Министерство образования Иркутской области

Государственное бюджетное профессиональное

образовательное учреждение Иркутской области

«Иркутский авиационный техникум»

(ГБПОУИО «ИАТ»)

КР.09.02.07-5.24.222.15 ПЗ

ВЕБ-ПРИЛОЖЕНИЕ

«МАГАЗИН КОМПЬЮТЕРНОЙ ПЕРИФЕРИИ»

|  |  |  |
| --- | --- | --- |
| Председатель ВЦК: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (подпись, дата) | (М.А. Кудрявцева) |
| Руководитель: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (подпись, дата) | (Н.Р. Карпова) |
| Студент: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (подпись, дата) | (Е.Ю. Лиханова) |

Иркутск 2024

Содержание

Введение 3

1 Описание предметной области 5

2 Анализ инструментальных средств разработки 8

3 Техническое задание 12

4 Проектирование ВЕБ 13

4.1 Структурная схема веб-приложение 13

4.2 Функциональная схема ВЕБ 17

4.3 Проектирование базы данных 20

4.4 Проектирование интерфейса 24

5 Разработка ВЕБ 28

5.1 Разработка интерфейса ВЕБ 28

5.2 Разработка базы данных ВЕБ 30

5.3 Разработка ВЕБ 33

6 Документирование программного продукта 37

6.1 Руководство пользователя ВЕБ 37

Заключение 42

Список используемых источников 43

Приложение А – Техническое задание 44

Приложение Б – Листинг кода контроллера товара 50

**Введение**

В современных условиях рынок компьютерной периферии является одним из наиболее конкурентных и динамичных сегментов электронной коммерции. Для успешного ведения бизнеса в данной отрасли, компаниям необходимо постоянно совершенствовать и оптимизировать свое веб-приложение, уделяя особое внимание внедрению инновационных технологий и автоматизации рутинных операций.

Компьютерная периферия, такие как мыши, клавиатуры, гарнитуры являются важными элементами, которые обеспечивают комфорт и эффективность работы пользователей. Они обладают широкими возможностями и функциональностью, которые можно настраивать в соответствии с индивидуальными потребностями пользователя. Однако выбор подходящего устройства из множества вариантов может быть затруднительным, особенно для тех, кто не обладает глубокими знаниями в этой области.

В таких случаях на помощь приходят магазины компьютерной периферии, предлагающие широкий выбор устройств в различных ценовых сегментах, с подробными характеристиками и возможностью быстрого выбора подходящего варианта. Магазины компьютерной периферии являются неотъемлемой частью рынка компьютерной техники, предлагая удобные решения для различных категорий потребителей.

С помощью результата курсовой работы потребитель программного продукта сможет быстрее выбрать более подходящий вариант компьютерной периферии для своих потребностей. Потребитель сможет легко и быстро изучить основную информацию о характеристиках устройств, тем самым найти нужный ему вариант по цене и качеству.

Целью курсовой работы является разработка веб-приложения «Магазин компьютерной периферии», которое позволит повысить производительность, снизить трудозатраты, улучшить качество обслуживания.

Для осуществления обозначенной цели служат следующие задачи:

* Описание предметной области.
* Анализ инструментальных средств разработки.
* Техническое задание.
* Проектирование веб-приложение.
* Разработка веб-приложение.
* Документирование программного продукта.

# **Описание предметной области**

Предметной областью курсовой работы является автоматизация работы магазина компьютерной периферии.

Под периферийными устройствами предполагается – любое вспомогательное устройство, которое подключается к компьютеру и работает с ним:

* либо для ввода/вывода информации в него;
* либо для извлечения информации.

Автоматизация магазина представляет собой информацию о различных устройствах и аксессуарах. Он предназначен для помощи пользователям и специалистам в выборе необходимых устройств/аксессуар.

Автоматизация магазина содержит информацию о клавиатурах, мышек мониторах, проводной гарнитуре и ковриках для мышек. Пользователь сможет ознакомиться с различными моделями, чтобы выбрать подходящие для него устройства/аксессуар. В веб-приложение пользователи могут найти характеристики каждого устройства/аксессуара, чтобы выбрать то, которое наиболее соответствует их потребностям.

Объектами данной предметной области выступают администратор и пользователь.

Администратор – лицо ответственное за управление и обслуживание автоматизация работы магазина компьютерной периферии.

Администратор имеет доступ к базе данных веб-приложения и может управлять ее содержимым. Функционал для администратора включает в себя возможность удаления, редактирования и создания новых товаров на странице через специальную панель администратора. Администратор добавляет картинку, описание и цену. Таким образом, администратор обеспечивает актуальность и достоверность информации в каталоге.

Пользователь ­­­– это человек, который использует каталог компьютерной периферии для просмотра товаров.

Функционал для пользователя включает возможность просмотра сайта и товаров. Пользователь может просматривать доступные категории товаров, переходить на страницы с описанием каждого устройства, просматривать изображения и технические характеристики.

Администратор, Пользователь и Веб-приложение взаимодействуют между собой для обеспечения функциональности и удобства использования каталога компьютерной периферии. Вот описание взаимодействия между этими объектами:

1. Администратор:

* Входит в веб-приложения с помощью учетных данных.
* Имеет доступ к панели администратора, где может управлять содержимым веб-приложения.
* Добавляет новый товар указывая его название, описание, изображения и цены.
* Редактирует информацию о существующих товарах.
* Удаляет товары, которые больше не доступны для продажи.
* Добавляет новую категорию товара указывая его название и изображение.
* Редактирует существующие категории.
* Удаляет категорию.

1. Пользователь:

* Регистрируется и авторизуется в веб-приложении.
* Посещает веб-приложение.
* Просматривает категории товаров, может пользоваться поиском.
* Переходит на страницы товаров для получения информации, включая описание, изображения и цены.
* Оформляет заказ.

1. Веб-приложение:

* Обеспечивает доступ к автоматизации работы магазина компьютерной периферии через веб-браузер.
* Авторизует администратора, позволяя ему входить в систему и использовать функциональность приложения.
* Отображает категории товаров, результаты поиска и страницы товаров с соответствующей информацией и изображениями.
* Предоставляет возможность поиска товаров.
* Взаимодействует с базой данных для хранения информации о товарах и других сущностях.

В соответствии с предметной областью «Автоматизация работы магазина компьютерной периферии» можно выделить базовые сущности проектируемого программного продукта:

1. Администратор.
2. Пользователь.
3. Товары.
4. Заказы.
5. Детали заказа.
6. Корзина.

* Атрибуты пользователя: код пользователя, имя, фамилия, отчество, номер телефона, логин, пароль, роль: 1-админ, 2-пользователь.
* Атрибуты товары: код товара, название товара, описание товара, цена, изображение товара.
* Атрибуты заказы: код заказа, дата заказа, цена, статус заказа.
* Атрибуты заказы: код детали заказа, количество, цена, дата, дата обновления.
* Атрибуты корзины: код корзины, имя пользователя, название товара.

# **Анализ инструментальных средств разработки**

Для создания веб-приложения понадобиться выбрать язык программирования для разработки, а таrже определится с СУБД для создания базы данных.

Для создания серверной части программного продукта используются веб-инструменты. Был рассмотрен язык программирования PHP.

PHP – это язык программирования, который позволяет создавать веб-приложения любой сложности при помощи готовых движков или фреймворков.

Этот язык достаточно гибким и мощным, поэтому он такой популярный и используется в проектах любого масштаба: от простого блога до крупнейших веб-приложений в Интернете.

JavaScript – следующий язык программирования, это язык программирования, который позволяет делать веб-приложения динамичными и интерактивными. С его помощью сайты делают интерактивными: добавляют всплывающие окна, анимацию, кнопки лайков и формы для отправки информации.

В процессе анализа инструментальных средств разработки было проведено сравнение языков. В таблице 1 представлено сравнение двух языков программирования PHP и JavaScript.

Таблица 1 – Сравнение языков программирования

|  |  |  |
| --- | --- | --- |
| Характеристики | PHP | JavaScript |
| Простота использования | + | – |
| Безопасность | – | + |
| Производительность | + | – |
| Обработка данных | + | – |

После анализа сравнительной таблицы было принято решение использовать PHP, так как он удобен в использовании.

Laravel – это бесплатный PHP-фреймворк с открытым исходным кодом, специально разработанный для создания сложных сайтов и веб-приложений. Он позволяет упростить: аутентификацию; маршрутизацию; сессии; кэширование; архитектуру приложения; работу с базой данных.

Symfony – свободный фреймворк, написанный на PHP. Symfony предлагает быструю разработку и управление веб-приложениями, позволяет легко решать рутинные задачи веб-программиста.

В процессе анализа инструментальных средств разработки было проведено сравнение фреймворков. В таблице 2 представлено сравнение двух фреймвоков Laravel и Symfony.

Таблица 2 – Сравнение фреймворков.

|  |  |  |
| --- | --- | --- |
| Характеристики | Laravel | Symfony |
| Сложность изучения | + | – |
| Производительность | + | + |
| Безопасность | + | + |

После анализа сравнительной таблицы было принято решение использовать Laravel.

Так как автоматизация веб-приложения «Магазин компьютерной периферии» разрабатывается с помощью веб-инструментов то необходимо использовать HTML и CSS.

HTML – это язык гипертекстовой разметки текста.

Он нужен, чтобы размещать на веб-странице элементы: текст, картинки, таблицы и видео.

CSS – это язык разметки, используемый для визуального оформления веб-сайтов. Он отвечает за то, как выглядят объекты, расположенные на странице: их размер, цвет, фоновое изображение, степень прозрачности, расположение относительно других элементов, поведение при наведении курсора, визуальное изменение кнопок при нажатии и т.п.

В проекте необходимо использовать среду разработки. Для этого было проведено сравнение интегрированных сред разработки (таблица 1).

Atom – это многофункциональный текстовый редактор от разработчиков GitHub. Он поддерживает огромное количество различных расширений, благодаря которым его можно сравнить с настоящей средой разработки.

Visual Studio Code – это редактор исходного кода. Его разработал Microsoft для всех популярных операционных систем: Windows, Linux и macOS.

