducts();
    } else {
        const data = await response.json();
        alert(data.error || 'Ошибка при добавлении товара');
    }
} catch (error) {
    console.error(error);
    alert('Ошибка при отправке данных');
}
});

```

4.5 Основные алгоритмы взаимодействия клиента и сервера

4.5.1 Алгоритм регистрации и авторизации пользователя

Одним из ключевых алгоритмов работы системы является процесс регистрации и авторизации. Для использования всех возможностей интернет-магазина, таких как добавление товаров в корзину, оформление заказов и доступ к личному кабинету, пользователь должен зарегистрироваться и войти в систему. Если неавторизованный пользователь попытается воспользоваться функцией, которая доступна только после входа, ему будет предложено выполнить авторизацию.

Регистрация осуществляется через форму, где пользователь заполняет поля:

- имя;
- email;
- пароль;
- повтор пароля.

После нажатия кнопки "Зарегистрироваться" данные отправляются на сервер, где проводится их обработка. Сервер проверяет корректность ввода данных:

- должны быть заполнены все обязательные поля;
- email должен соответствовать формату электронной почты;
- пароли должны совпадать;
- пользователь с таким email не должен существовать в базе данных.

Если все проверки пройдены успешно, пароль хешируется с помощью bcrypt, а затем сохраняется вместе с остальными данными в таблице users. После этого пользователь перенаправляется на страницу авторизации.

В случае возникновения ошибок, таких как дублирование email или некорректный ввод данных, система возвращает соответствующее сообщение, которое отображается на клиентской части, и пользователю предлагается исправить введенную информацию и повторить регистрацию.

Авторизация пользователя происходит аналогично. Пользователь вводит свой email и пароль, которые отправляются на сервер. Сервер проверяет наличие пользователя в базе данных и сверяет введенный пароль с хешированным значением. Если данные совпадают, сервер генерирует JWT-токен, который передается обратно клиенту и сохраняется в браузере (например, в localStorage). Этот токен используется в дальнейшем для проверки прав доступа при работе с защищенными маршрутами.

Если данные введены неверно, сервер возвращает ошибку, которую интерфейс отображает пользователю. В этом случае вход не осуществляется, и пользователь может повторить попытку.

Эта последовательность действий обеспечивает безопасность и контроль доступа, а также позволяет четко разделить права между обычными пользователями и администраторами.

4.5.2 Алгоритм просмотра каталога товаров и добавления в корзину

Ещё одним важным алгоритмом является взаимодействие пользователя с каталогом товаров и корзиной. Каталог представляет собой список товаров, организованных по категориям: форма, обувь, аксессуары и т.д. Товары можно фильтровать по цене, популярности и другим параметрам.

На клиентской стороне, при загрузке страницы каталога, JavaScript отправляет GET-запрос на /api/products, чтобы получить актуальный список товаров. Сервер получает запрос, обращается к PostgreSQL, выбирает товары и возвращает их в виде JSON-ответа. На основе этих данных интерфейс динамически формирует карточки товаров.

При нажатии на кнопку «Добавить в корзину», клиентская часть отправляет POST-запрос на /api/cart/add с указанием идентификатора товара и количества. Сервер принимает запрос, проверяет наличие товара в базе данных, а также проверяет, авторизован ли пользователь. Если пользователь авторизован, его идентификатор извлекается из JWT-токена, и товар добавляется в корзину, которая хранится либо в Redis (для ускорения), либо в таблице cart_items.

					ИГУ 1.090301.24.001 ПЗ	Лист
						44
Изм.	Лист	№ докум.	Подпись	Дата		

После добавления товара в корзину, сервер возвращает обновлённое состояние корзины, которое отображается на клиенте – например, меняется количество товаров в иконке корзины или показывается обновлённый список.

Если произошла ошибка (например, товар не найден или пользователь не авторизован), сервер возвращает соответствующее сообщение, которое отображается в интерфейсе.

4.5.3 Алгоритм оформления заказа

Следующим важным алгоритмом является оформление заказа, которое возможно только для авторизованных пользователей. Когда пользователь решает купить выбранные товары, он переходит в корзину, где видит перечень добавленных позиций, их количество и общую стоимость.

При нажатии на кнопку «Оформить заказ» клиент отправляет POST-запрос на /api/orders/create, в теле которого содержится информация о товарах и адрес доставки. Сервер проверяет, есть ли такие товары в наличии, и если всё верно, создаёт запись в таблице orders и очищает корзину пользователя. Также в заказе указывается текущая цена каждого товара, чтобы исключить возможность изменения цены на стороне клиента.

После успешного создания заказа, пользователь получает подтверждение и может перейти в личный кабинет, где будет отображаться история его покупок. В случае ошибки (например, истечение срока действия токена или отсутствие товара) система информирует пользователя и предлагает повторить действие.

4.5.4 Алгоритм управления товарами (для администратора)

Администратор имеет расширенный доступ к функционалу магазина. Он может:

- добавлять новые товары;
- редактировать информацию о существующих товарах;
- удалять товары из каталога;
- просматривать список пользователей и изменять их роли;
- анализировать статистику продаж.

Процесс добавления нового товара начинается с заполнения формы на странице административной панели. Все данные отправляются на сервер через POST-запрос `/api/products/create`. Сервер проверяет корректность данных, валидирует их (например, обязательные поля, типы данных, уникальность названия), после чего создаёт новую запись в таблице `products`.

Аналогично реализованы редактирование и удаление товаров. При редактировании администратор вносит изменения в форму, и данные отправляются на сервер через PUT-запрос `/api/products/:id/edit`. При удалении – отправляется DELETE-запрос `/api/products/:id/delete`. Сервер обрабатывает эти запросы, вносит изменения в базу данных и возвращает клиенту обновлённый список товаров.

В случае если пользователь, пытающийся выполнить одно из этих действий, не имеет достаточных прав, сервер возвращает ошибку, и система блокирует выполнение операции. Это позволяет эффективно разграничивать права между обычными пользователями и администраторами.

4.5.5 Алгоритм работы с личным кабинетом

Личный кабинет доступен только авторизованным пользователям. Здесь они могут:

- посмотреть историю своих заказов;
- изменить контактные данные;
- добавить или изменить адрес доставки;
- просмотреть статусы заказов.

При переходе в личный кабинет клиент отправляет GET-запрос на `/api/user/me`, в котором в заголовке передаётся JWT-токен. Сервер проверяет токен, извлекает информацию о пользователе и возвращает данные, необходимые для отображения профиля.

Изменение данных профиля происходит через PUT-запрос `/api/user/update`, в котором отправляются обновлённые поля (например, имя или номер телефона).

					ПГУ 1.090301.24.001 ПЗ	Лист
						46
Изм.	Лист	№ докум.	Подпись	Дата		

Сервер проверяет введенные данные, обновляет информацию в таблице users и возвращает обновленный профиль.

					<i>ПГУ 1.090301.24.001 ПЗ</i>	<i>Лист</i>
						47
<i>Изм.</i>	<i>Лист</i>	<i>№ докум.</i>	<i>Подпись</i>	<i>Дата</i>		

5 Тестирование приложения

Для проверки работоспособности клиент-серверного приложения «FootballGear» необходимо провести его тестирование.

Тестирование будет состоять из следующих действий:

- регистрация пользователя;
- авторизация пользователя;
- просмотр, добавление в корзину и покупка товара;
- редактирование и удаление аккаунта пользователя;
- выход из аккаунта.

5.1 Регистрация пользователя

Для проведения тестирования регистрации пользователя, необходимо нажать на кнопку «Войти» в правой верхней части страницы, и ввести следующие запрашиваемые данные: имя пользователя, email и пароль. После введения данных и выполнения запроса на добавление пользователя, в базе данных должна появиться запись о новом пользователе.

Проверка начального состояния сущности users в базе данных представлена на рисунке 12.

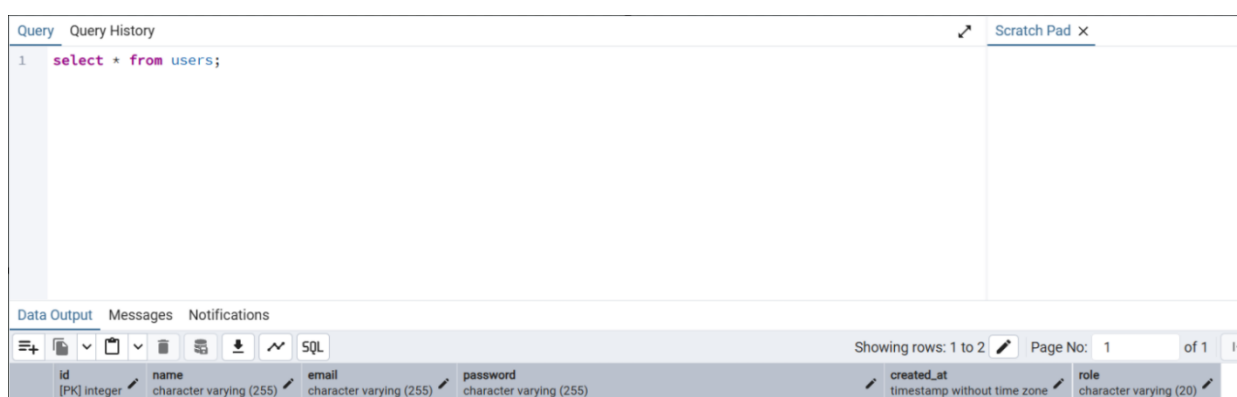


Рисунок 12 – База данных до добавления пользователя

Ввод данных в форму для регистрации приведен на рисунке 13.

Регистрация

Имя

Дима

Email

cacka.sss@mail.ru

Пароль

.....

Зарегистрироваться

Уже есть аккаунт? Войти

Рисунок 13 – Регистрация пользователя

Проверка состояния сущности users в базе данных после регистрации представлена на рисунке 14.

Query

Query History

↗

Scratch Pad ×

1

select * from users;

Data Output

Messages

Notifications

+

📄

▼

📁

▼

🗑

📦

📥

📶

SQL

Showing rows: 1 to 1

Page No: 1

o

id	name	email	password	created_at	role
[PK] integer	character varying (255)	character varying (255)	character varying (255)	timestamp without time zone	character varying (20)
1	Дима	cacka.sss@mail.ru	\$2b\$10\$10DwmhIDxe5j1gibUdku.OvmaFrvKWjaAIQMy1RhkYYqXDdJtteG.	2025-04-08 11:03:08.618946	admin

Рисунок 14 – База данных после добавления пользователя

После проведения регистрации, мы видим, что данные введенные на сайте успешно были сохранены в БД.

5.2 Авторизация пользователя

При авторизации пользователь вводит ранее добавленные данные в БД. Если введенные данные и данные, которые хранятся в базе данных, совпадают, авторизация проходит успешно. В противном случае пользователю будет показано сообщение об ошибке.

На рисунке 15 представлен сценарий тестирования с ошибочными данными.

Рисунок 15 – Ввод неправильных данных

Результат входа с неправильными данными показан на рисунке 16.

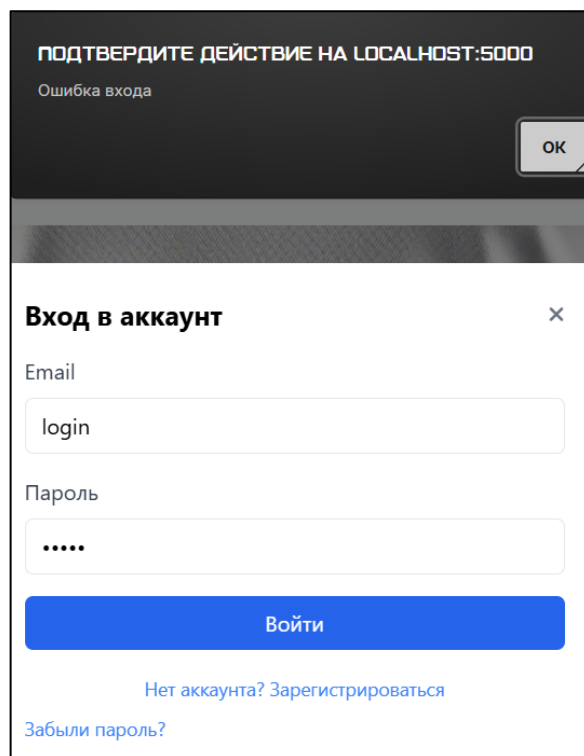


Рисунок 16 – Сообщение об ошибке

Далее введем достоверные данные. На рисунке 17 представлен сценарий тестирования с достоверными данными.

Рисунок 17 – Ввод верных данных

Результат входа с верными данными показан на рисунке 18

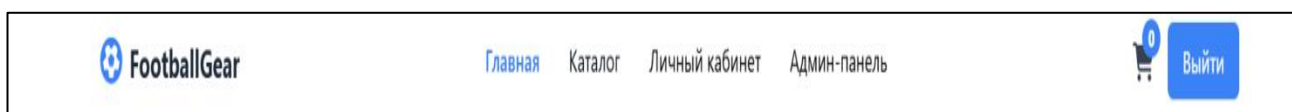


Рисунок 18 – Результат авторизации

В ходе тестирования все сценарии прошли успешно. При неправильных данных пользователю была выведена ошибка. С верно введенными данными произошел переход на главную страницу где отображено имя пользователя, который выполнил вход.

5.3 Тестирование функционала взаимодействия с товарами

Авторизованным пользователям доступен функционал просмотра, добавления в корзину и покупки товаров. Тестовый сценарий состоит из перехода на страницу каталог, где товар будет добавлен в корзину и куплен.

На рисунке 19 отображено добавление товара в корзину.



Рисунок 19 – Загрузка книги в базу данных

На рисунке 20 представлена корзина и уведомление о покупке.

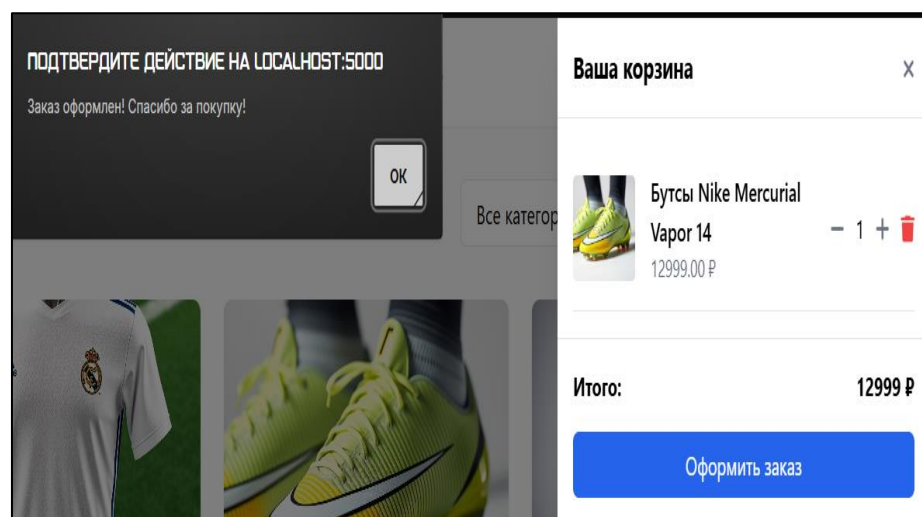


Рисунок 20 – Результат загрузки книги в базу данных

Также пользователь с ролью «Администратор» сможет добавлять новые товары, и в дальнейшем мы рассмотрим сценарий добавления новой позиции при помощи админ-панели на рисунке 21.

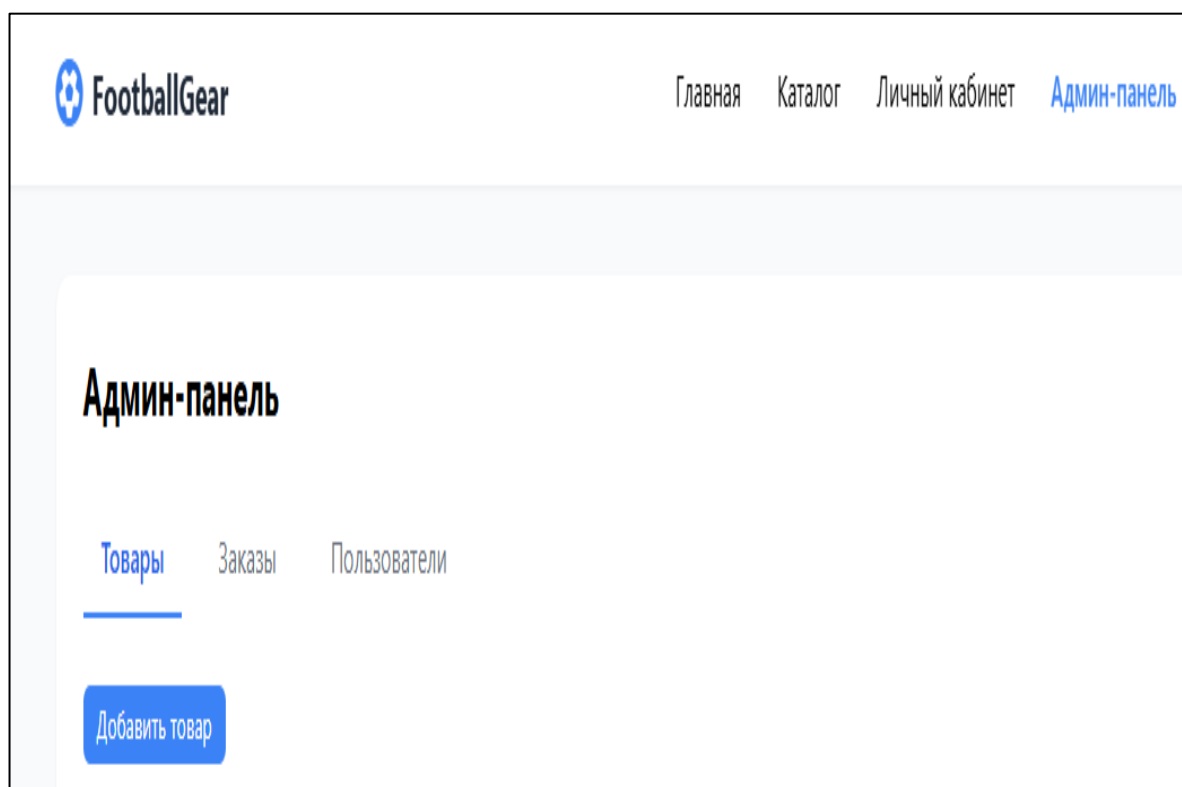


Рисунок 21 – Вид админ-панели и кнопки «Добавить товар»

После нажатие на кнопку добавления товара перед администратором всплывает окно добавления товара, что мы видим на рисунке 22

Рисунок 22 – Окно добавления товара

Добавленные товары, которые хранятся в базе данных отображены на рисунке 23.

Query Query History