В процессе анализа инструментальных средств разработки было проведено сравнение редакторов кода. В таблице 2 представлено сравнение двух редакторов кода Visual Studio Code и Atom.

Таблица 3 – Сравнение текстовых редакторов.

|  |  |  |
| --- | --- | --- |
| Характеристики | Visual Studio Code | Atom |
| Завершение кода | + | +– |
| Простота использования | + | +– |
| Тестирование | + | – |
| Расширения | + | – |

После анализа сравнительной таблицы было принято решение использовать Visual Studio Code, так как редактор удобен в использовании, имеет понятный интерфейс.

На этапе проектирования программного продукта использовались CASE-средства.

CASE-средства – это методы и технологии, которые позволяют проектировать различные программные продукты (базы данных) и автоматизировать их создание. В работе были использованы Draw.io.

Draw.io – бесплатная кроссплатформенное программное обеспечение для рисования графиков с открытым исходным кодом.

Для работы с базами данными использовался инструмент MySQL Workbench.

PhpMyAdmin – это веб-приложение с открытым кодом, написанное на языке PHP и представляющее собой веб-интерфейс для администрирования СУБД MySQL.

MySQL – это система управления реляционными базами данных с открытым исходным кодом с моделью клиент-сервер.

В процессе анализа инструментов было проведено сравнение СУБД. В таблице 4 представлено сравнение двух СУБД MongoDB и MySQL.

Таблица 4 – Сравнение средств реализации базы данных

|  |  |  |
| --- | --- | --- |
| Название СУБД | MySQL | MongoDB |
| Популярность | + | + |
| Отказоустойчивость | + | – |
| Простота использования | + | + |
| Многофункциональность | + | + |
| Масштабируемость | + | – |

После анализа сравнительной таблицы было принято решение использовать MySQL т.к. данное СУБД более удобное и широко используемое среди СУБД.

Для создания программного продукта было решено использовать средства:

1. Для создания структурных схем, контекстной и диаграмм декомпозиции использовались CASE-средства – Draw.io.
2. Для наглядного составления структуры базы данных использовался инструмент для визуального проектирования баз данных MySQL и создание ER-модели с помощью инструмента PhpMyAdmin.
3. Для разработки дизайна использовался онлайн-сервис Figma.
4. Для разработки прототипа веб-приложения использовался онлайн-сервис Draw.io.

# **Техническое задание**

В начале разработки создавалось техническое задание, в котором указывались основные требования.

Для создания технического задания использовался стандарт ГОСТ 34.602-2020.

Согласно ГОСТ 34.602-2020 техническое задание должно включать следующие разделы:

1. Введение.
2. Основание для разработки.
3. Назначение веб-приложения.
4. Требования к системе веб-приложения.
5. Требование к техническому обеспечению.
6. Требования к программному обеспечению.
7. Организационно-технические требования.

Техническое задание на разработку Веб-приложения представлено в приложении А.

# **Проектирование веб-приложение**

# **Структурная схема веб-приложение**

Проектирование Веб-приложения начинается с построения диаграммы прецедентов (рисунок 1). Она содержит 2 актера, которые могут выполнять суммарно 11 функций, часть выполняется только определенными актерами.

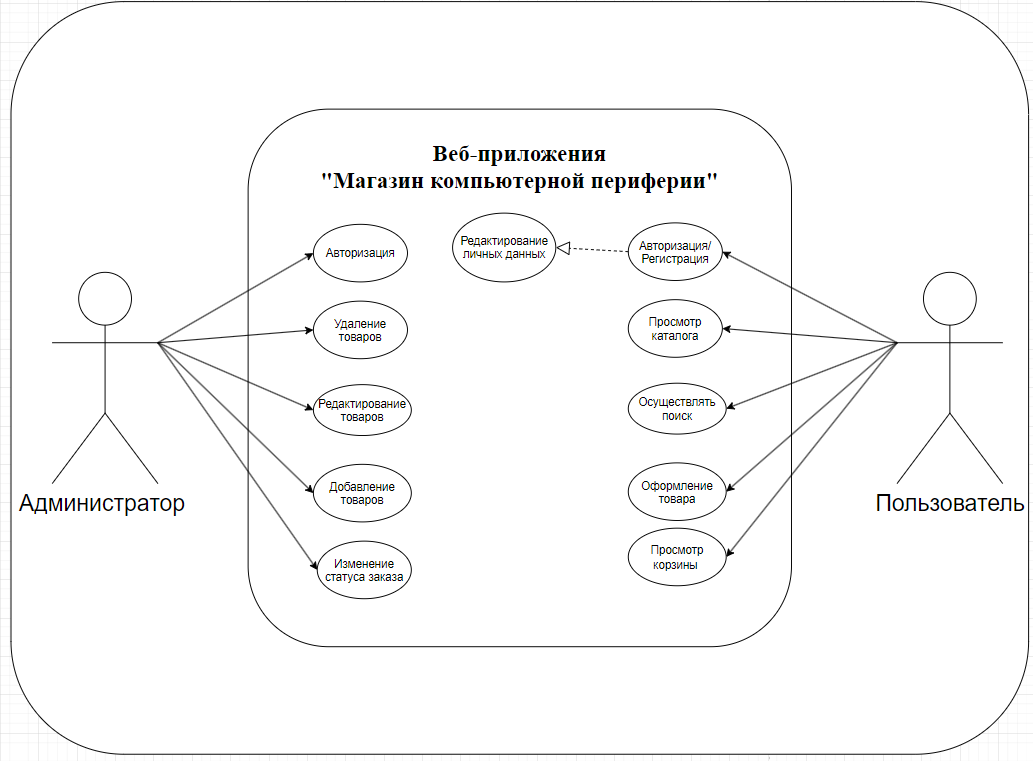


Рисунок 1 – Диаграмма прецедентов

На рисунке 2 представлена диаграмма взаимодействия. Она содержит 4 роли. Процесс начинается с пользователя, который авторизуется. Далее пользователь собирает товары необходимые для себя и добавляет их в корзину. Администратор участвует в процессе с помощью: просмотра пользователей и их корзины. Пользователь оформляет заказ, веб-приложение обрабатывает и выдает статус к заказу.

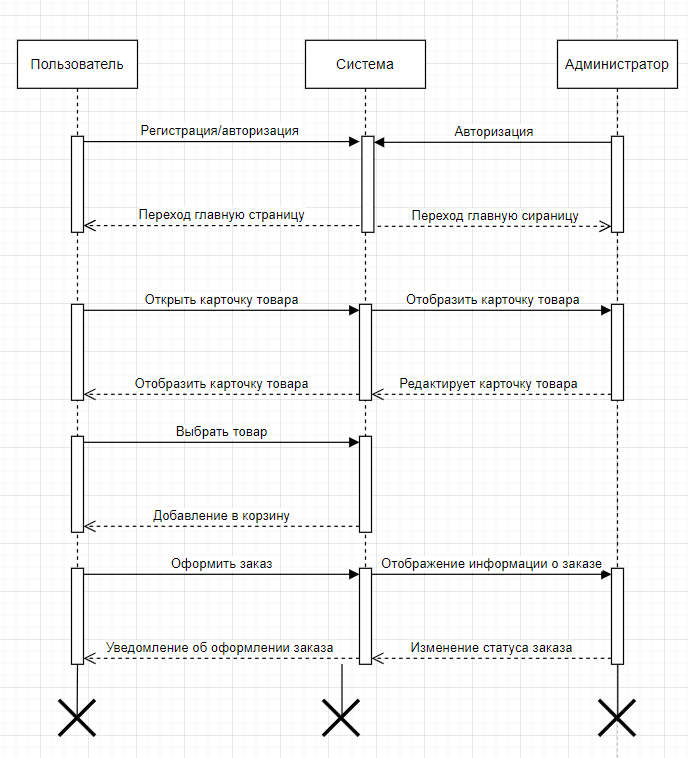


Рисунок 2 – Диаграмма взаимодействия

На рисунке 3 представлена диаграмма компонентов. Взаимодействие компонентов:

* Модели взаимодействуют с базой данных, предоставляя данные для контроллеров.
* Контроллеры обрабатывают запросы пользователей, взаимодействуют с моделями для получения или сохранения данных и передают результаты в представления (views).
* Представления отображают данные пользователю, используя шаблоны и данные, полученные от контроллеров.

Диаграмма компонентов наглядно демонстрирует, как различные компоненты (модели, контроллеры и представления) взаимодействуют друг с другом в веб-приложении магазина компьютерной периферии.