```
1 select * from products;
```

Data Output Messages Notifications

	id [PK] integer	name character varying (255)	category character varying (100)	price numeric (10,2)	image character varying (255)	description text
1	1	Футболка Real Madrid Домашняя 2023	Футболки	4999.00	/images/1.png	Официальная
2	2	Бутсы Nike Mercurial Vapor 14	Бутсы	12999.00	/images/2.png	Бутсы для тве
3	3	Мяч Adidas Champions League	Мячи	5999.00	/images/3.png	Официальный
4	4	Футболка Barcelona Гостевая 2023	Футболки	4799.00	/images/4.png	Официальная
5	5	Бутсы Adidas Predator Edge	Бутсы	14999.00	/images/5.png	Бутсы для нат
6	6	Гетры Nike Strike	Аксессуары	1999.00	/images/6.png	Высококачест
7	7	Мяч Nike Premier League	Мячи	5499.00	/images/7.png	Официальный
8	8	Футболка Manchester United Домашняя 2023	Футболки	5199.00	/images/8.png	Официальная
9	9	Бутсы Puma Future Z 1.4	Бутсы	11999.00	/images/9.png	Бутсы с техно.
10	10	Щитки Nike Mercurial Lite	Аксессуары	2499.00	/images/10.png	Легкие щитки

Showing row

Рисунок 23 – Добавленные в базу данных товары

Добавление товара

Кроссовки

Обувь

10000

https://outmaxshop.ru/components/com_jshopping/files/img_products/31355/nike-shox-tl-31355.jpg

Тестовые кроссовки

☐ Популярный товар

Сохранить товар

Далее наш товар отображается на сайте, что мы видим на рисунке 25



1

select * from products;

Data Output

Messages

Notifications

+

📄

▼

📄

🗑️

🔍

📶

SQL

Showing rows: 1 to 11

Page No: 1 of 1

	id [PK] integer	name character varying (255)	category character varying (100)	price numeric (10,2)	image character varying (255)
1	12	Кроссовки	Обувь	10000.00	https://outmaxshop.ru/components/com_jshopping/files/img_products/31355/nike-shox-tl-31355-1.jpg
2	1	Футболка Real Madrid Домашняя 2023	Футболки	4999.00	/images/1.png
3	2	Бутсы Nike Mercurial Vapor 14	Бутсы	12999.00	/images/2.png
4	3	Мяч Adidas Champions League	Мячи	5999.00	/images/3.png
5	4	Футболка Barcelona Гостевая 2023	Футболки	4799.00	/images/4.png
6	5	Бутсы Adidas Predator Edge	Бутсы	14999.00	/images/5.png
7	6	Гетры Nike Strike	Аксессуары	1999.00	/images/6.png
8	7	Мяч Nike Premier League	Мячи	5499.00	/images/7.png

					ПГУ 1.090301.24.001 ПЗ	Лист
						55
Изм.	Лист	№ докум.	Подпись	Дата		

После проделанных действий, ожидаемый результат совпал с действительным, тестирование данных функций прошли успешно.

					ПГУ 1.090301.24.001 ПЗ	Лист
						56
Изм.	Лист	№ докум.	Подпись	Дата		

6 Безопасность жизнедеятельности

Безопасность жизнедеятельности при работе за компьютером является важным фактором для разработчиков и других профессионалов, которые проводят много времени за компьютером. Несоблюдение мер безопасности может привести к различным проблемам со здоровьем, включая напряжение глаз, нарушения осанки, повышенный риск развития карпального туннельного синдрома и других заболеваний.

В данном разделе будут рассмотрены основные аспекты безопасности жизнедеятельности и рекомендации при работе за компьютером для разработчиков.

6.1 Требования к помещению и рабочему месту

При выборе рабочего места, оборудованного компьютером, предпочтение следует отдавать помещениям, ориентированным на север или северо-восток. Площадь на одно рабочее место с ПК с жидкокристаллическими мониторами должна составлять не менее - 4,5 м. кв. Помещения, предусмотренные для работы с ПК, обязательно должны иметь искусственное и естественное освещение. Должна быть достаточная вентиляция, чтобы обеспечить свежий воздух и уменьшить концентрацию вредных веществ, таких как формальдегид, которые могут выделяться из компьютера. Содержание вредных химических веществ в воздухе таких помещений не должно превышать предельно допустимых концентраций вредных веществ в воздухе рабочей зоны в соответствии с действующими гигиеническими нормативами. Рекомендуемый уровень шума в помещении не должен превышать 50 дБ, чтобы избежать отвлекающих звуков и повышенного стресса [6].

Помещение должно быть оборудовано защитным заземлением. Поскольку одним из вредных факторов влияния на человека при использовании компьютера являются электромагнитные излучения, создаваемые монитором, клавиатурой,

соединительными кабелями. Превышение нормы такого излучения случается при использовании устаревших моделей ЭВМ, но наиболее частая причина – неправильно выполненное подключение, отсутствие или некачественное выполнение заземления. При использовании компьютера в таком помещении особенно в сочетании с высоким зрительным и нервно-эмоциональным напряжением, вызывает существенные изменения со стороны центральной нервной и сердечно-сосудистой системы (головные боли, раздражительность, нарушение сна, неадекватные психические реакции).

Рабочий стол должен размещаться таким образом, чтобы монитор был ориентирован боковой стороной к окнам, а естественный свет падал преимущественно слева. В целях защиты от прямых солнечных лучей на окнах должны быть солнцезащитные жалюзи или однотонные шторы из плотной ткани.

Также должны соблюдаться следующие требования:

- температура 22 °С (допустимая – 20-24 °С);
- оптимальная относительная влажность – 40-60% (допустимая – не более 75%);
- скорость движения воздуха не более 0.1м/с.

6.2 Требования к освещению

Соблюдение норм освещенности при работе за ПК это необходимое условие, так как недостаточное или неправильное освещение может привести к различным проблемам со здоровьем. Можно выделить следующую часто возникающую проблему это напряжение глаз, при ее наличии увеличивается усталость, что приводит к понижению производительности. С другой стороны, избыточное освещение также вызвать усталость глаз и головную боль.

Важно понимать, что освещение соответствует не только общей яркости, но также и равномерности распределения света в помещении. Если свет распределен неправильно, то это приводит к неравномерной яркости на поверхности рабочего стола, бликам на экране монитора и другим проблемам. Для предотвращения этих

					ПГУ 1.090301.24.001 ПЗ	Лист
						58
Изм.	Лист	№ докум.	Подпись	Дата		

проблем в помещении должно быть, как искусственное освещение, так и естественное.

Искусственное освещение в помещениях для эксплуатации ПК должно осуществляться системой общего равномерного освещения, такой системой могут служить люстры и потолочные светильники. При необходимости к общему освещению дополнительно можно устанавливать светильники местного освещения. В качестве источников света при искусственном освещении могут использоваться лампы накаливания или люминесцентные лампы типа желтого свечения. Лампы голубого свечения не используются. Не следует использовать в одном помещении люминесцентные лампы и лампы накаливания. Люминесцентные лампы должны быть защищены экранирующими решетками или рассеивателями для предотвращения слепящего действия ламп на глаза и возникновения бликов на поверхности экранов мониторов.

Рекомендации по требованиям к освещению при работе с ПК:

- общая освещенность была на уровне 300 лк;
- яркость на поверхности рабочего стола была равномерной и не превышала 500 лк;
- рабочее место должно находиться возле окна, чтобы использовать естественный свет;
- цветовая температура искусственного освещения должна быть в диапазоне от 4000 до 6500 К, так как это диапазон наиболее близкий к естественному свету.

6.3 Требования к режиму труда и отдыха

Требования к режиму труда и отдыха при работе с ПК условно могут быть разделены на три категории. Эти категории включают в себя физические, психологические и организационные требования.

Физические требования:

- правильное положение тела при работе с ПК;

					<i>ПГУ 1.090301.24.001 ПЗ</i>	Лист
						59
Изм.	Лист	№ докум.	Подпись	Дата		

- регулярные перерывы для растяжки и разминки;
- установка рабочего места на оптимальной высоте и расстоянии от глаз;
- правильное освещение рабочего места.

Психологические требования:

- регулярные перерывы для предотвращения усталости и переутомления;
- управление стрессом и снятие напряжения.

Организационные требования:

- обеспечение комфортной температуры и влажности в помещении;
- продуманный график рабочего времени;
- отдых в выходные и праздничные дни.

Рекомендуется делать перерывы каждые 60-90 минут работы и на 10-15 минут выйти из помещения или сделать небольшую физическую зарядку. Соблюдение этих требований снижает риск возникновения усталости и уменьшает уровень стресса. Правильный режим труда и отдыха может также повысить производительность и эффективность работы.

6.4 Рекомендации по использованию программного обеспечения

Программное обеспечение, которое помогает программисту следить за требованиями по обеспечению безопасного использования компьютера, предотвращает напряжение глаз, несоблюдение режима работы и другие проблемы, связанные с работой за ПК.

Рекомендуется использовать следующее программное обеспечение для улучшения безопасности жизнедеятельности:

- фильтры синего света. Программы, которые фильтруют синий свет, могут помочь уменьшить напряжение глаз, связанное с работой за компьютером. Некоторые из таких программ: Night Shift (для Mac), Night Light (для Windows 10);
- приложения для управления временем. Они помогают управлять временем, тем самым повышая производительность. Также эти приложения,

					<i>ПГУ 1.090301.24.001 ПЗ</i>	Лист
						60
Изм.	Лист	№ докум.	Подпись	Дата		

могут уведомлять о необходимости делать перерывы, что помогает уменьшить усталость глаз и предотвратить перенапряжение мышц. Некоторые из таких программ: Workrave или EyeLeo;

- программы для управления шрифтами. Позволяют управлять шрифтами, могут помочь уменьшить усталость глаз при чтении. Некоторые из таких программ: Cleartype Tuner (для Windows), HiDPI Renderer (для Mac);

- программы для управления звуком. Могут помочь уменьшить стресс и улучшить настроение. Некоторые из таких программ: Noisli, RainyMood;

- программы управления задачами. Такие как Trello или ToDoist, помогают планировать работу и управлять задачами, что может снизить уровень стресса и повысить производительность.

Таким образом, использование выше перечисленного программного обеспечения обеспечит комфортные и безопасные условия работы за компьютером.

Заключение

В ходе выполнения выпускной квалификационной работы были выполнены все поставленные задачи: проведён анализ предметной области, выбраны технологии для реализации клиент-серверного взаимодействия, спроектирована база данных, разработано клиент-серверное приложение «Магазин футбольных принадлежностей», реализованы основные функциональные модули и выполнено тестирование.

Приложение позволяет пользователям просматривать каталог товаров, фильтровать их по категории и цене, добавлять в корзину и оформлять заказы. Для авторизованных пользователей доступны личный кабинет и история заказов. Администратор имеет возможность управлять ассортиментом, добавляя, редактируя и удаляя товары.

Для реализации была использована современная технологическая база: Node.js и Express.js – для серверной части, PostgreSQL – для хранения данных, Tailwind CSS и Vanilla JavaScript – для клиентской части, Nginx – для размещения на сервере. Также были применены механизмы безопасности: JWT для аутентификации и bcrypt для шифрования паролей.

Тестирование показало, что приложение работает стабильно и отвечает всем заявленным требованиям. Основные сценарии – регистрация, авторизация, просмотр товаров, работа с корзиной и заказами – выполняются корректно.

В дальнейшем приложение может быть расширено за счёт дополнительных возможностей, таких как интеграция с платёжными системами, улучшение интерфейса, добавление мобильной версии или аналитики продаж для администратора.

Выполнение данной работы позволило получить практические навыки в разработке клиент-серверных приложений, работе с базами данных и внедрении REST API, что несомненно пригодится в профессиональной деятельности.

					<i>ПГУ 1.090301.24.001 ПЗ</i>	Лист
						62
Изм.	Лист	№ докум.	Подпись	Дата		

Список использованных источников

1. Кузин, А. В. Информационные системы и технологии: учебник / А. В. Кузин, С. А. Новожилов. – Москва: ЮНИТИ-ДАНА, 2021. – 479 с.
2. Олифер, В. Г. Компьютерные сети: принципы, технологии, протоколы / В. Г. Олифер, Н. А. Олифер. – Санкт-Петербург : Питер, 2020. – 896 с.
3. Лаптев, В. В. Разработка веб-приложений с использованием клиент-серверной архитектуры / В. В. Лаптев // Информационные технологии и системный анализ. – 2021. – № 2. – С. 45–53.
4. Гордеев, А. Н. Web-программирование и разработка информационных систем / А. Н. Гордеев. – Санкт-Петербург: БХВ-Петербург, 2022. – 608 с.
5. Флёров, А. А. Разработка интернет-магазинов на основе современных технологий / А. А. Флёров // Научный вестник НГТУ. – 2019. – № 4(77). – С. 112–121.
6. Денисов, В. П. Проектирование баз данных в среде PostgreSQL / В. П. Денисов // Программирование и компьютерные технологии. – 2020. – № 3. – С. 78–85.
7. Санитарно-гигиенические рекомендации для организации рабочего места пользователя персональным компьютером. [Электронный ресурс] / Режим доступа: <https://sakhgig.ru/main/info/664-sanitarno-gigienicheskie-rekomendacii-dlya-organizacii-rabochego-mesta-polzovatelya-personalnym-kompyuterom.html>, свободный (дата обращения 26.05.2025).

Приложение А

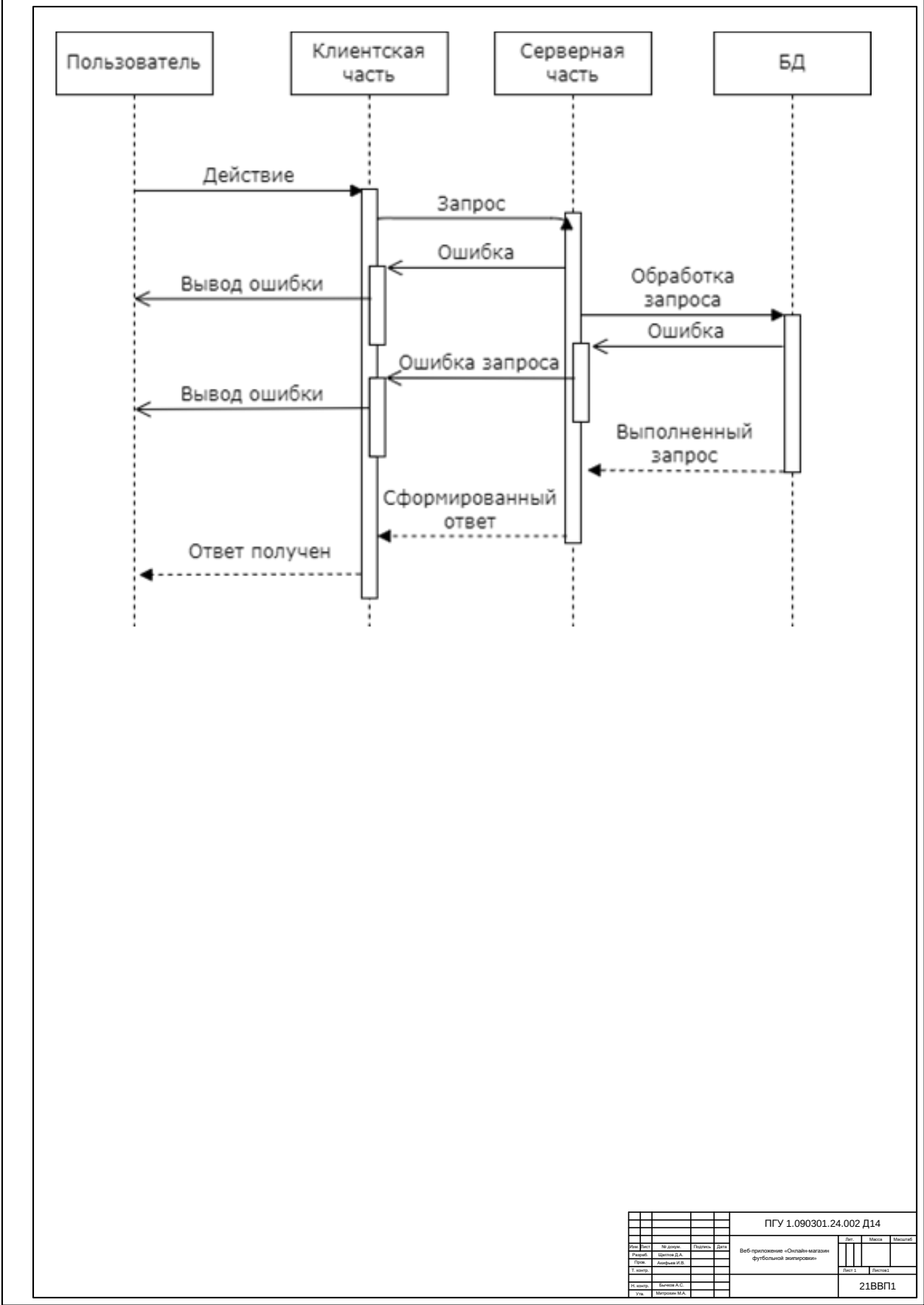
(обязательное)

Веб-приложение «Онлайн магазин футбольных принадлежностей»

UML-диаграммы

					ПГУ 1.090301.24.001 ПЗ	Лист
						64
Изм.	Лист	№ докум.	Подпись	Дата		

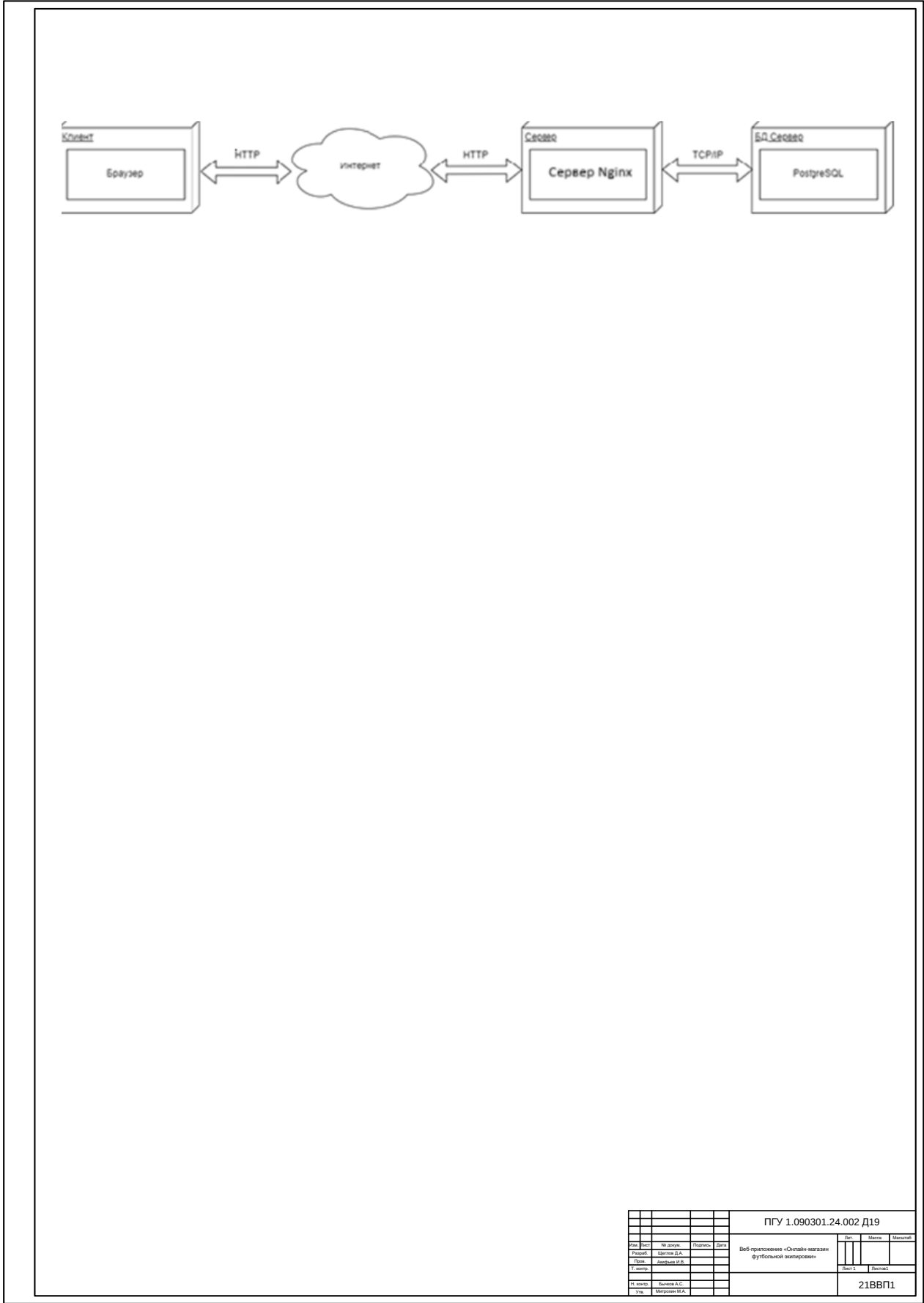
ПриложениеА.1-Диаграмма последовательностей



					ПГУ 1.090301.24.002 Д14				
Имя	Фамилия	Имя	Фамилия	Дата	Имя	Фамилия	Имя	Фамилия	Дата
Имя	Фамилия	Имя	Фамилия	Дата	Веб-приложение «Онлайн-магазин футбольной экипировки»				
Имя	Фамилия	Имя	Фамилия	Дата					
Имя	Фамилия	Имя	Фамилия	Дата					
Имя	Фамилия	Имя	Фамилия	Дата					
Имя	Фамилия	Имя	Фамилия	Дата					
Имя	Фамилия	Имя	Фамилия	Дата					
Имя	Фамилия	Имя	Фамилия	Дата					
Имя	Фамилия	Имя	Фамилия	Дата					
					21ВВП1				

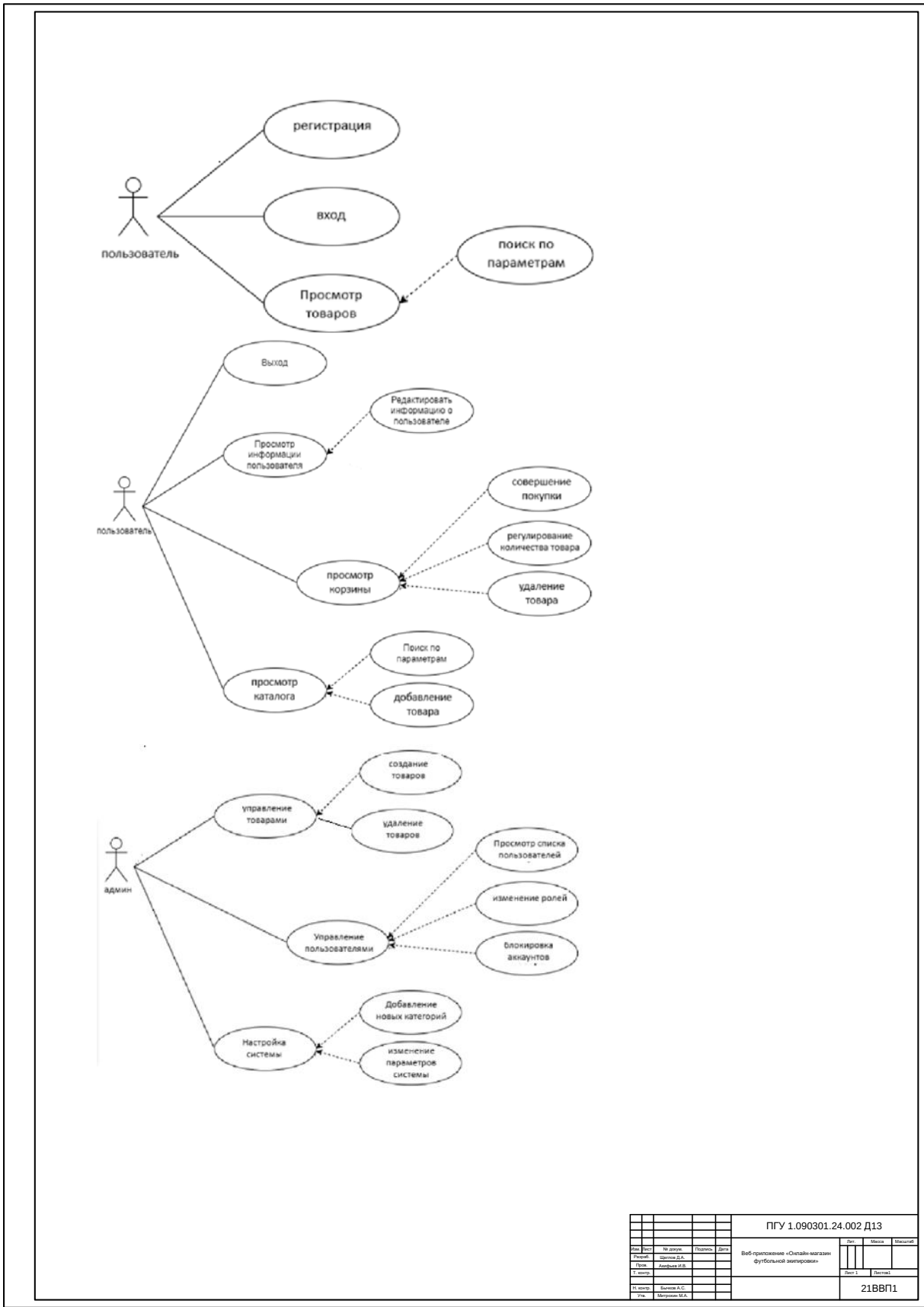
Изм.	Лист	№ докум.	Подпись	Дата

ПриложениеА.2-Диаграмма развертывания

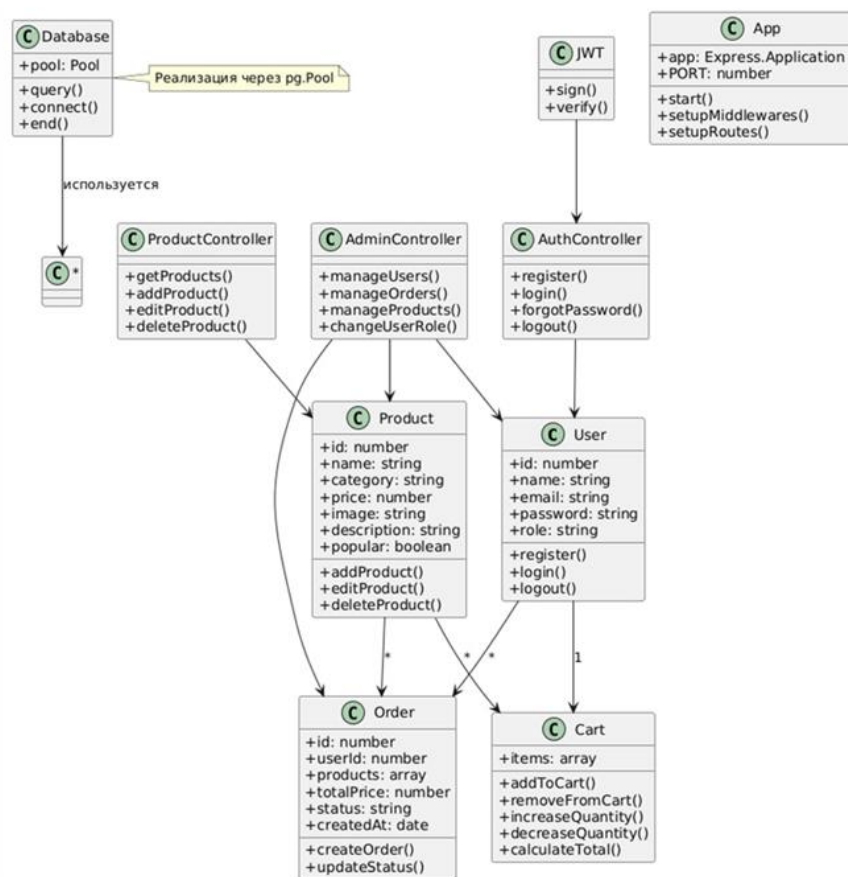


					ПГУ 1.090301.24.002 Д19				
Имя	Фамилия	№ докум.	Подпись	Дата	Вед-приложение «Онлайн-магазин футбольной экипировки»				
Подпись									
Т. номер									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									
Имя									
И. номер									

Приложение А.3-Диаграмма вариантов использования



Приложение А.4-Диаграмма классов



						ПУГ 1.090301.24.002 Д11							
						Вед-приложение «Онлайн-магазин футбольной экипировки»							
Имя (Фамилия)	№ документа	Подпись	Дата								Лист	Масштаб	Масштаб
Рязанов	Степанов Д.А.												
Прото	Александр И.В.												
Т. номер													
И. номер	Выпущен А.С.					Лист 1			Листов				
						21ВВП1							

Приложение Б

(обязательное)

Клиент серверное приложение «Онлайн-магазин футбольной экипировки»

Листинг программы

					ПГУ 1.090301.24.001 ПЗ	Лист
						69
Изм.	Лист	№ докум.	Подпись	Дата		

Файл index.html

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>FootballGear - Магазин футбольной экипировки</title>
  <script src="https://cdn.tailwindcss.com"></script>
  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/tailwindcss@2.2.19/dist/tailwind.min.css">
  <style>
    .active-tab {
      border-bottom: 3px solid #3b82f6;
      color: #3b82f6;
      font-weight: 600;
    }
    .product-card:hover {
      transform: translateY(-5px);
      box-shadow: 0 10px 25px -5px rgba(0, 0, 0, 0.1);
    }
    .cart-item-enter {
      opacity: 0;
      transform: translateX(-20px);
    }
    .cart-item-enter-active {
      opacity: 1;
      transform: translateX(0);
      transition: all 300ms ease-in;
    }
    .cart-item-exit {
      opacity: 1;
      transform: translateX(0);
    }
    .cart-item-exit-active {
      opacity: 0;
      transform: translateX(20px);
      transition: all 300ms ease-out;
    }
  </style>
</head>
<body class="bg-gray-50 font-sans">
  <!-- Шапка сайта -->
  <header class="bg-white shadow-sm sticky top-0 z-10">
    <div class="container mx-auto px-4 py-3 flex justify-between items-center">
      <div class="flex items-center space-x-2">
        <i class="fas fa-futbol text-blue-500 text-2xl"></i>
        <h1 class="text-xl font-bold text-gray-800">FootballGear</h1>
      </div>

      <nav class="hidden md:flex space-x-6">
        <a href="#" class="tab-link py-2 px-1 border-b-2 border-transparent hover:text-blue-500 transition" data-tab="home">Главная</a>
      </nav>
    </div>
  </header>
</body>
</html>
```