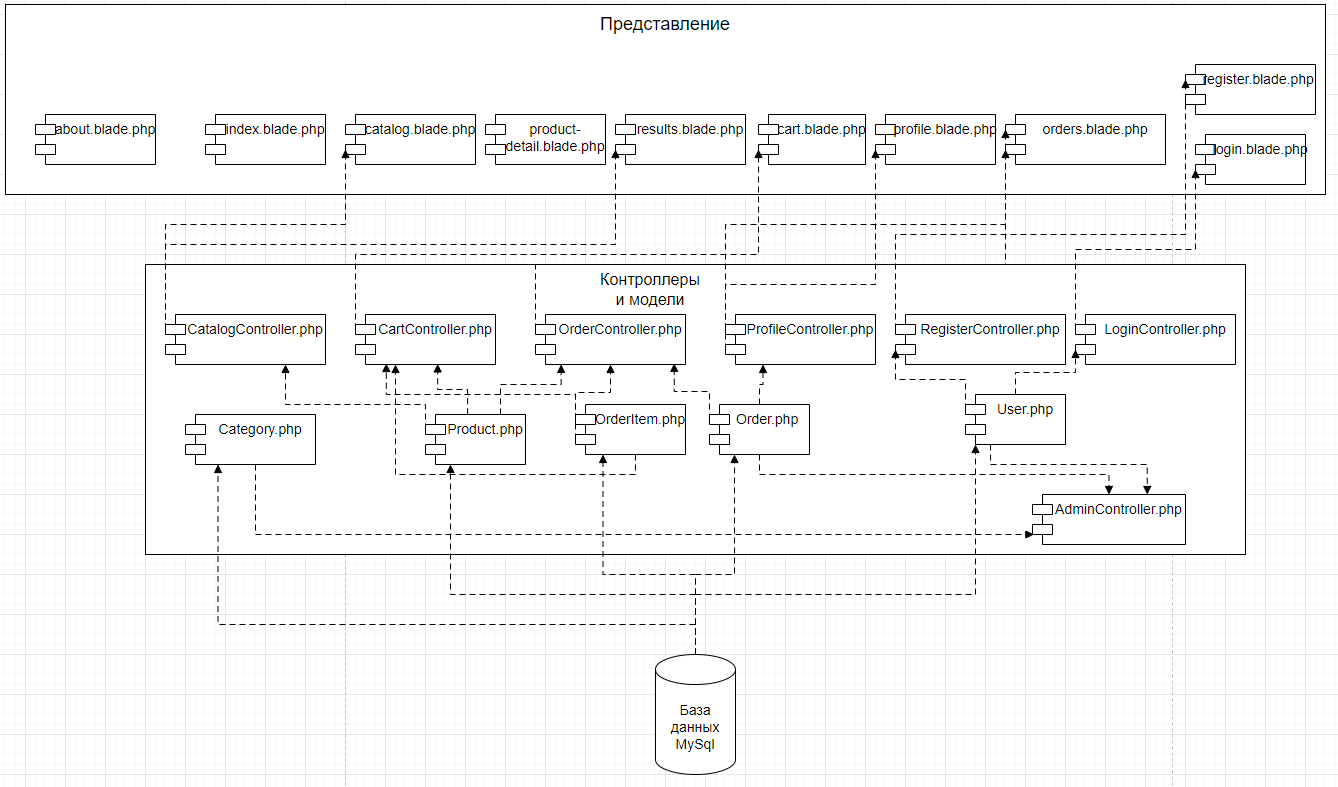


Рисунок 3 – Диаграмма компонентов

На рисунке 4 представлена диаграмма развертывания. Диаграмма развертывания показывать сервер, сервер базы данных, клиентское устройство и какие программные компоненты работают на каждом узле.

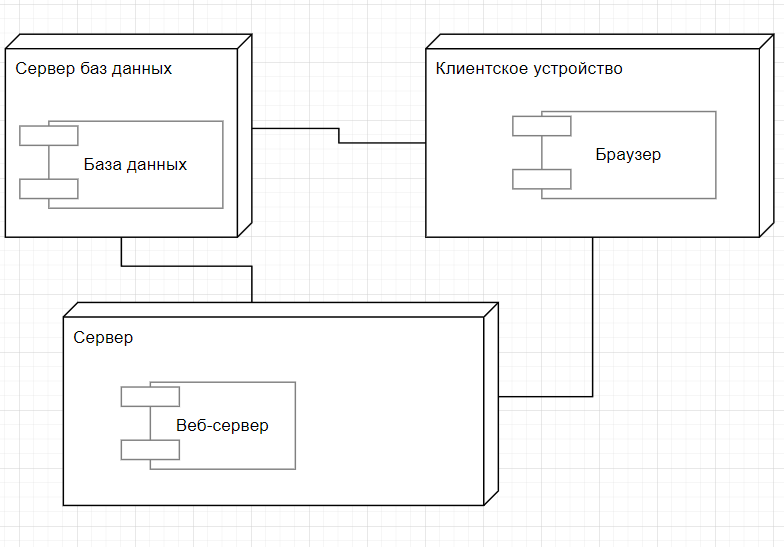
****

Рисунок 4 **–** Диаграммы развертывания

На рисунке 5 представлена диаграмма деятельности, которая описывает последовательность действий и взаимодействие между пользователем, базой данных и администратором при выполнении процесса оформления заказа в веб-приложении магазина компьютерной периферии. Диаграмма деятельности позволяет наглядно увидеть, как различные компоненты системы взаимодействуют друг с другом, чтобы обеспечить успешную обработку заказа.

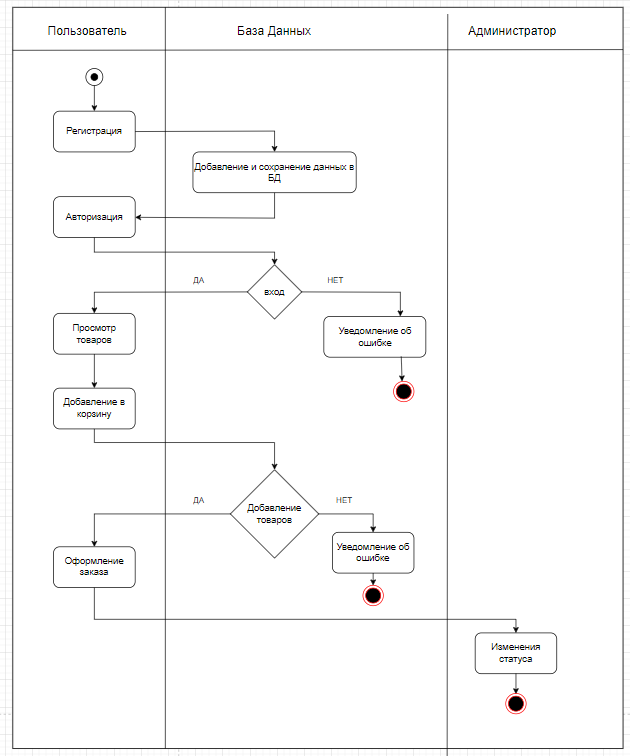


Рисунок 5 **–** Диаграмма деятельности

# **Функциональная схема ВЕБ**

На рисунке 6 представлена контекстная диаграмма на рисунке 6 предоставляет высокоуровневый обзор взаимодействия веб-приложения с внешними сущностями, отображая основные потоки данных, управления и механизмов, необходимых для функционирования системы.

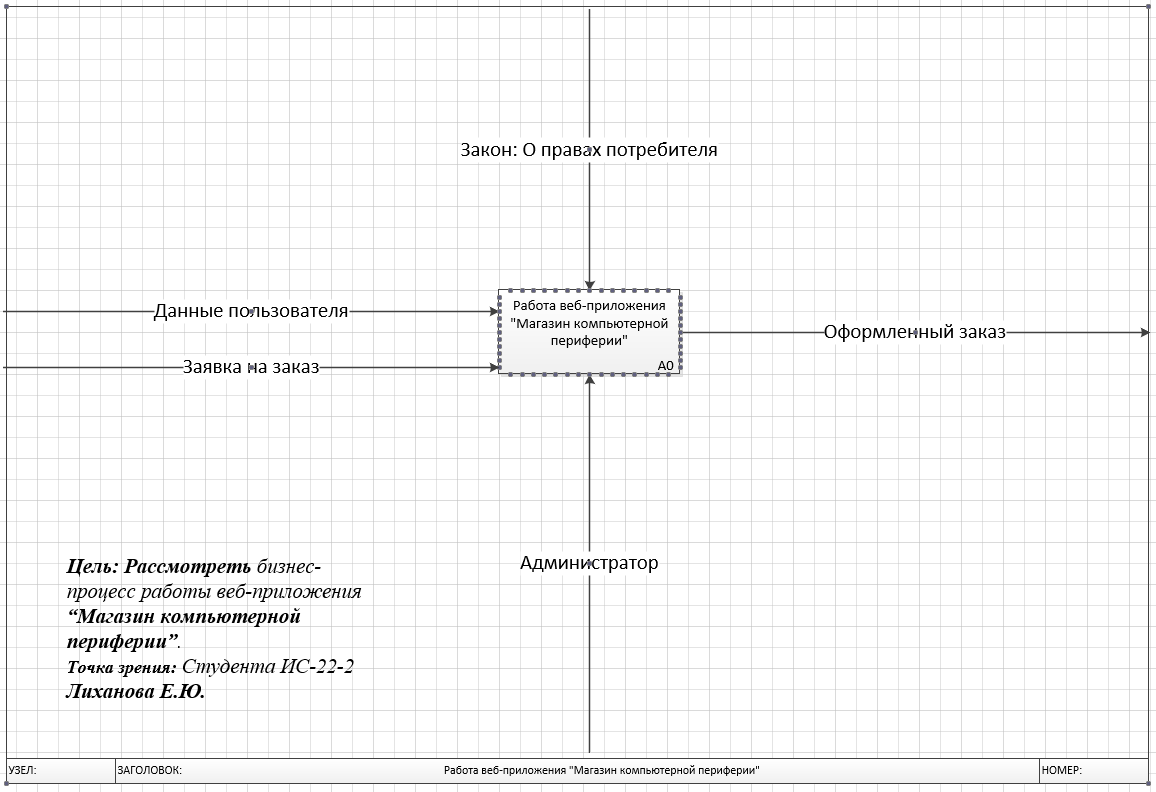


Рисунок 6 – Контекстная диаграмма IDEF0

Диаграмма декомпозиций (А1), представленная на рисунке 7, является частью модели IDEF0 и предназначена для детализации работы, которая была описана на более высоком уровне в контекстной диаграмме. В данном случае, диаграмма декомпозиций А1 детализирует процесс "Просмотр каталога" для аутентифицированного пользователя в веб-приложении "Магазин компьютерной периферии".

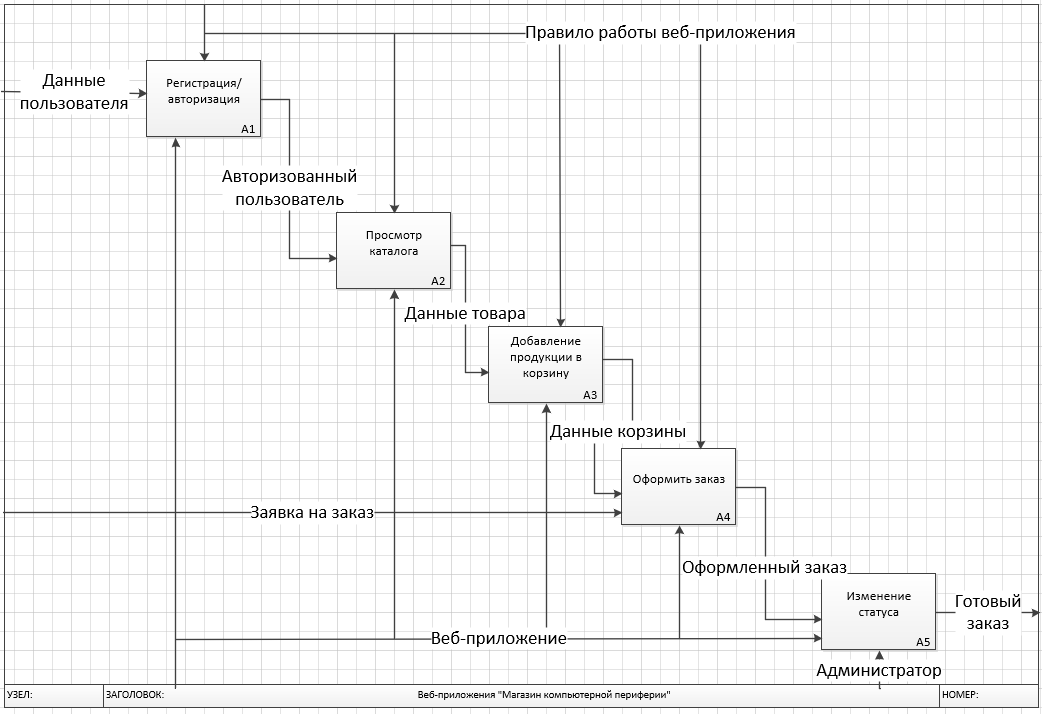


Рисунок 7 **–** Диаграмма декомпозиций А1

На рисунке 8 представлена диаграмма классов, она является важным инструментом для описания структуры базы данных и взаимосвязей между различными сущностями (классами) в системе управления продуктами, пользователями, заказами и корзиной. Эта диаграмма помогает понять, как данные организованы и как различные компоненты системы взаимодействуют друг с другом.

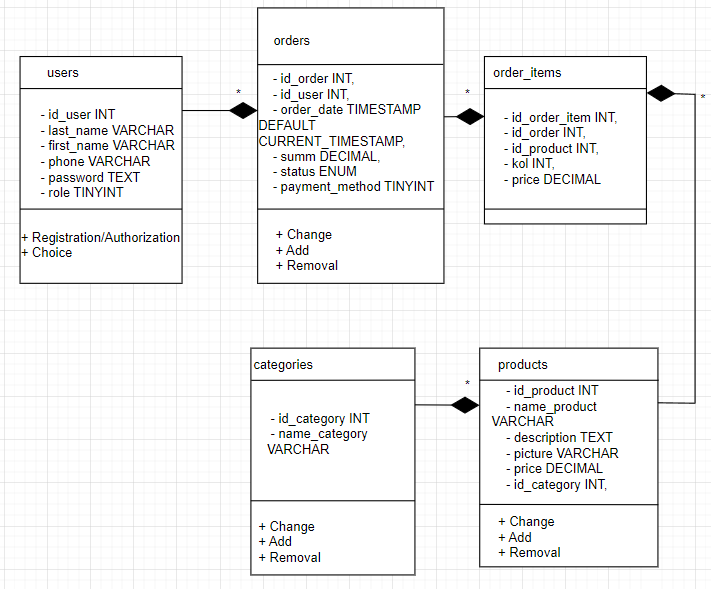


Рисунок 8 – Диаграмма классов

На рисунке 9 представлена диаграмма потоков данных (DFD), отображающая моделирования веб-приложения с точки зрения хранения, обработки и передачи данных.

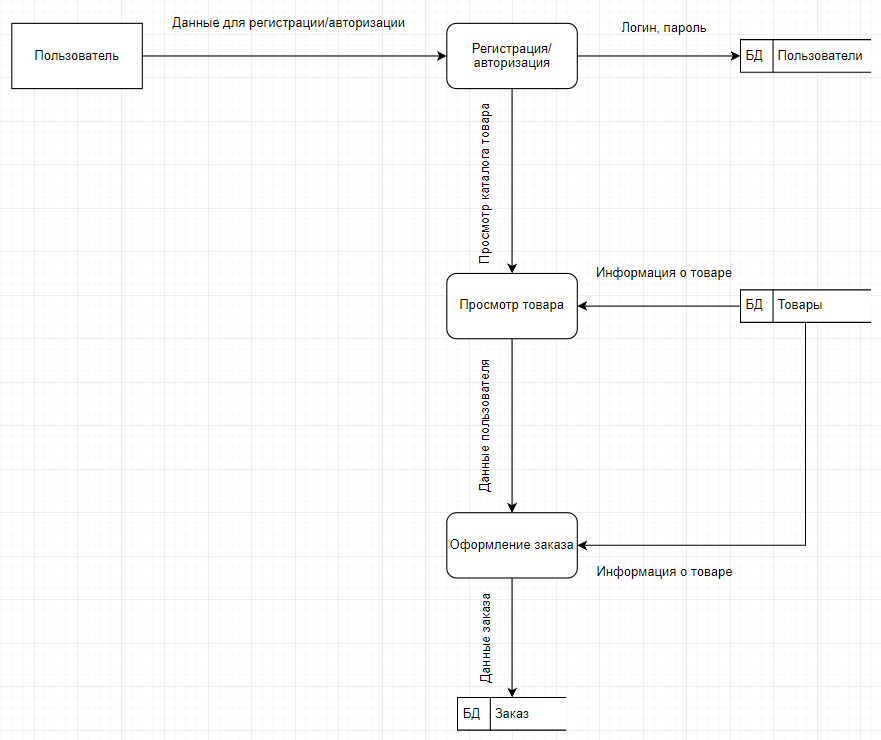
****

Рисунок 9 – Диаграмма потоков данных DFD

# **Проектирование базы данных**

Проектирование базы данных начинается с концептуального проектирование базы данных.

Концептуальное проектирование – построение семантической модели предметной области, то есть информационной модели наиболее высокого уровня абстракции. Такая модель создаётся без ориентации на какую-либо конкретную СУБД и модель данных.

На рисунке 10 представлена инфологическая модель базы данных.

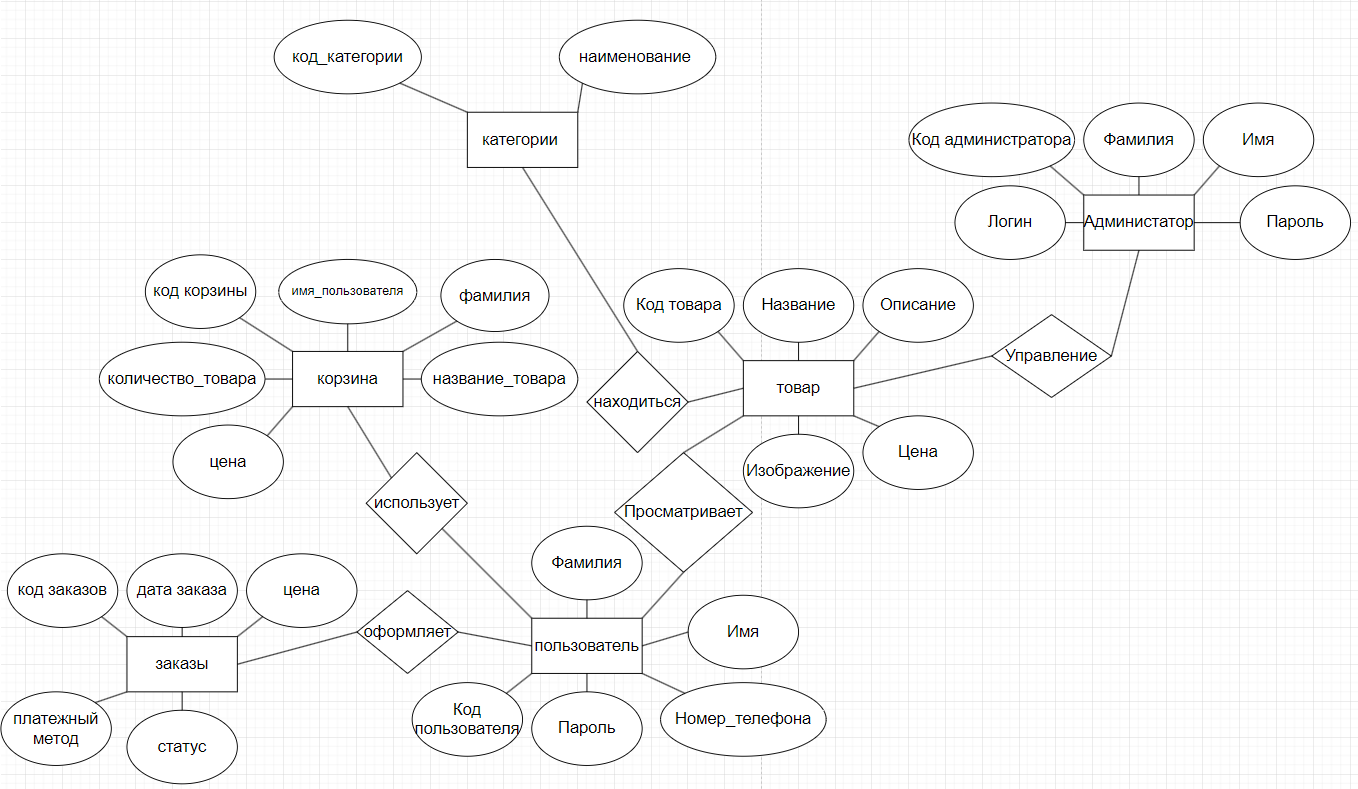
****

Рисунок 10 – Инфологическая модель базы данных

Далее происходит преобразование концептуальной модели в логическую модель, по формальным правилам. Таким образом, логическое (даталогическое) проектирование - создание схемы базы данных на основе конкретной модели данных.

На даталогической модели базы данных (рисунок 11) отображены сущности веб-приложения, а также первичные и внешние ключи, связывающие сущности между собой.

На рисунке 11 представлена ER-диаграмма базы данных. Она содержит 6 таблиц для полного функционирования и качественной сортировки информации.