					ПГУ 1.090301.24.001 ПЗ	Лист
						70
Изм.	Лист	№ докум.	Подпись	Дата		

```

        <a href="#" class="tab-link py-2 px-1 border-b-2 border-
transparent hover:text-blue-500 transition" data-tab="catalog">Каталог</a>
        <a href="#" class="tab-link py-2 px-1 border-b-2 border-
transparent hover:text-blue-500 transition" data-tab="account">Личный кабинет</a>
        <a href="#" class="tab-link py-2 px-1 border-b-2 border-
transparent hover:text-blue-500 transition" data-tab="admin">Админ-панель</a>
    </nav>

    <div class="flex items-center space-x-4">
        <button class="relative" id="cart-button">
            <i class="fas fa-shopping-cart text-gray-700 hover:text-blue-
500 text-xl"></i>
            <span class="absolute -top-2 -right-2 bg-blue-500 text-white
text-xs font-bold rounded-full h-5 w-5 flex items-center justify-center" id="cart-
count">0</span>
        </button>

        <button id="auth-button" class="bg-blue-500 hover:bg-blue-600
text-white px-4 py-2 rounded-md transition">
            Войти
        </button>
        <button class="md:hidden" id="mobile-menu-button">
            <i class="fas fa-bars text-gray-700 text-xl"></i>
        </button>
    </div>
</div>
<div class="md:hidden hidden bg-white shadow-lg absolute w-full z-20"
id="mobile-menu">
    <div class="container mx-auto px-4 py-3 flex flex-col space-y-3">
        <a href="#" class="tab-link block py-2 px-1 hover:text-blue-500
transition" data-tab="home">Главная</a>
        <a href="#" class="tab-link block py-2 px-1 hover:text-blue-500
transition" data-tab="catalog">Каталог</a>
        <a href="#" class="tab-link block py-2 px-1 hover:text-blue-500
transition" data-tab="account">Личный кабинет</a>
        <a href="#" class="tab-link block py-2 px-1 hover:text-blue-500
transition" data-tab="admin">Админ-панель</a>
    </div>
</div>
</header>
<div class="fixed inset-0 bg-black bg-opacity-50 z-30 hidden" id="cart-modal">
    <div class="absolute right-0 top-0 h-full bg-white w-full md:w-96 shadow-
xl overflow-y-auto">
        <div class="p-4 flex justify-between items-center border-b">
            <h3 class="text-lg font-bold">Ваша корзина</h3>
            <button id="close-cart">
                <i class="fas fa-times text-gray-500 hover:text-gray-700"></i>
            </button>
        </div>
        <div class="p-4" id="cart-items">
            <div class="text-center py-8 text-gray-500" id="empty-cart-
message">