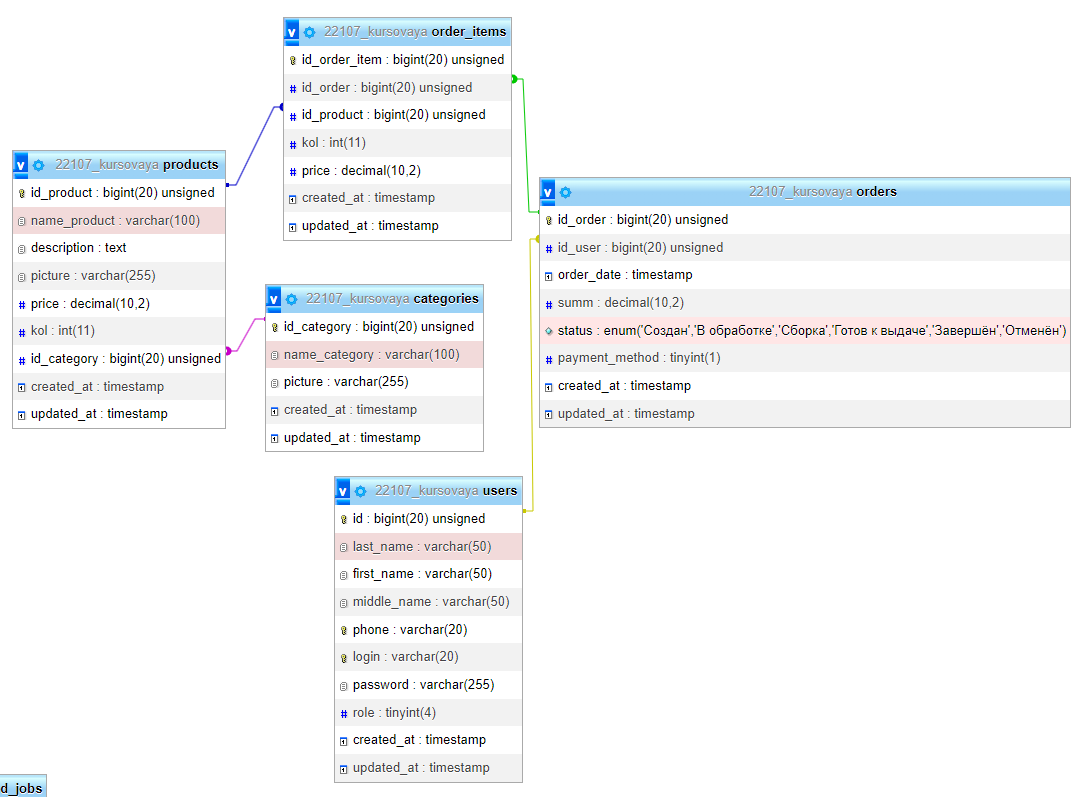


Рисунок 11 – ER-модель базы данных

Перечень таблиц представлены в таблице 5. Подробное содержание таблиц (таблица 6–10).

Таблица 5 – Таблицы ER-модели

|  |  |
| --- | --- |
| Таблица | Описание |
| user | Таблица пользователей |
| categories | Таблица категории |
| products | Таблица товаров добавленных пользователем в корзину |
| orders | Таблица заказов |
| order\_items | Таблица деталей заказа |

Таблица 6 – Таблица «users»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id\_user (PK) | INT | Id пользователей |
| last\_name | VARCHAR(50) | Имя пользователя |
| first\_name | VARCHAR(50) | Фамилия пользователя |
| middle\_name | VARCHAR(50) | Отчество пользователя |
| phone | VARCHAR(20) | Телефон |
| password | VARCHAR(20) | Пароль |
| role | INT | 0 – пользователь  1 – администратор |
| created\_at | TIMESTAMP | Добавление |
| updated\_at | TIMESTAMP | Обновление |

Таблица 7 – Таблица «categories»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id\_category (PK) | INT | Id категорий |
| name\_category | VARCHAR(100) | Название категории |
| picture | VARCHAR(255) | Фото категории |
| created\_at | TIMESTAMP | Добавление |
| updated\_at | TIMESTAMP | Обновление |

Таблица 8 – Таблица «products»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id\_product (PK) | INT | Id товаров |
| name\_product | VARCHAR(100) | Название товара |
| description | TEXT | Описание |
| picture | VARCHAR(255) | Изображение |
| price | DECIMAL(10, 2) | Цена |
| kol | INT | Количество |
| created\_at | TIMESTAMP | Добавление |
| updated\_at | TIMESTAMP | Обновление |
| id\_category (FK) | INT | Id категории |

Таблица 9 – Таблица «orders»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id\_order (PK) | INT | Id заказов |
| order\_date | TIMESTAMP DEFAULT CURRENT\_TIMESTAMP | Дата заказа |
| summ | DECIMAL(10, 2) | Цена |
| status | ENUM('В обработке', 'Создан', 'Сборка', 'Готов к выдаче' , 'Завершён', 'Отменён') | Статус заказа |

Продолжен те таблица 9 – Таблица «orders»

|  |  |  |
| --- | --- | --- |
| payment\_method | TINYINT(1) | Платежный метод 0: наличные, 1: карта |
| created\_at | TIMESTAMP | Добавление |
| updated\_at | TIMESTAMP | Обновление |
| id\_user (FK) | INT | Id пользователей |

Таблица 10 – Таблица «order\_items»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id\_order\_item (PK) | INT | Id деталей заказов |
| kol | INT | Количество |
| price | DECIMAL(10, 2) | Цена |
| created\_at | TIMESTAMP | Добавление |
| updated\_at | TIMESTAMP | Обновление |
| id\_order (FK) | INT | Id заказов |
| id\_product (FK) | INT | Id товаров |

# **Проектирование интерфейса**

Для разработки пользовательского интерфейса был выбран инструмент draw.io – браузерный инструмент для создания диаграмм, блок-схем и прочего.

В результате проектирование интерфейса будущей веб-приложением были спроектированы прототипы семи страниц.

На рисунке 12 представлен прототип «Регистрация», где видно, что пользователь будет вводить при регистрации своего аккаунта.

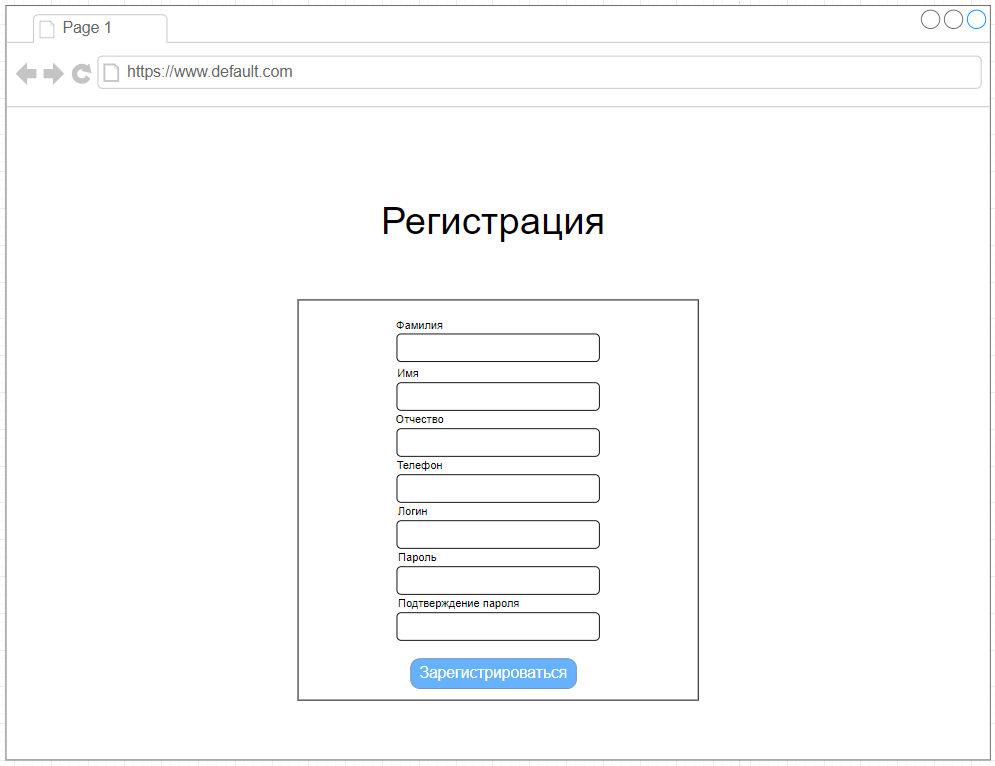


Рисунок 12 – Регистрация пользователя

На рисунке 13 представлен прототип страницы "Главная", который демонстрирует, как будет выглядеть пользовательский интерфейс для регистрации пользователя в веб-приложении "Магазин компьютерной периферии".

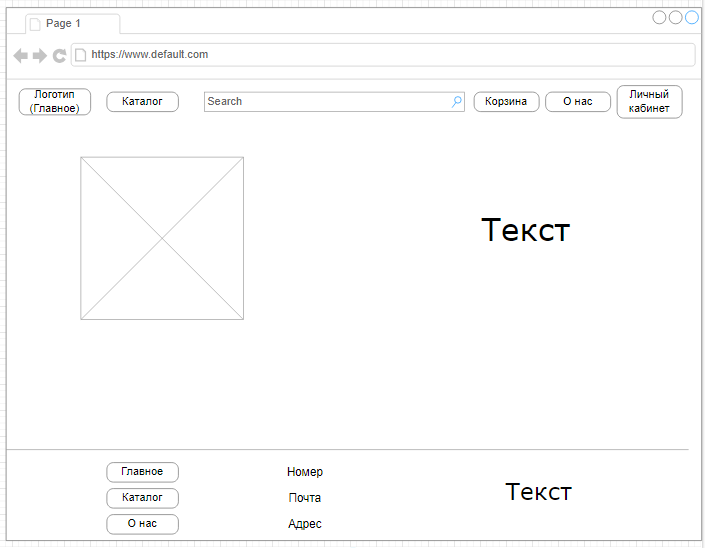


Рисунок 13 – Главная страница

На рисунке 14 представлен прототип «Каталога», показывающий как, будет выглядит страница.

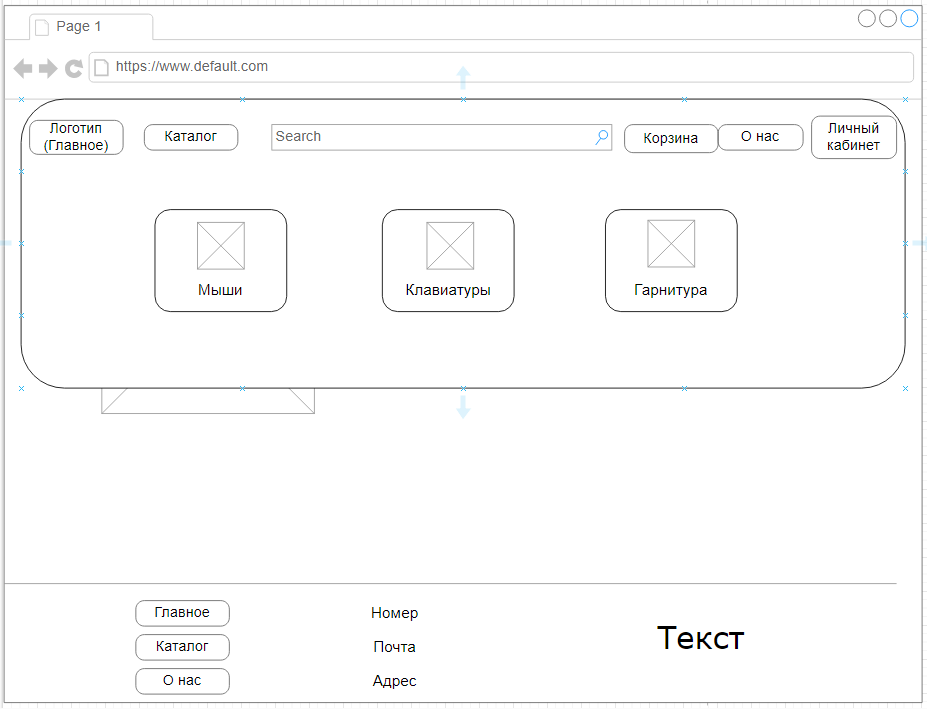


Рисунок 14 – Каталог товаров

На рисунке 15 представлен прототип «Карточкой товара», показывающий как, будет выглядит страница.

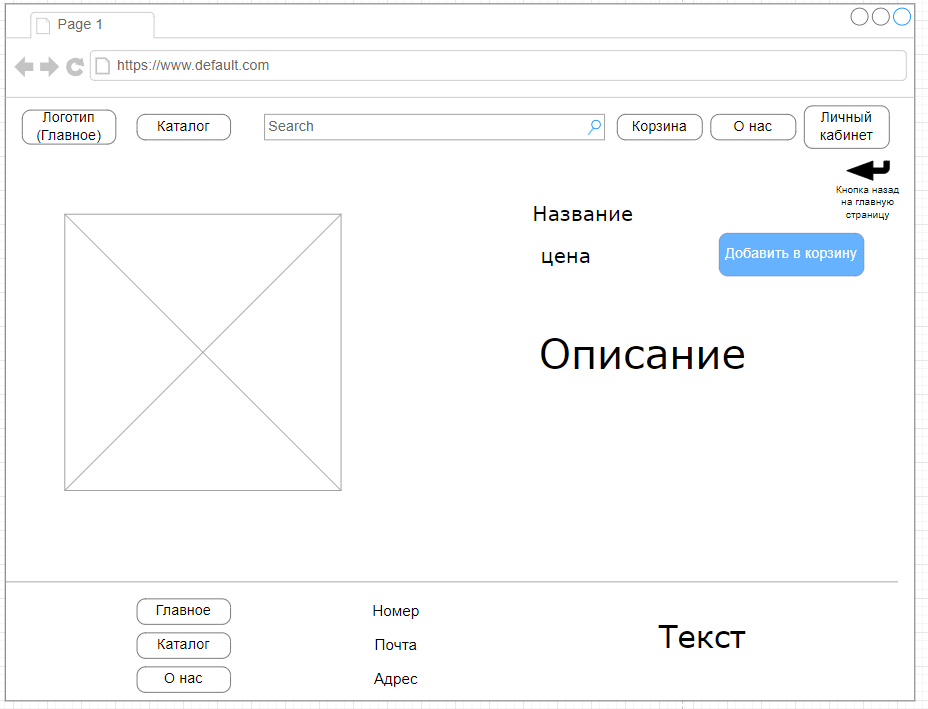


Рисунок 15 – Карточка товара.

На рисунке 16 представлен прототип «Корзины с товаром», показывающий как, будет выглядит страница.

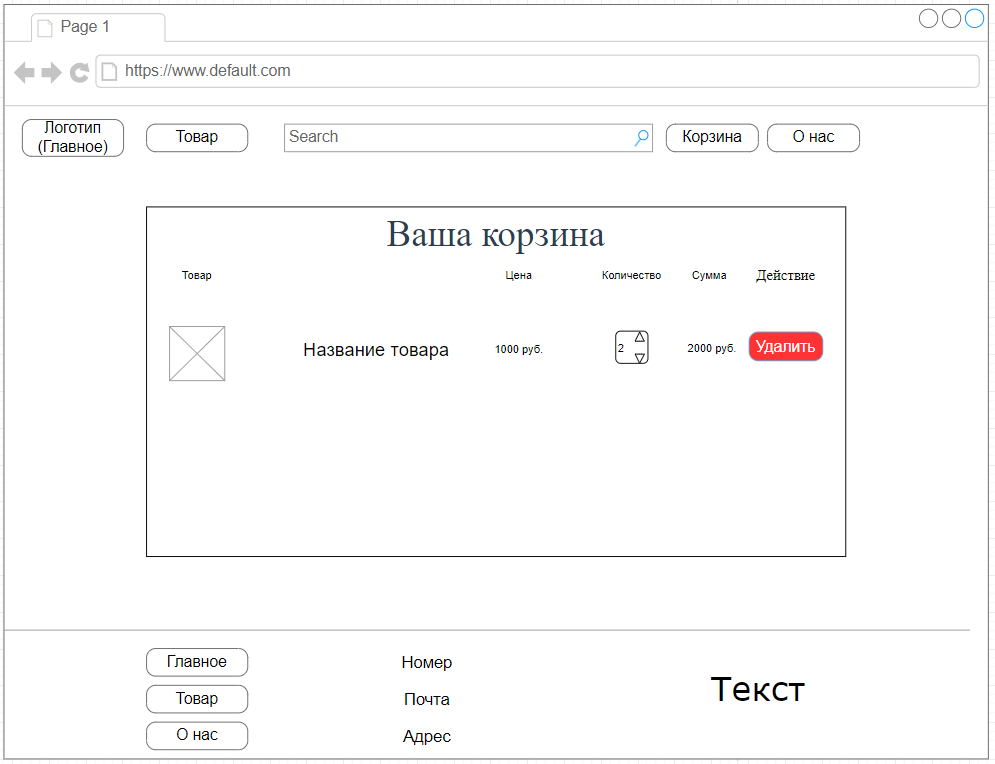


Рисунок 16 – Корзина с товаром

На рисунке 17 представлен прототип «Мои заказы», показывающий как, будет выглядит страница.

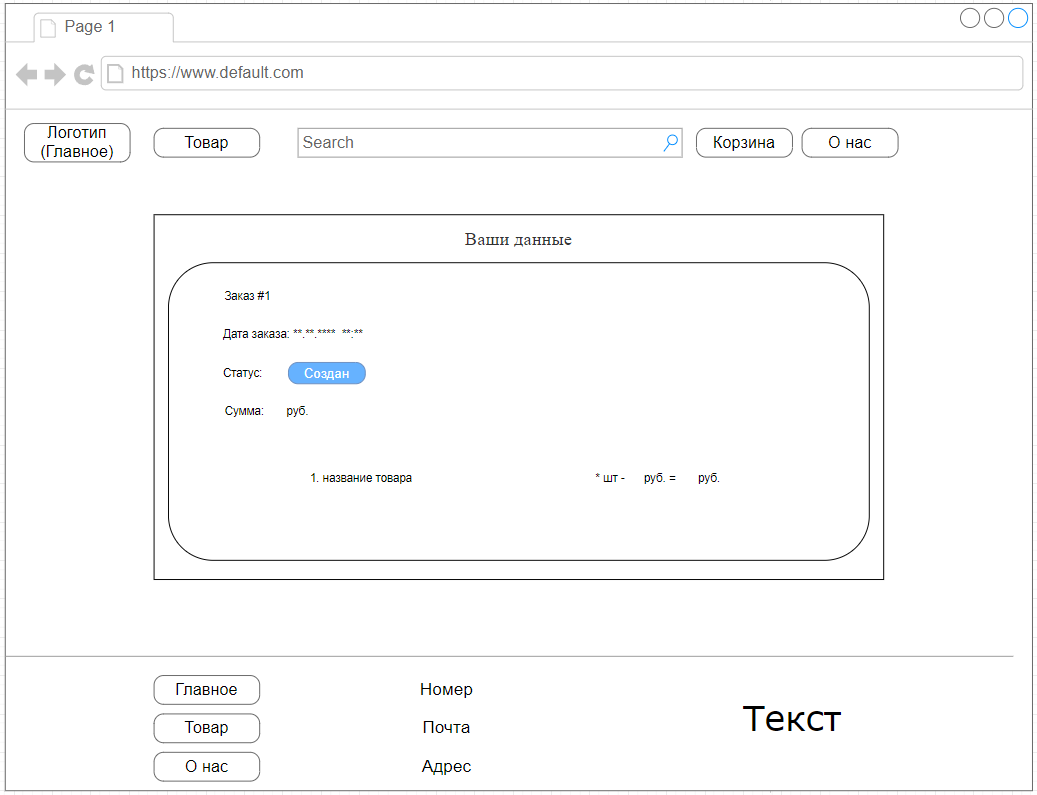


Рисунок 17 – Мои заказы

На рисунке 18 представлен прототип «Личный кабинет», показывающий как, будет выглядит страница.