```

```

        <i class="fas fa-shopping-cart text-4xl mb-2"></i>
        <p>Ваша корзина пуста</p>
    </div>
</div>

<div class="p-4 border-t">
    <div class="flex justify-between mb-4">
        <span class="font-bold">Итого:</span>
        <span class="font-bold" id="cart-total">0 Р</span>
    </div>
    <button class="w-full bg-blue-500 hover:bg-blue-600 text-white py-
2 rounded-md transition" id="checkout-button">
        Оформить заказ
    </button>
</div>
</div>
</div>
<div class="fixed inset-0 bg-black bg-opacity-50 z-30 hidden" id="auth-modal">
    <div class="absolute top-1/2 left-1/2 transform -translate-x-1/2 -
translate-y-1/2 bg-white rounded-lg shadow-xl w-full max-w-md">
        <div class="p-6">
            <div class="flex justify-between items-center mb-4">
                <h3 class="text-xl font-bold" id="auth-modal-title">Вход в
аккаунт</h3>
                <button id="close-auth">
                    <i class="fas fa-times text-gray-500 hover:text-gray-
700"></i>
                </button>
            </div>
            <div id="login-form">
                <div class="mb-4">
                    <label class="block text-gray-700 mb-2" for="login-
email">Email</label>
                    <input type="email" id="login-email" class="w-full px-3
py-2 border rounded-md focus:outline-none focus:ring-2 focus:ring-blue-500">
                </div>
                <div class="mb-4">
                    <label class="block text-gray-700 mb-2" for="login-
password">Пароль</label>
                    <input type="password" id="login-password" class="w-full
px-3 py-2 border rounded-md focus:outline-none focus:ring-2 focus:ring-blue-500">
                </div>
                <div class="mb-4">
                    <button class="w-full bg-blue-500 hover:bg-blue-600 text-
white py-2 rounded-md transition" id="login-button">
                        Войти
                    </button>
                </div>
            </div>
            <div class="text-center">
                <button class="text-blue-500 hover:text-blue-700 text-sm"
id="show-register">
                    Нет аккаунта? Зарегистрироваться
                </button>
            </div>
        </div>
    </div>
</div>