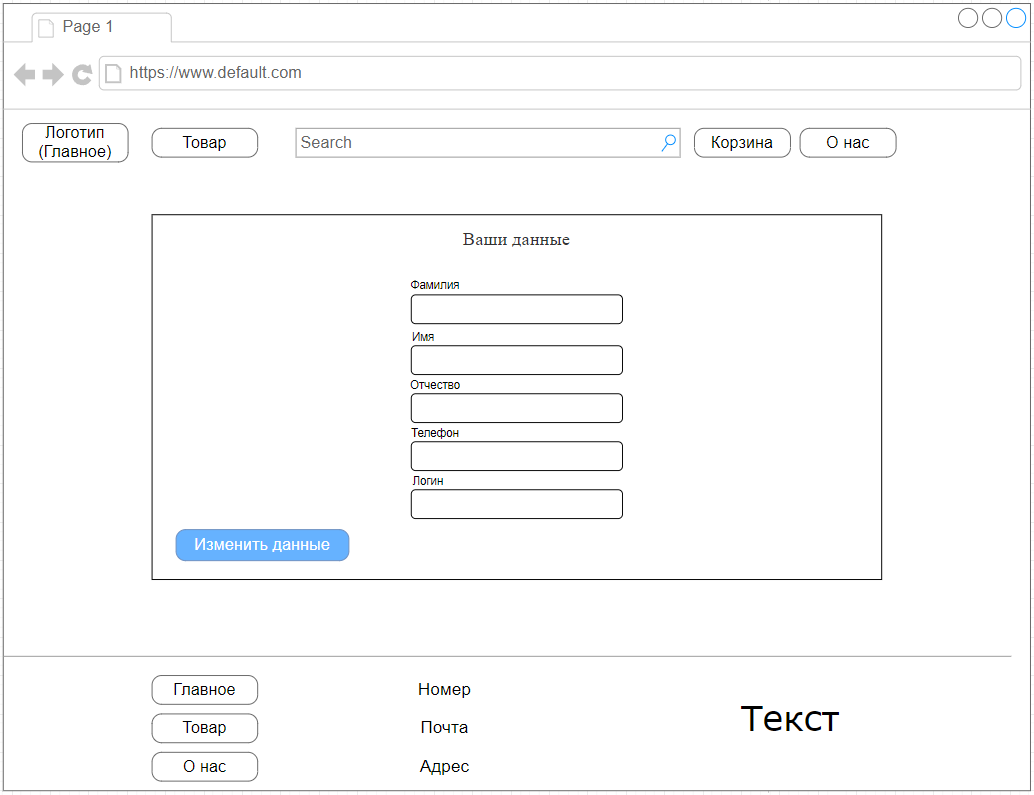


Рисунок 18 – Личный кабинет (модальное окно)

# **Разработка ВЕБ**

# **Разработка интерфейса ВЕБ**

В разрабатываемом веб-приложении для магазина компьютерной периферии будет использоваться формат \*.php, что подразумевает использование языка программирования PHP. Для упрощения разработки и повышения эффективности был выбран фреймворк Laravel, который предоставляет мощный набор инструментов для создания современных веб-приложений.

Для реализации клиентской части веб-приложения, отвечающей за внешний вид и пользовательский интерфейс, был использован CSS позволяет создавать визуально привлекательные и удобные интерфейсы, управляя стилями элементов HTML, таких как формы, кнопки, текстовые поля и другие компоненты. На рисунке 19 представлен код главной страницы.

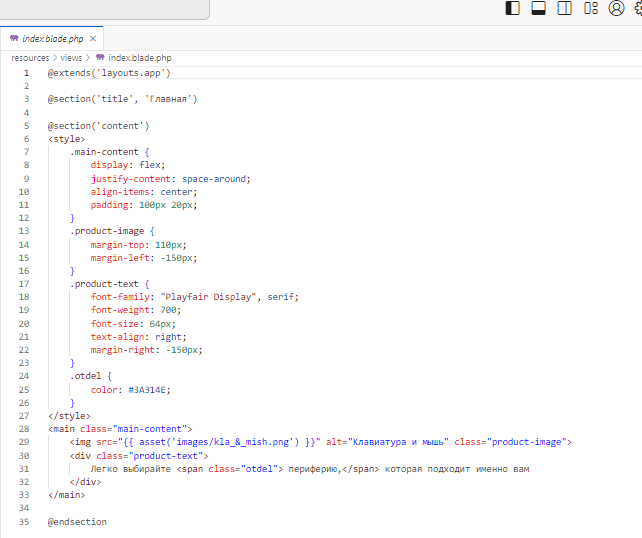


Рисунок 19 – Код главной страницы

На рисунке 20 показана главная страница то, как она выглядит со стилями фреймворка.

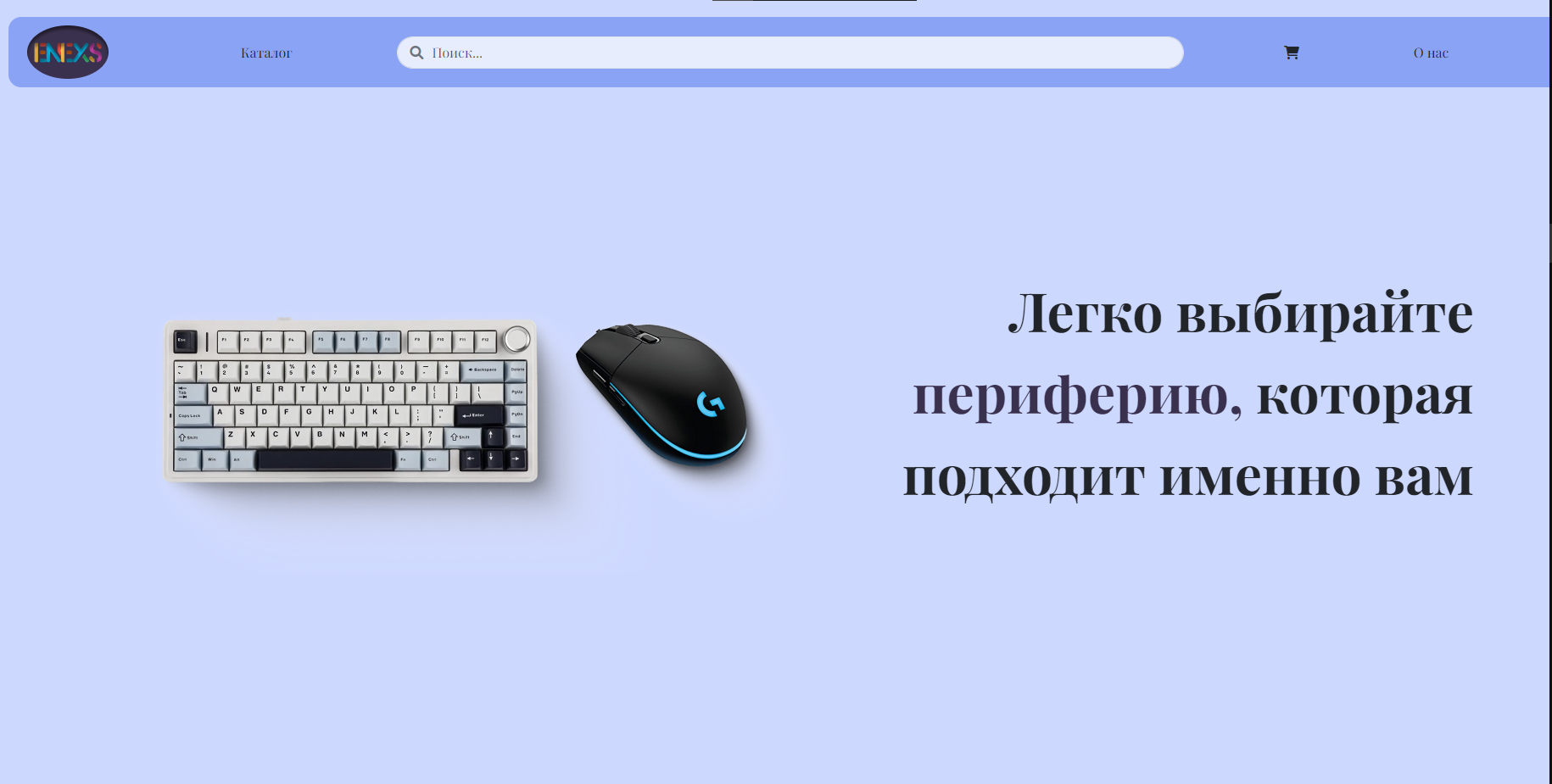


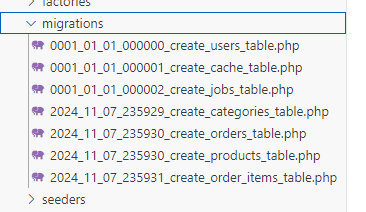
Рисунок 20 – Главная страница

* 1. Разработка базы данных ВЕБ

В качестве СУБД для реализации Веб-приложения «Магазин компьютерной периферии» была выбрана MySQL по ряду причин, описанные в разделе «Анализ инструментальных средств разработки».

База данных содержит 5 таблиц (ER-модель на рисунке 11).

Таблицы в проекте были созданы и настроены с использованием миграций в фреймворке PHP – Laravel. Все файлы миграций хранятся в папке migrations, представлено на рисунке 21.

****Рисунок 21 – Файлы миграций

На рисунке 22 представлена миграция таблицы «users».



Рисунок 22 – Миграция таблицы «users»

На рисунке 23 представлена миграция таблицы «categories».



Рисунок 23 – Миграция таблицы «categories»

На рисунке 24 представлена миграция таблицы «orders».



Рисунок 24 – Миграция таблицы «orders»

На рисунке 25 представлена миграция таблицы «products».



Рисунок 25 – Миграция таблицы «products»

На рисунке 26 представлена миграция таблицы «order\_items».



Рисунок 26 – Миграция таблицы «order\_items»

# **Разработка ВЕБ**

Подключение к базе данных и вывод таблицы осуществляется с помощью env файла (рисунок 27).

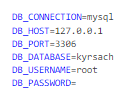


Рисунок 27 – env файла

Первым шагом в процессе разработки веб-приложения «Магазин компьютерной периферии» является создание модели данных. В фреймворке Laravel это делается с помощью команды Artisan make:model. Эта команда генерирует файл модели, который представляет собой класс, описывающий структуру данных и взаимосвязи с другими таблицами в базе данных.