```

					ПГУ 1.090301.24.001 ПЗ	Лист
						72
Изм.	Лист	№ докум.	Подпись	Дата		

```

        </button>
        <button class="block text-blue-500 hover:text-blue-700
text-sm mt-2" id="show-forgot">
            Забыли пароль?
        </button>
    </div>
</div>

<div id="register-form" class="hidden">
    <div class="mb-4">
        <label class="block text-gray-700 mb-2" for="register-
name">Имя</label>
        <input type="text" id="register-name" class="w-full px-3
py-2 border rounded-md focus:outline-none focus:ring-2 focus:ring-blue-500">
    </div>
    <div class="mb-4">
        <label class="block text-gray-700 mb-2" for="register-
email">Email</label>
        <input type="email" id="register-email" class="w-full px-3
py-2 border rounded-md focus:outline-none focus:ring-2 focus:ring-blue-500">
    </div>
    <div class="mb-4">
        <label class="block text-gray-700 mb-2" for="register-
password">Пароль</label>
        <input type="password" id="register-password" class="w-
full px-3 py-2 border rounded-md focus:outline-none focus:ring-2 focus:ring-blue-
500">
    </div>
    <div class="mb-4">
        <button class="w-full bg-blue-500 hover:bg-blue-600 text-
white py-2 rounded-md transition" id="register-button">
            Зарегистрироваться
        </button>
    </div>
    <div class="text-center">
        <button class="text-blue-500 hover:text-blue-700 text-sm"
id="show-login">
            Уже есть аккаунт? Войти
        </button>
    </div>
</div>

<div id="forgot-form" class="hidden">
    <div class="mb-4">
        <p class="text-gray-700 mb-4">Введите ваш email, и мы
вышлем инструкции по восстановлению пароля</p>
        <label class="block text-gray-700 mb-2" for="forgot-
email">Email</label>
        <input type="email" id="forgot-email" class="w-full px-3
py-2 border rounded-md focus:outline-none focus:ring-2 focus:ring-blue-500">
    </div>
    <div class="mb-4">

```

```

        <button class="w-full bg-blue-500 hover:bg-blue-600 text-
white py-2 rounded-md transition" id="forgot-button">
            Восстановить пароль
        </button>
    </div>
    <div class="text-center">
        <button class="text-blue-500 hover:text-blue-700 text-sm"
id="forgot-back">
            Назад к входу
        </button>
    </div>
</div>
</div>
</div>
</div>
<main class="container mx-auto px-4 py-8">
    <!-- Главная страница -->
    <div id="home-tab" class="tab-content">
        <h3 class="text-2xl font-bold mb-6">Популярные товары</h3>
        <div class="grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-4 gap-6"
id="popularProductsContainer">
            </div>
        </div>

        <!-- Каталог товаров -->
        <div id="catalog-tab" class="tab-content hidden">
            <div class="flex justify-between items-center mb-8">
                <h2 class="text-2xl font-bold">Каталог товаров</h2>
                <div class="flex space-x-2">
                    <select class="border rounded-md px-3 py-2 focus:outline-none
focus:ring-2 focus:ring-blue-500">
                        <option>Все категории</option>
                        <option>Футболки</option>
                        <option>Бутсы</option>
                        <option>Мячи</option>
                        <option>Аксессуары</option>
                    </select>
                    <select class="border rounded-md px-3 py-2 focus:outline-none
focus:ring-2 focus:ring-blue-500">
                        <option>Сортировка</option>
                        <option>По популярности</option>
                        <option>По возрастанию цены</option>
                        <option>По убыванию цены</option>
                        <option>По новизне</option>
                    </select>
                </div>
            </div>
            <div class="grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-3 xl:grid-
cols-4 gap-6" id="catalog-items">
                <!-- Товары каталога будут добавляться сюда динамически -
-->
            </div>
        </div>
    </div>

```

```

</div>

<!-- Личный кабинет -->
<div id="account-tab" class="tab-content hidden">
  <div class="bg-white rounded-xl shadow-sm p-6">
    <h2 class="text-2xl font-bold mb-6">Личный кабинет</h2>

    <div class="hidden" id="account-not-auth">
      <div class="text-center py-8">
        <i class="fas fa-user-lock text-4xl text-gray-400 mb-4"></i>

        <p class="text-gray-600 mb-4">Для доступа к личному кабинету необходимо авторизоваться</p>
        <button class="bg-blue-500 hover:bg-blue-600 text-white px-6 py-2 rounded-md transition" id="account-login-button">Войти</button>
      </div>
    </div>

    <div class="hidden" id="account-content">
      <div class="flex flex-col md:flex-row gap-8">
        <div class="md:w-1/3">
          <div class="flex items-center space-x-4 mb-6">
            <div class="bg-blue-100 text-blue-600 rounded-full h-12 w-12 flex items-center justify-center">
              <i class="fas fa-user text-xl"></i>
            </div>
            <div>
              <h3 class="font-bold" id="account-name">Иван Иванов</h3>
              <p class="text-sm text-gray-500" id="account-email">ivan@example.com</p>
            </div>
          </div>
          <nav class="space-y-2">
            <button class="account-nav block w-full text-left px-4 py-2 rounded-md bg-blue-50 text-blue-600 font-medium">
              <i class="fas fa-user mr-2"></i> Профиль
            </button>
            <button class="account-nav block w-full text-left px-4 py-2 rounded-md hover:bg-gray-100 transition">
              <i class="fas fa-history mr-2"></i> История заказов
            </button>
            <button class="account-nav block w-full text-left px-4 py-2 rounded-md hover:bg-gray-100 transition">
              <i class="fas fa-heart mr-2"></i> Избранное
            </button>
            <button class="account-nav block w-full text-left px-4 py-2 rounded-md hover:bg-gray-100 transition">
              <i class="fas fa-lock mr-2"></i> Безопасность

```

					ПГУ 1.090301.24.001 ПЗ	Лист
						75
Изм.	Лист	№ докум.	Подпись	Дата		

```

        </button>
        <button class="account-nav block w-full text-left
px-4 py-2 rounded-md hover:bg-gray-100 transition" id="logout-button">
            <i class="fas fa-sign-out-alt mr-2"></i> ВЫЙТИ
        </button>
    </nav>
</div>

<div class="md:w-2/3" id="account-details">
    <h3 class="text-xl font-bold mb-4">Мой профиль</h3>

    <div class="grid grid-cols-1 md:grid-cols-2 gap-4 mb-
6">

        <div>
            <label class="block text-gray-700 mb-
1">Имя</label>

            <input type="text" class="w-full px-3
py-2 border rounded-md" value="Иван" id="account-first-name">
        </div>
        <div>
            <label class="block text-gray-700 mb-
1">Фамилия</label>

            <input type="text" class="w-full px-3 py-2
border rounded-md" value="Иванов" id="account-last-name">
        </div>
        <div>
            <label class="block text-gray-700 mb-
1">Email</label>

            <input type="email" class="w-full px-3 py-2
border rounded-md" value="ivan@example.com" disabled>
        </div>
        <div>
            <label class="block text-gray-700 mb-
1">Телефон</label>

            <input type="tel" class="w-full px-3 py-2
border rounded-md" value="+7 (900) 123-45-67" id="account-phone">
        </div>
    </div>

    <button class="bg-blue-500 hover:bg-blue-600 text-
white px-6 py-2 rounded-md transition">
        Сохранить изменения
    </button>
</div>
</div>
</div>
</div>
</div>

<div id="admin-tab" class="tab-content hidden">
    <div class="bg-white rounded-xl shadow-sm p-6">
        <h2 class="text-2xl font-bold mb-6">Админ-панель</h2>

```

```

<!-- Сообщение о недоступности -->
<div class="hidden" id="admin-not-auth">
  <div class="text-center py-8">
    <i class="fas fa-lock text-4xl text-gray-400 mb-4"></i>
    <p class="text-gray-600 mb-4">Доступ только для
администраторов</p>
    <button class="bg-blue-500 hover:bg-blue-600 text-white
px-6 py-2 rounded-md transition" id="admin-login-button">
      Войти как администратор
    </button>
  </div>
</div>
<div class="hidden" id="admin-content">
  <div class="flex space-x-4 mb-6">
    <button class="admin-nav px-4 py-2 border-b-2 border-blue-
500 font-medium text-blue-600" data-admin-tab="products">
      Товары
    </button>
    <button class="admin-nav px-4 py-2 border-b-2 border-
transparent text-gray-500 hover:text-blue-500 transition" data-admin-tab="orders">
      Заказы
    </button>
    <button class="admin-nav px-4 py-2 border-b-2 border-
transparent text-gray-500 hover:text-blue-500 transition" data-admin-tab="users">
      Пользователи
    </button>
  </div>
  <div id="admin-products">
    <div class="mb-4">
      <button id="add-product-button" class="bg-blue-500
hover:bg-blue-600 text-white px-3 py-1 rounded-md text-sm transition">
        Добавить товар
      </button>
    </div>
    <div id="add-product-form" class="hidden mb-6 bg-gray-50 p-
4 rounded-md border">
      <h3 class="text-lg font-bold mb-4">Добавление
товара</h3>
      <div class="grid grid-cols-1 md:grid-cols-2 gap-4">
        <input type="text" id="product-name"
placeholder="Название товара" class="border px-3 py-2 rounded-md">
        <input type="text" id="product-category" placehold-
er="Категория" class="border px-3 py-2 rounded-md">
        <input type="number" id="product-price" placehold-
er="Цена" class="border px-3 py-2 rounded-md">
        <input type="text" id="product-image"
placeholder="Ссылка на изображение" class="border px-3 py-2 rounded-md">
      </div>
      <textarea id="product-description"
placeholder="Описание" class="border px-3 py-2 mt-4 w-full rounded-md"></textarea>
      <div class="flex items-center mt-4 space-x-2">

```

```

        <input type="checkbox" id="product-popular">
        <label for="product-popular">Популярный
товар</label>
    </div>
    <button id="save-product" class="mt-4 bg-blue-500
hover:bg-blue-600 text-white px-4 py-2 rounded-md">
        Сохранить товар
    </button>
</div>
<table class="min-w-full bg-white">
    <thead>
        <tr>
            <th class="py-2 px-4 border-b text-
left">ID</th>
            <th class="py-2 px-4 border-b text-
left">Изображение</th>
            <th class="py-2 px-4 border-b text-
left">Название</th>
            <th class="py-2 px-4 border-b text-
left">Категория</th>
            <th class="py-2 px-4 border-b text-
left">Цена</th>
            <th class="py-2 px-4 border-b text-
left">Действия</th>
        </tr>
    </thead>
    <tbody id="admin-products-list">
</tbody>
</table>
</div>
<div id="admin-users" class="hidden">
    <table class="min-w-full bg-white">
        <thead>
            <tr>
                <th class="py-2 px-4 border-b text-
left">ID</th>
                <th class="py-2 px-4 border-b text-
left">Имя</th>
                <th class="py-2 px-4 border-b text-
left">Email</th>
                <th class="py-2 px-4 border-b text-
left">Роль</th>
            </tr>
        </thead>
        <tbody id="admin-users-list">
        </tbody>
    </table>
</div>
</div>
</div>
</div>
</main>
<footer class="bg-gray-800 text-white py-8">

```

```

<div class="container mx-auto px-4">
  <div class="grid grid-cols-1 md:grid-cols-4 gap-8">
    <div>
      <h3 class="text-lg font-bold mb-4">FootballGear</h3>
      <p class="text-gray-400">Официальный магазин футбольной
экипировки и аксессуаров</p>
    </div>
    <div>
      <h3 class="text-lg font-bold mb-4">Каталог</h3>
      <ul class="space-y-2">
        <li><a href="#" class="text-gray-400 hover:text-white
transition" data-tab="catalog">Футболки</a></li>
        <li><a href="#" class="text-gray-400 hover:text-white
transition" data-tab="catalog">Бутсы</a></li>
        <li><a href="#" class="text-gray-400 hover:text-white
transition" data-tab="catalog">Мячи</a></li>
        <li><a href="#" class="text-gray-400 hover:text-white
transition" data-tab="catalog">Аксессуары</a></li>
      </ul>
    </div>
    <div>
      <h3 class="text-lg font-bold mb-4">Информация</h3>
      <ul class="space-y-2">
        <li><a href="#" class="text-gray-400 hover:text-white
transition">О нас</a></li>
        <li><a href="#" class="text-gray-400 hover:text-white
transition">Доставка и оплата</a></li>
        <li><a href="#" class="text-gray-400 hover:text-white
transition">Возврат</a></li>
        <li><a href="#" class="text-gray-400 hover:text-white
transition">Контакты</a></li>
      </ul>
    </div>
    <div>
      <h3 class="text-lg font-bold mb-4">Контакты</h3>
      <ul class="space-y-2">
        <li class="flex items-center space-x-2">
          <i class="fas fa-phone text-gray-400"></i>
          <span class="text-gray-400">+7 (800) 123-45-67</span>
        </li>
        <li class="flex items-center space-x-2">
          <i class="fas fa-envelope text-gray-400"></i>
          <span class="text-gray-
400">info@footballgear.ru</span>
        </li>
        <li class="flex items-center space-x-2">
          <i class="fas fa-map-marker-alt text-gray-400"></i>
          <span class="text-gray-400">Пенза, ул.
Спорта, 32</span>
        </li>
      </ul>
    </div>
  </div>

```

```

        </div>
        <div class="border-t border-gray-700 mt-8 pt-6 flex flex-col md:flex-
row justify-between items-center">
            <p class="text-gray-400 mb-4 md:mb-0">© 2025 FootballGear. Все
права защищены.</p>
            <div class="flex space-x-4">
                <a href="#" class="text-gray-400 hover:text-white transi-
tion"><i class="fab fa-vk"></i></a>
                <a href="#" class="text-gray-400 hover:text-white transi-
tion"><i class="fab fa-telegram"></i></a>
                <a href="#" class="text-gray-400 hover:text-white transi-
tion"><i class="fab fa-instagram"></i></a>
            </div>
        </div>
    </div>
</div>
</footer>

<script>
    async function fetchProducts() {
    try {
        const token = localStorage.getItem('token');
        if (!token) {
            throw new Error('Требуется авторизация');
        }

        const response = await fetch('https://footballgeer.ru/api/products', {
            headers: { 'Authorization': `Bearer ${token}` }
        });

        if (!response.ok) {
            throw new Error(`Ошибка загрузки товаров (${response.status})`);
        }

        return await response.json();
    } catch (error) {
        console.error('Ошибка:', error);
        alert('Ошибка при загрузке товаров');
        return [];
    }
}

    async function renderProducts() {

        const products = await fetchProducts();

        const popularProducts = products.filter(product => product.popular);
        let popularHTML = '';
        popularProducts.forEach(product => {
            popularHTML += `
                <div class="product-card bg-white rounded-lg shadow-sm
overflow-hidden transition duration-300">
                    <div class="relative pb-3/4 h-48">

```

					ПГУ 1.090301.24.001 ПЗ	Лист
						80
Изм.	Лист	№ докум.	Подпись	Дата		

```

        
    </div>
    <div class="p-4">
        <h3 class="font-bold mb-1">{product.name}</h3>
        <p class="text-gray-600 text-sm mb-2">{product.category}</p>
        <div class="flex justify-between items-center">
            <span class="font-bold text-lg">{product.price} Р</span>
            <button class="add-to-cart bg-blue-500 hover:bg-blue-600
text-white px-3 py-1 rounded-md text-sm transition" data-id="{product.id}">
                В корзину
            </button>
        </div>
    </div>
</div>
`;
});
elements.popularProductsContainer.innerHTML = popularHTML;
let catalogHTML = '';
products.forEach(product => {
    catalogHTML += `
        <div class="product-card bg-white rounded-lg shadow-sm overflow-hidden
transition duration-300">
            <div class="relative pb-3/4 h-48">
                
            </div>
            <div class="p-4">
                <h3 class="font-bold mb-1">{product.name}</h3>
                <p class="text-gray-600 text-sm mb-2">{product.category}</p>
                <p class="text-gray-500 text-xs mb-3 line-clamp-
2">{product.description}</p>
                <div class="flex justify-between items-center">
                    <span class="font-bold text-lg">{product.price} Р</span>
                    <button class="add-to-cart bg-blue-500 hover:bg-blue-600
text-white px-3 py-1 rounded-md text-sm transition" data-id="{product.id}">
                        В корзину
                    </button>
                </div>
            </div>
        </div>
    `;
});
elements.catalogItemsContainer.innerHTML = catalogHTML;
document.querySelectorAll('.add-to-cart').forEach(button => {
    button.addEventListener('click', (e) => {
        const id = parseInt(e.currentTarget.getAttribute('data-id'));
        addToCart(id);
    });
});
}

const state = {

```

					ПГУ 1.090301.24.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		81

```

        cart: [],
        currentTab: 'home',
        isAuthenticated: false,
        isAdmin: false,
        user: null
    };

    const elements = {
        tabLinks: document.querySelectorAll('.tab-link'),
        tabContents: document.querySelectorAll('.tab-content'),
        cartButton: document.getElementById('cart-button'),
        cartModal: document.getElementById('cart-modal'),
        closeCart: document.getElementById('close-cart'),
        cartItems: document.getElementById('cart-items'),
        cartCount: document.getElementById('cart-count'),
        cartTotal: document.getElementById('cart-total'),
        emptyCartMessage: document.getElementById('empty-cart-message'),
        checkoutButton: document.getElementById('checkout-button'),
        authButton: document.getElementById('auth-button'),
        authModal: document.getElementById('auth-modal'),
        closeAuth: document.getElementById('close-auth'),
        loginForm: document.getElementById('login-form'),
        registerForm: document.getElementById('register-form'),
        forgotForm: document.getElementById('forgot-form'),
        showRegister: document.getElementById('show-register'),
        showLogin: document.getElementById('show-login'),
        showForgot: document.getElementById('show-forgot'),
        forgotBack: document.getElementById('forgot-back'),
        loginButton: document.getElementById('login-button'),
        registerButton: document.getElementById('register-button'),
        forgotButton: document.getElementById('forgot-button'),
        logoutButton: document.getElementById('logout-button'),
        mobileMenuButton: document.getElementById('mobile-menu-button'),
        mobileMenu: document.getElementById('mobile-menu'),
        popularProductsContainer: document.querySelector('#popularProductsContainer'), // Используйте прямой ID
        catalogItemsContainer: document.getElementById('catalog-items'),
        accountNotAuth: document.getElementById('account-not-auth'),
        accountContent: document.getElementById('account-content'),
        accountLoginButton: document.getElementById('account-login-button'),
        adminNotAuth: document.getElementById('admin-not-auth'),
        adminContent: document.getElementById('admin-content'),
        adminLoginButton: document.getElementById('admin-login-button'),
        adminNavs: document.querySelectorAll('.admin-nav'),

        adminProducts: document.getElementById('admin-products'),
        adminOrders: document.getElementById('admin-orders'),
        adminUsers: document.getElementById('admin-users'),
        adminProductsList: document.getElementById('admin-products-list'),
        addProductButton: document.getElementById('add-product-button')
    };
};

```

```

    function init() {
    loadState();
    render();
    renderProducts();
    setupEventListeners();
}

    function loadState() {
        const savedState = localStorage.getItem('footballGearState');
        if (savedState) {
            Object.assign(state, JSON.parse(savedState));
            console.log('Состояние загружено:', state); // Добавьте лог
        }

    }

    function saveState() {
    localStorage.setItem('footballGearState', JSON.stringify(state));
    console.log('Состояние сохранено:', state); // Добавьте лог
}

    function render() {
        renderTabs();
        renderCart();
        renderProducts();
        renderAuthState();
        renderAccount();
        renderAdminPanel();
    }

    function renderTabs() {
        elements.tabLinks.forEach(link => {
            link.classList.remove('active-tab');
        });
        elements.tabContents.forEach(content => {
            content.classList.add('hidden');
        });
        document.querySelectorAll(`.tab-link[data-tab="${state.currentTab}"]`).forEach(link => {
            link.classList.add('active-tab');
        });
        document.getElementById(`${state.currentTab}-tab`).classList.remove('hidden');
    }

    function renderCart() {
        elements.cartCount.textContent = state.cart.reduce((total, item) => total +
item.quantity, 0);
        if (state.cart.length === 0) {

            elements.emptyCartMessage.classList.remove('hidden');
            elements.cartItems.innerHTML = '';
        } else {
            elements.emptyCartMessage.classList.add('hidden');
        }
        fetch('https://footballgeer.ru/api/products')
            .then(response => {
                if (!response.ok) {
                    throw new Error(`Ошибка загрузки товаров (${response.status})`);
                }
            })
    }

```

```

    }
    return response.json();
  })
  .then(products => {
    let cartHTML = '';
    let total = 0;

    state.cart.forEach(item => {
      const product = products.find(p => p.id === item.id);
      if (product) {
        total += product.price * item.quantity;
        cartHTML += `
          <div class="flex items-center justify-between py-4
border-b">

            <div class="flex items-center space-x-4">
              
              <div>
                <h4 class="font-
medium">${product.name}</h4>
                <p class="text-gray-500 text-
sm">${product.price} ₺</p>
              </div>
            </div>
            <div class="flex items-center space-x-3">
              <button class="decrease-quantity text-gray-500
hover:text-blue-500" data-id="${product.id}">
                <i class="fas fa-minus"></i>
              </button>
              <span class="quantity">${item.quantity}</span>
              <button class="increase-quantity text-gray-500
hover:text-blue-500" data-id="${product.id}">
                <i class="fas fa-plus"></i>
              </button>
              <button class="remove-from-cart text-red-500
hover:text-red-700" data-id="${product.id}">
                <i class="fas fa-trash"></i>
              </button>
            </div>
          </div>
        `;
      }
    });

    elements.cartItems.innerHTML = cartHTML;

    elements.cartTotal.textContent = ` ${total} ₺ `;

    // Навешиваем обработчики на кнопки корзины
    document.querySelectorAll('.decrease-quantity').forEach(button =>
    {
      button.addEventListener('click', (e) => {

```

```

        const id = parseInt(e.currentTarget.getAttribute('data-
id'));
        decreaseQuantity(id);
    });
});
document.querySelectorAll('.increase-quantity').forEach(button =>
{
    button.addEventListener('click', (e) => {
        const id = parseInt(e.currentTarget.getAttribute('data-
id'));
        increaseQuantity(id);
    });
});
document.querySelectorAll('.remove-from-cart').forEach(button => {
    button.addEventListener('click', (e) => {
        const id = parseInt(e.currentTarget.getAttribute('data-
id'));
        removeFromCart(id);
    });
});
})
.catch(error => {
    console.error('Ошибка при загрузке товаров для корзины:', error);
});
}
}

async function renderProducts() {
try {
    const response = await fetch('https://footballgeer.ru/api/products');
    if (!response.ok) {
        throw new Error(`Ошибка загрузки товаров (${response.status})`);
    }
    const products = await response.json(); // Данные о товарах
    const popularProducts = products.filter(product => product.popular);
    let popularHTML = '';
    popularProducts.forEach(product => {
        popularHTML += `
            <div class="product-card bg-white rounded-lg shadow-sm overflow-
hidden transition duration-300">
                <div class="relative pb-3/4 h-48">
                    
                </div>
                <div class="p-4">

                    <h3 class="font-bold mb-1">${product.name}</h3>
                    <p class="text-gray-600 text-sm mb-
2">${product.category}</p>
                    <div class="flex justify-between items-center">
                        <span class="font-bold text-lg">${product.price}
₽</span>

                        <button class="add-to-cart bg-blue-500 hover:bg-blue-
600 text-white px-3 py-1 rounded-md text-sm transition" data-id="${product.id}">

```

					ПГУ 1.090301.24.001 ПЗ	Лист
						85
Изм.	Лист	№ докум.	Подпись	Дата		

```

        В корзину
      </button>
    </div>
  </div>
</div>
`;
});
elements.popularProductsContainer.innerHTML = popularHTML;
let catalogHTML = '';
products.forEach(product => {
  catalogHTML += `
    <div class="product-card bg-white rounded-lg shadow-sm overflow-
hidden transition duration-300">
      <div class="relative pb-3/4 h-48">
        
      </div>
      <div class="p-4">
        <h3 class="font-bold mb-1">${product.name}</h3>
        <p class="text-gray-600 text-sm mb-
2">${product.category}</p>
        <p class="text-gray-500 text-xs mb-3 line-clamp-
2">${product.description}</p>
        <div class="flex justify-between items-center">
          <span class="font-bold text-lg">${product.price}
₽</span>
          <button class="add-to-cart bg-blue-500 hover:bg-blue-
600 text-white px-3 py-1 rounded-md text-sm transition" data-id="${product.id}">
            В корзину
          </button>
        </div>
      </div>
    </div>
  `;
});
elements.catalogItemsContainer.innerHTML = catalogHTML;
document.querySelectorAll('.add-to-cart').forEach(button => {
  button.addEventListener('click', (e) => {
    const id = parseInt(e.currentTarget.getAttribute('data-id'));
    addToCart(id);
  });
});
});

} catch (error) {
  console.error('Ошибка при загрузке товаров:', error);
}

function renderAuthState() {
  if (state.isAuthenticated) {
    elements.authButton.textContent = 'Выйти';
    if (state.user) {

```

```

        document.getElementById('account-name').textContent = state.user.name
    || 'Пользователь';
        document.getElementById('account-email').textContent =
state.user.email || '';
    }
    } else {
        elements.authButton.textContent = 'Войти';
    }
}

function renderAccount() {
    if (state.isAuthenticated) {
        elements.accountNotAuth.classList.add('hidden');
        elements.accountContent.classList.remove('hidden');
    } else {
        elements.accountNotAuth.classList.remove('hidden');
        elements.accountContent.classList.add('hidden');
    }
}

async function renderAdminPanel() {
if (state.isAdmin) {
    elements.adminNotAuth.classList.add('hidden');
    elements.adminContent.classList.remove('hidden');

    // Загрузка товаров с учетом токена
    const products = await fetchProducts();

    // Рендер товаров
    let adminProductsHTML = '';
    products.forEach(product => {
        adminProductsHTML += `
            <tr>
                <td>${product.id}</td>
                <td></td>
                <td>${product.name}</td>
                <td>${product.category}</td>
                <td>${product.price} ₺</td>
                <td>
                    <button class="edit-product" data-id="${product.id}">
                        Редактировать
                    </button>
                    <button class="delete-product" data-id="${product.id}">
                        Удалить
                    </button>
                </td>
            </tr>
        `;
    });

    elements.adminProductsList.innerHTML = adminProductsHTML;
} else {
    elements.adminNotAuth.classList.remove('hidden');
    elements.adminContent.classList.add('hidden');
}
}

```

```

    }
}

function setupEventListeners() {
    // Переключение вкладок
    elements.tabLinks.forEach(link => {
        link.addEventListener('click', (e) => {
            e.preventDefault();
            state.currentTab = e.currentTarget.getAttribute('data-tab');
            saveState();
            render();
        });
    });
    elements.cartButton.addEventListener('click', () => {
        elements.cartModal.classList.remove('hidden');
    });

    elements.closeCart.addEventListener('click', () => {
        elements.cartModal.classList.add('hidden');
    });

    elements.checkoutButton.addEventListener('click', () => {
        if (state.cart.length === 0) return;

        if (state.isAuthenticated) {
            alert('Заказ оформлен! Спасибо за покупку!');
            state.cart = [];
            saveState();
            render();
            elements.cartModal.classList.add('hidden');
        } else {
            alert('Для оформления заказа необходимо авторизоваться');
            elements.cartModal.classList.add('hidden');
            elements.authModal.classList.remove('hidden');
        }
    });
    elements.authButton.addEventListener('click', () => {
        if (state.isAuthenticated) {
            // Выход
            state.isAuthenticated = false;
            state.isAdmin = false;
            state.user = null;
            saveState();
            render();
        } else {
            // Показ формы входа

            elements.authModal.classList.remove('hidden');
            elements.loginForm.classList.remove('hidden');
            elements.registerForm.classList.add('hidden');
            elements.forgotForm.classList.add('hidden');
            document.getElementById('auth-modal-title').textContent =
'Вход в аккаунт';
        }
    });
});

```

```

elements.closeAuth.addEventListener('click', () => {
    elements.authModal.classList.add('hidden');
});

elements.showRegister.addEventListener('click', () => {
    elements.loginForm.classList.add('hidden');
    elements.registerForm.classList.remove('hidden');
    elements.forgotForm.classList.add('hidden');
    document.getElementById('auth-modal-title').textContent =
'Регистрация';
});

elements.showLogin.addEventListener('click', () => {
    elements.loginForm.classList.remove('hidden');
    elements.registerForm.classList.add('hidden');
    elements.forgotForm.classList.add('hidden');
    document.getElementById('auth-modal-title').textContent = 'Вход в
аккаунт';
});

elements.showForgot.addEventListener('click', () => {
    elements.loginForm.classList.add('hidden');
    elements.registerForm.classList.add('hidden');
    elements.forgotForm.classList.remove('hidden');
    document.getElementById('auth-modal-title').textContent =
'Восстановление пароля';
});

elements.forgotBack.addEventListener('click', () => {
    elements.loginForm.classList.remove('hidden');
    elements.registerForm.classList.add('hidden');
    elements.forgotForm.classList.add('hidden');
    document.getElementById('auth-modal-title').textContent = 'Вход в
аккаунт';
});

elements.loginButton.addEventListener('click', async () => {
    try {
        const email = document.getElementById('login-
email').value;
        const password = document.getElementById('login-
password').value;

        const response = await
fetch('https://footballgeer.ru/api/login', {

            method: 'POST',
            headers: { 'Content-Type': 'application/json' },
            body: JSON.stringify({ email, password })
        });

        if (!response.ok) {

```

```

        throw new Error('Ошибка входа');
    }

    const data = await response.json();
    localStorage.setItem('token', data.token); // □ ДОБАВЬ ЭТО
    state.isAuthenticated = true;
    state.isAdmin = data.user.role === 'admin';
    state.user = data.user;
    saveState();

    console.log('Состояние после входа:', state);

    render();
  } catch (error) {
    console.error(error);
    alert('Ошибка входа');
  }
});

elements.registerButton.addEventListener('click', async () => {
  const name = document.getElementById('register-name').value;
  const email = document.getElementById('register-email').value;
  const password = document.getElementById('register-password').value;

  if (!name || !email || !password) {
    alert('Пожалуйста, заполните все поля');
    return;
  }

  try {
    const response = await fetch('https://footballgeer.ru/api/register', {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify({ name, email, password }),
    });

    const data = await response.json();
    if (response.ok) {
      alert('Регистрация прошла успешно!');
      elements.authModal.classList.add('hidden');
    } else {
      alert(data.error || 'Ошибка регистрации');
    }
  } catch (error) {
    console.error(error);
    alert('Ошибка при отправке данных');
  }
});

elements.forgotButton.addEventListener('click', () => {
  const email = document.getElementById('forgot-email').value;

```

```

        if (email) {
            elements.authModal.classList.add('hidden');
            alert(`Инструкции по восстановлению пароля отправлены на ${email}`);
        } else {
            alert('Пожалуйста, введите ваш email');
        }
    });

    elements.logoutButton.addEventListener('click', () => {
        state.isAuthenticated = false;
        state.isAdmin = false;
        state.user = null;
        saveState();
        render();
    });

    elements.accountLoginButton.addEventListener('click', () => {
        elements.authModal.classList.remove('hidden');
        elements.loginForm.classList.remove('hidden');
        elements.registerForm.classList.add('hidden');
        elements.forgotForm.classList.add('hidden');
        document.getElementById('auth-modal-title').textContent = 'Вход в
аккаунт';
    });

    elements.adminLoginButton.addEventListener('click', () => {
        elements.authModal.classList.remove('hidden');
        elements.loginForm.classList.remove('hidden');
        elements.registerForm.classList.add('hidden');
        elements.forgotForm.classList.add('hidden');
        document.getElementById('auth-modal-title').textContent = 'Вход
для администратора';
    });

    elements.mobileMenuButton.addEventListener('click', () => {
        elements.mobileMenu.classList.toggle('hidden');
    });

    elements.adminNavs.forEach(nav => {
        nav.addEventListener('click', (e) => {
            const tab = e.currentTarget.getAttribute('data-admin-tab');
            elements.adminNavs.forEach(n => {
                n.classList.remove('border-blue-500', 'text-blue-600');
                n.classList.add('border-transparent', 'text-gray-500');
            });
            e.currentTarget.classList.add('border-blue-500', 'text-blue-600');
            e.currentTarget.classList.remove('border-transparent', 'text-gray-500');
            elements.adminProducts.classList.add('hidden');
            elements.adminOrders.classList.add('hidden');
            elements.adminUsers.classList.add('hidden');
            document.getElementById(`admin-${tab}`).classList.remove('hidden');
        });
    });
});

```

```

async function renderAdminProducts() {
  try {
    const token = localStorage.getItem('token');
    if (!token) {
      alert('Требуется авторизация');
      return;
    }

    const response = await fetch('https://footballgeer.ru/api/products', {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
        'Authorization': 'Bearer ' + localStorage.getItem('token')
      },
      body: JSON.stringify({
        name: 'Пример',
        category: 'Test',
        price: 123,
        image: 'https://example.com/image.jpg',
        description: 'Описание',
        popular: true
      })
    });

    if (!response.ok) {
      throw new Error(`Ошибка загрузки товаров (${response.status})`);
    }

    const products = await response.json();

    let adminProductsHTML = '';
    products.forEach(product => {
      adminProductsHTML += `
        <tr>
          <td class="py-2 px-4 border-b">${product.id}</td>
          <td class="py-2 px-4 border-b"></td>
          <td class="py-2 px-4 border-b">${product.name}</td>
          <td class="py-2 px-4 border-b">${product.category}</td>
          <td class="py-2 px-4 border-b">${product.price} Р</td>
          <td class="py-2 px-4 border-b">
            <button class="text-blue-500 hover:text-blue-700 mr-2
edit-product" data-id="${product.id}">
              <i class="fas fa-edit"></i>
            </button>
            <button class="text-red-500 hover:text-red-700 delete-
product" data-id="${product.id}">
              <i class="fas fa-trash"></i>
            </button>
          </td>
        </tr>
      `;
    });
  }
}

```

					ПГУ 1.090301.24.001 ПЗ	Лист
						92
Изм.	Лист	№ докум.	Подпись	Дата		

```

        elements.adminProductsList.innerHTML = adminProductsHTML;
    } catch (error) {
        console.error(error);
        alert('Ошибка при загрузке товаров');
    }
    document.querySelectorAll('.delete-product').forEach(button => {
    button.addEventListener('click', async () => {
        const productId = button.getAttribute('data-id');
        const token = localStorage.getItem('token');

        if (!confirm('Вы точно хотите удалить этот товар?')) return;

        try {
            const response = await fetch(`/api/products/${productId}`, {
                method: 'DELETE',
                headers: {
                    'Authorization': `Bearer ${token}`,
                    'Content-Type': 'application/json'
                }
            });

            const result = await response.json();

            if (response.ok) {
                alert('Товар удалён');
                const row = button.closest('tr');
                if (row) row.remove();
            } else {
                alert(result.error || 'Ошибка при удалении');
            }
        } catch (error) {
            console.error('Ошибка удаления:', error);
            alert('Сервер не отвечает');
        }
    });
    });

    elements.addProductButton.addEventListener('click', () => {
    const form = document.getElementById('add-product-form');
    form.classList.toggle('hidden');
    });

    document.getElementById('save-product').addEventListener('click', async () => {
    const token = localStorage.getItem('token');

    if (!token) {
        alert('Вы не авторизованы. Пожалуйста, войдите в систему как администратор.');
```

```

        return;
    }

    const product = {
```

```

    name: document.getElementById('product-name').value,
    category: document.getElementById('product-category').value,
    price: parseFloat(document.getElementById('product-price').value),
    image: document.getElementById('product-image').value,
    description: document.getElementById('product-description').value,
    popular: document.getElementById('product-popular').checked
  };

  if (!product.name || !product.category || !product.price || !product.image) {
    alert('Пожалуйста, заполните все обязательные поля');
    return;
  }

  try {
    const response = await fetch('https://footballgeer.ru/api/products', {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
        'Authorization': `Bearer ${token}`
      },
      body: JSON.stringify(product)
    });

    const data = await response.json();

    if (response.ok) {
      alert('Товар успешно добавлен');
      document.getElementById('add-product-form').classList.add('hidden');
      renderAdminPanel();
    } else {
      alert(data.error || 'Ошибка при добавлении товара');
    }
  } catch (error) {
    console.error(error);
    alert('Ошибка при отправке запроса');
  }
});

function addToCart(id) {

  const existingItem = state.cart.find(item => item.id === id);

  if (existingItem) {
    existingItem.quantity += 1;
  } else {
    state.cart.push({ id, quantity: 1 });
  }

  saveState();
  render();
  const button = document.querySelector(`.add-to-cart[data-id="${id}"]`);
  if (button) {

```

```

        button.innerHTML = '<i class="fas fa-check"></i> Добавлено';
        button.classList.remove('bg-blue-500');
        button.classList.add('bg-green-500');

        setTimeout(() => {
            button.innerHTML = 'В корзину';
            button.classList.remove('bg-green-500');
            button.classList.add('bg-blue-500');
        }, 2000);
    }
}

function removeFromCart(id) {
    state.cart = state.cart.filter(item => item.id !== id);
    saveState();
    render();
}

function increaseQuantity(id) {
    const item = state.cart.find(item => item.id === id);
    if (item) {
        item.quantity += 1;
        saveState();
        render();
    }
}

function decreaseQuantity(id) {
    const item = state.cart.find(item => item.id === id);
    if (item) {
        if (item.quantity > 1) {
            item.quantity -= 1;
        } else {
            state.cart = state.cart.filter(item => item.id !== id);
        }
        saveState();
        render();
    }
}

document.addEventListener('DOMContentLoaded', init);
</script>
</body>
</html>

```

Файл db.js

```

const { Pool } = require("pg");
require("dotenv").config();

const pool = new Pool({
    user: process.env.DB_USER,
    host: process.env.DB_HOST,
    database: process.env.DB_NAME,
    password: process.env.DB_PASSWORD,

```

					<p style="text-align: center;"><i>ПГУ 1.090301.24.001 ПЗ</i></p>	Лист
						95
Изм.	Лист	№ докум.	Подпись	Дата		

```

    port: process.env.DB_PORT,
    client_encoding: "utf8",
  });

```

Файл server.js

```

const express = require('express');
const jwt = require('jsonwebtoken');
const bcrypt = require('bcrypt');
const { Pool } = require('pg');
const path = require('path');
require('dotenv').config();
const app = express();
const PORT = process.env.PORT || 5000;
const authenticateToken = require('./routes/auth');
app.use(express.json());
app.use(express.static(path.join(__dirname, 'public')));
app.get('/', (req, res) => {
  res.sendFile(path.join(__dirname, 'public', 'index.html'));
});
const pool = new Pool({
  user: process.env.DB_USER,
  host: process.env.DB_HOST,
  database: process.env.DB_NAME,
  password: process.env.DB_PASSWORD,
  port: process.env.DB_PORT,
});
const JWT_SECRET = process.env.JWT_SECRET;
function isAdmin(req, res, next) {
  if (req.user.role !== 'admin') {
    return res.status(403).json({ error: 'Недостаточно прав' });
  }
  next();
}
app.post('/api/register', async (req, res) => {
  try {
    const { name, email, password } = req.body;
    const hashedPassword = await bcrypt.hash(password, 10);
    const query = 'INSERT INTO users (name, email, password, role) VALUES ($1, $2, $3, $4) RETURNING *';
    const values = [name, email, hashedPassword, 'user'];
    const { rows } = await pool.query(query, values);
    res.json({ message: 'Пользователь успешно зарегистрирован', user: rows[0] });
  } catch (err) {
    console.error(err);
    res.status(500).json({ error: 'Ошибка сервера' });
  }
});
app.post('/api/login', async (req, res) => {
  try {
    const { email, password } = req.body;

```

					<p style="text-align: center;"><i>ПГУ 1.090301.24.001 ПЗ</i></p>	Лист
						96
Изм.	Лист	№ докум.	Подпись	Дата		

```

    const user = await pool.query('SELECT * FROM users WHERE email = $1',
[email]);
    if (user.rows.length === 0) {
        return res.status(400).json({ error: 'Пользователь не найден' });
    }
    const userRow = user.rows[0];
    const isPasswordValid = await bcrypt.compare(password, userRow.password);
    if (!isPasswordValid) {
        return res.status(401).json({ error: 'Неверный пароль' });
    }
    const token = jwt.sign(
        { id: userRow.id, role: userRow.role },
        JWT_SECRET,
        { expiresIn: '1h' }
    );
    res.json({
        message: 'Вход выполнен успешно',
        token,
        user: {
            id: userRow.id,
            name: userRow.name,
            email: userRow.email,
            role: userRow.role
        }
    });
} catch (err) {
    console.error(err);
    res.status(500).json({ error: 'Ошибка сервера' });
}
});
app.get('/api/products', async (req, res) => {
    try {
        const query = 'SELECT * FROM products';
        const { rows } = await pool.query(query);
        res.json(rows);
    } catch (err) {
        console.error(err);
        res.status(500).json({ error: 'Ошибка сервера' });
    }
});
app.post('/api/products', authenticateToken, isAdmin, async (req, res) => {
    try {
        const { name, category, price, image, description, popular } = req.body;
        const query = `
            INSERT INTO products (name, category, price, image, description, popular)
            VALUES ($1, $2, $3, $4, $5, $6)
            RETURNING *
        `;
        const values = [name, category, price, image, description, popular];
        const { rows } = await pool.query(query, values);
        res.json({ message: 'Товар добавлен', product: rows[0] });
    } catch (err) {

```

```

        console.error(err);
        res.status(500).json({ error: 'Ошибка сервера' });
    }
});
app.put('/api/products/:id', authenticateToken, isAdmin, async (req, res) => {
    try {
        const { id } = req.params;
        const { name, category, price, image, description, popular } = req.body;
        const query = `
            UPDATE products
            SET name = $1, category = $2, price = $3, image = $4, description =
$5, popular = $6
            WHERE id = $7
            RETURNING *
        `;
        const values = [name, category, price, image, description, popular, id];
        const { rows } = await pool.query(query, values);
        if (rows.length === 0) {
            return res.status(404).json({ error: 'Товар не найден' });
        }
        res.json({ message: 'Товар обновлен', product: rows[0] });
    } catch (err) {
        console.error(err);
        res.status(500).json({ error: 'Ошибка сервера' });
    }
});
app.delete('/api/products/:id', authenticateToken, isAdmin, async (req, res) => {
    try {
        const { id } = req.params;
        const query = 'DELETE FROM products WHERE id = $1 RETURNING *';
        const { rows } = await pool.query(query, [id]);
        if (rows.length === 0) {
            return res.status(404).json({ error: 'Товар не найден' });
        }
        res.json({ message: 'Товар удалён', product: rows[0] });
    } catch (err) {
        console.error(err);
        res.status(500).json({ error: 'Ошибка сервера' });
    }
});
app.get('/api/users', authenticateToken, isAdmin, async (req, res) => {
    try {
        const { rows } = await pool.query('SELECT * FROM users');
        res.json(rows);
    } catch (err) {
        console.error(err);
        res.status(500).json({ error: 'Ошибка сервера' });
    }
});
app.put('/api/users/:id/role', authenticateToken, isAdmin, async (req, res) => {
    try {
        const { id } = req.params;
        const { role } = req.body;

```

```

const query = 'UPDATE users SET role = $1 WHERE id = $2 RETURNING *';
const { rows } = await pool.query(query, [role, id]);
if (rows.length === 0) {
  return res.status(404).json({ error: 'Пользователь не найден' });
}
res.json({ message: 'Роль обновлена', user: rows[0] });
} catch (err) {
  console.error(err);
  res.status(500).json({ error: 'Ошибка сервера' });
}
});
app.listen(PORT, () => {
  console.log(`Сервер запущен на http://localhost:${PORT}`);
});

```

Приложение В

(обязательное)

Веб-приложение «Онлайн-магазин футбольной экипировки»

Презентация

					ПГУ 1.090301.24.001 ПЗ	Лист
						100
Изм.	Лист	№ докум.	Подпись	Дата		

Обзор проекта интернет-магазина футбольной экипировки

Автор работы:

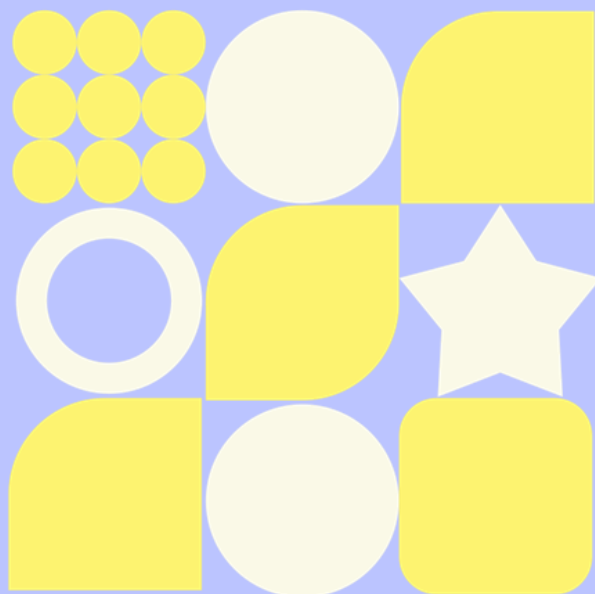
студент гр. 21ВВП1

Щеглов Дмитрий Алексеевич

Научный руководитель:

к.э.н., доцент кафедры ВТ ПГУ

Акифьев И.В.



Актуальность и цели проекта



Современные онлайн-магазины спортивной экипировки имеют недостатки. Задача — разработать улучшенное, стабильное решение с удобным интерфейсом и надежной архитектурой.



2

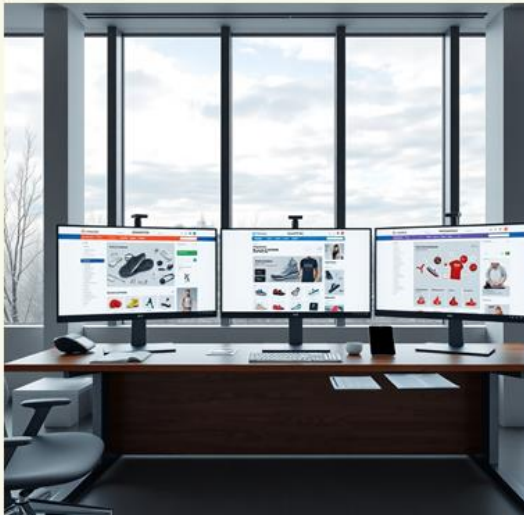
Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.24.001 ПЗ

Лист

101

Анализ рынка и выявление проблем



Исследование конкурентов

Изучены три ведущих магазина: Sportsmaster, Decathlon, desport.ru. Оценены интерфейс, скорость, фильтры, пользователи и функционал корзины.



Определение требований к системе

Выявлены недостатки: слабая персонализация, неудобство администрирования, снижение производительности при нагрузках. Созданы требования для лучшего пользовательского опыта.



3

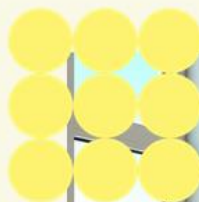
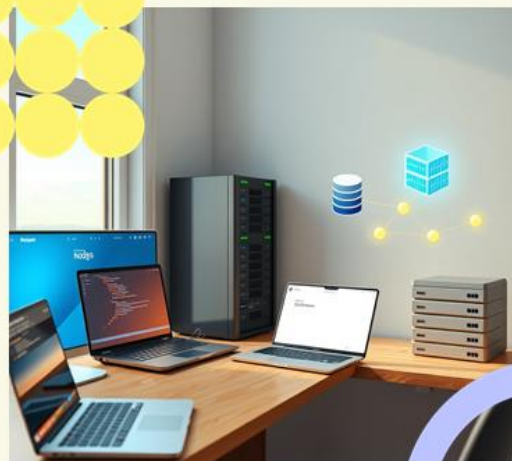
Выбор технологий

1

Node.js с Express обеспечивает производительный и однородный сервер на JavaScript. PostgreSQL отвечает за надежное хранение и управление связями данных.

2

Redis уменьшает задержки через кэширование. Tailwind CSS и Vanilla JS создают легкий, быстрый интерфейс без сложных фреймворков.



4

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.24.001 ПЗ

Лист

102

Архитектура приложения и UML-диаграммы

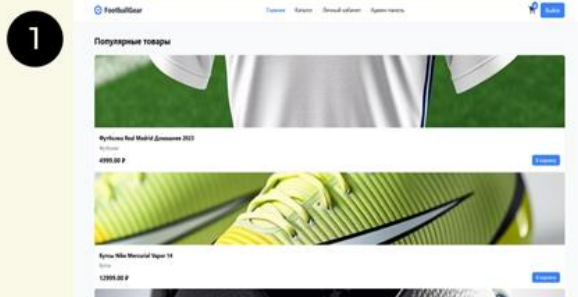
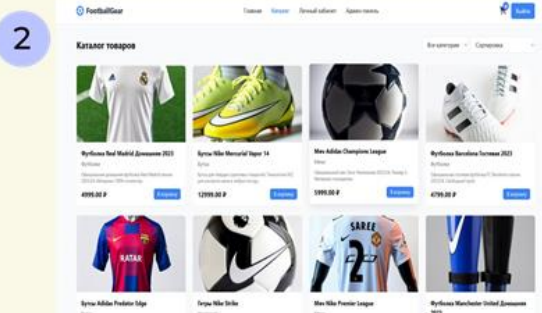
1 Приложение разделено на клиентскую и серверную части с коммуникацией через RESTful API для гибкости и масштабируемости.

2 UML-диаграммы отражают роли пользователей и процессы регистрации, заказов и администрирования, облегчая понимание и поддержку.



5

Интерфейс приложения



6

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.24.001 ПЗ

Лист

103

Реализация функционала и алгоритмы

Внедрена регистрация и авторизация с JWT и bcrypt для безопасности, реализованы операции управления корзиной и заказами.

Организован CRUD для администрирования товаров, написаны корректные SQL-запросы.



7

Тестирование и безопасность жизнедеятельности

1 Функциональное тестирование
Проверены регистрация, авторизация, корзина и администрирование. Результаты положительные, ошибки обрабатываются корректно, права доступа соблюдены.

2 Организация рабочего места
Режим труда включает оптимальное освещение, микроклимат и профилактические перерывы для защиты здоровья и повышения продуктивности.



8

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.24.001 ПЗ

Лист

104

Итоги и дальнейшее развитие

Проект успешно реализован: функциональный интернет-магазин с безопасной и масштабируемой архитектурой. Планируется мобильная версия и интеграция новых сервисов.

Спасибо за внимание

					<i>ПГУ 1.090301.24.001 ПЗ</i>	Лист
						105
Изм.	Лист	№ докум.	Подпись	Дата		