На рисунке 28 изображена модель, которая представляет собой структуру данных, используемую для работы с товарами (Product). В данной модели также указаны связи с таблицами категорий (Category) и элементов заказов (OrderItem). Эти связи позволяют организовать взаимодействие данных между различными сущностями в базе данных, что является ключевым аспектом для обеспечения целостности и логической структуры данных.



Рисунок 28 – Модель продукта

Контроллеры позволяют сгруппировать связанную логику в одном месте, что упрощает управление и поддержку кода.

На рисунке 29 контроллер представляет собой класс, который отвечает за обработку запросов, связанных с товарами и категориями. Он демонстрирует функционал для отображения всех категорий, товаров по категориям, а также детальной информации о конкретном товаре. Этот подход улучшает читаемость и организацию кода, делая его более модульным и легко расширяемым. Весь код контроллера продаж представлен в приложении Б.



Рисунок 29 – Контроллер товаров

На рисунке 30 представлен контроллер категорий товаров.

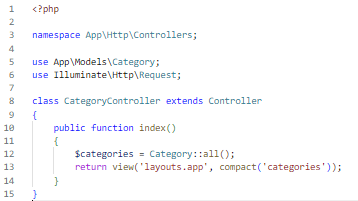


Рисунок 30 – Контроллер категорий товаров

Для корректной работы функций, реализованных в контроллерах, на страницах системы необходимо настроить маршруты в файле маршрутов, как это показано на рисунке 31. В начале файла маршрутов следует указать подключение соответствующих контроллеров.



Рисунок 31 – Файл маршрутов

# **Документирование программного продукта**

# **Руководство пользователя ВЕБ**

Для запуска веб-приложения в терминале выполните команду php artisan serve. Эта команда запустит локальный сервер Artisan, который будет доступен по адресу http://localhost:8000. Сервер Artisan автоматически загружает маршруты, контроллеры, модели и представления, что значительно упрощает процесс разработки и тестирования. Такой подход позволяет быстро проверять изменения в коде без необходимости настройки сложных серверных конфигураций.

После ввода адреса http://localhost:8000 в браузере, отобразится главная страница вашего веб-приложения (рисунок 32). Это позволяет быстро оценить внешний вид и функциональность интерфейса, а также протестировать взаимодействие с различными элементами системы. Если вы вносите изменения в код, сервер автоматически перезагружается, чтобы отразить эти изменения, что делает процесс разработки более удобным и эффективным.

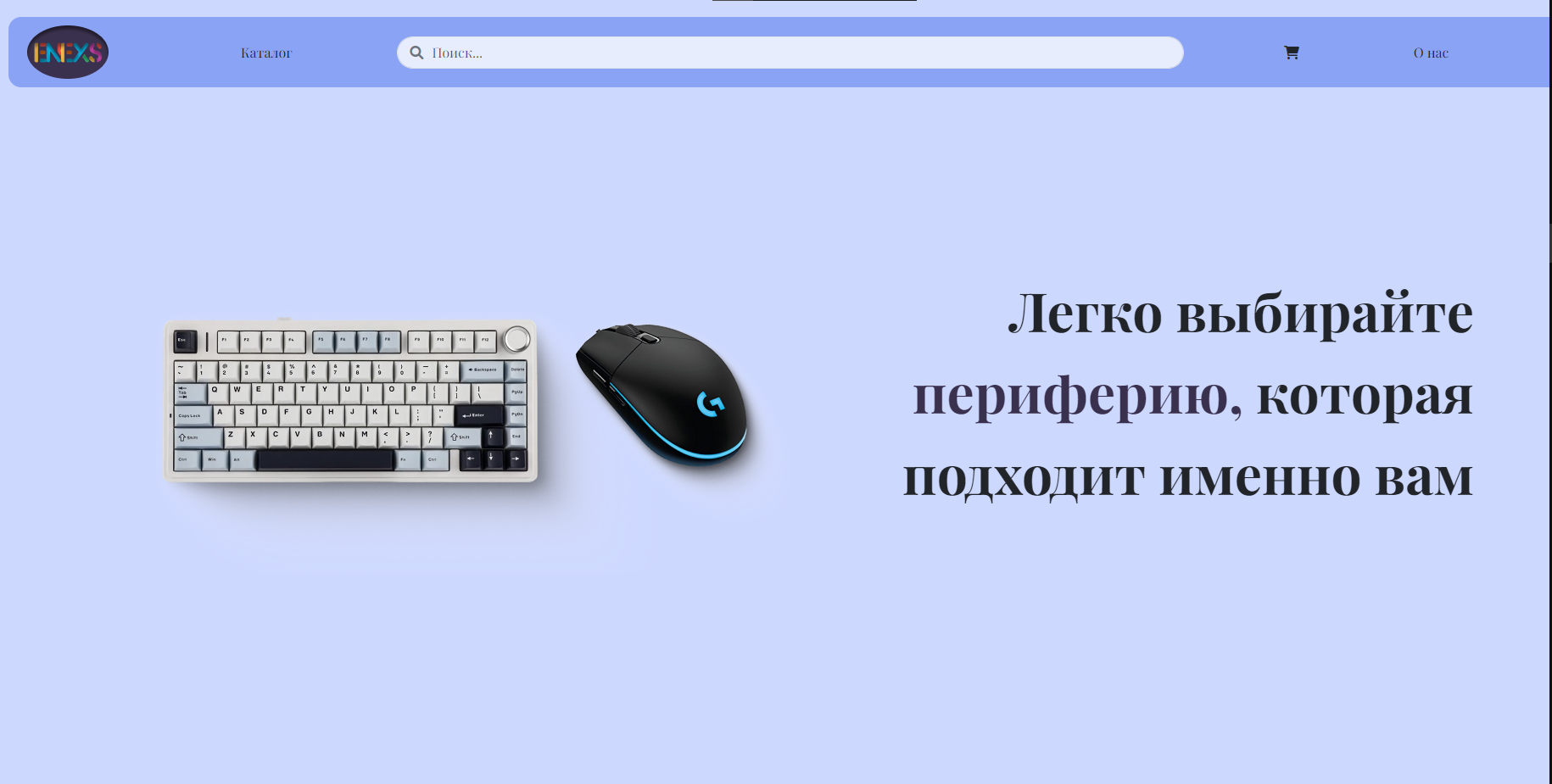


Рисунок 32 – Главная страница

Чтобы пользователь имел возможность добавлять в корзину, то нужно авторизоваться. Страница авторизации представлена на рисунке 33.

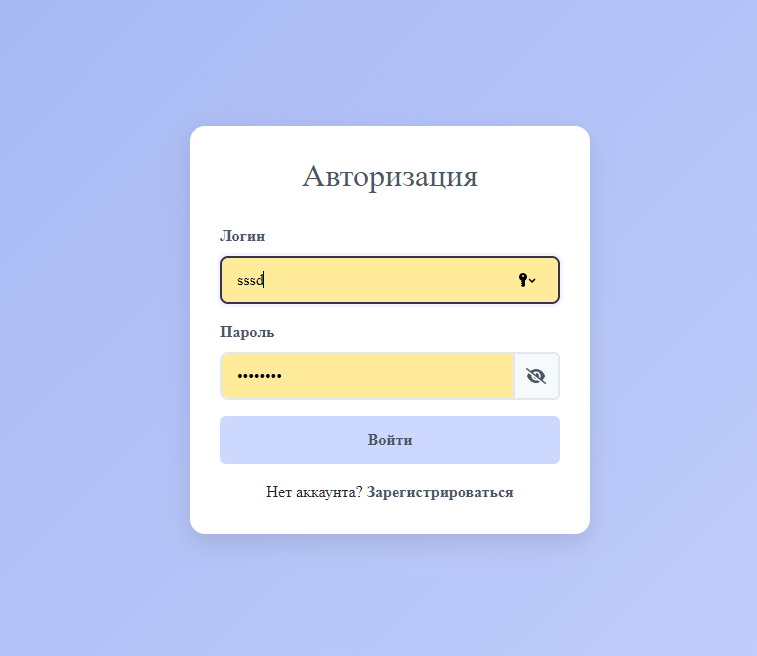


Рисунок 33 – Окно авторизации

Если у пользователя нет аккаунта, то ему нужно перейти на страницу регистрации (Рисунок 34).

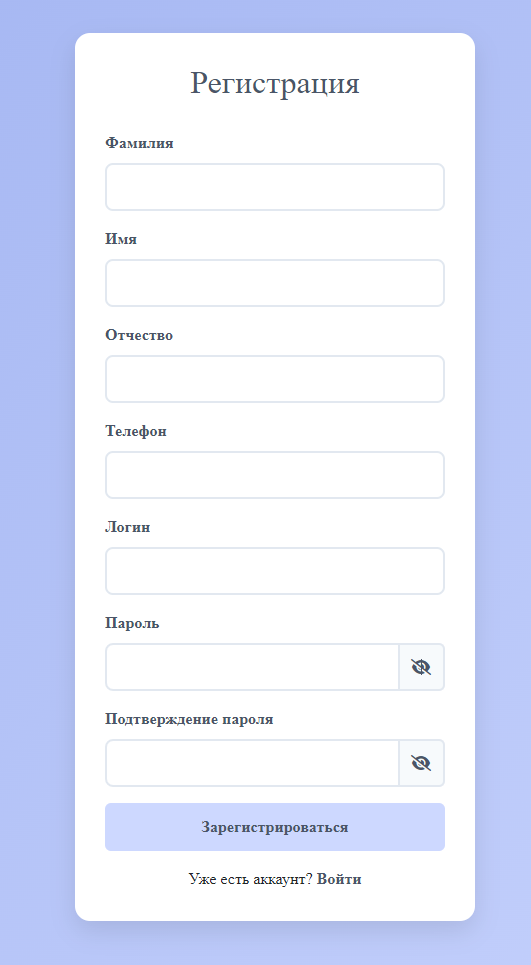


Рисунок 34 – Страница регистрации

После того, как пользователь зашёл в систему, то ему открывается возможность добавлять товары в корзину (рисунок 35). На странице корзины пользователь может добавить количество товара, удалять добавленный товар и оформлять заказ.

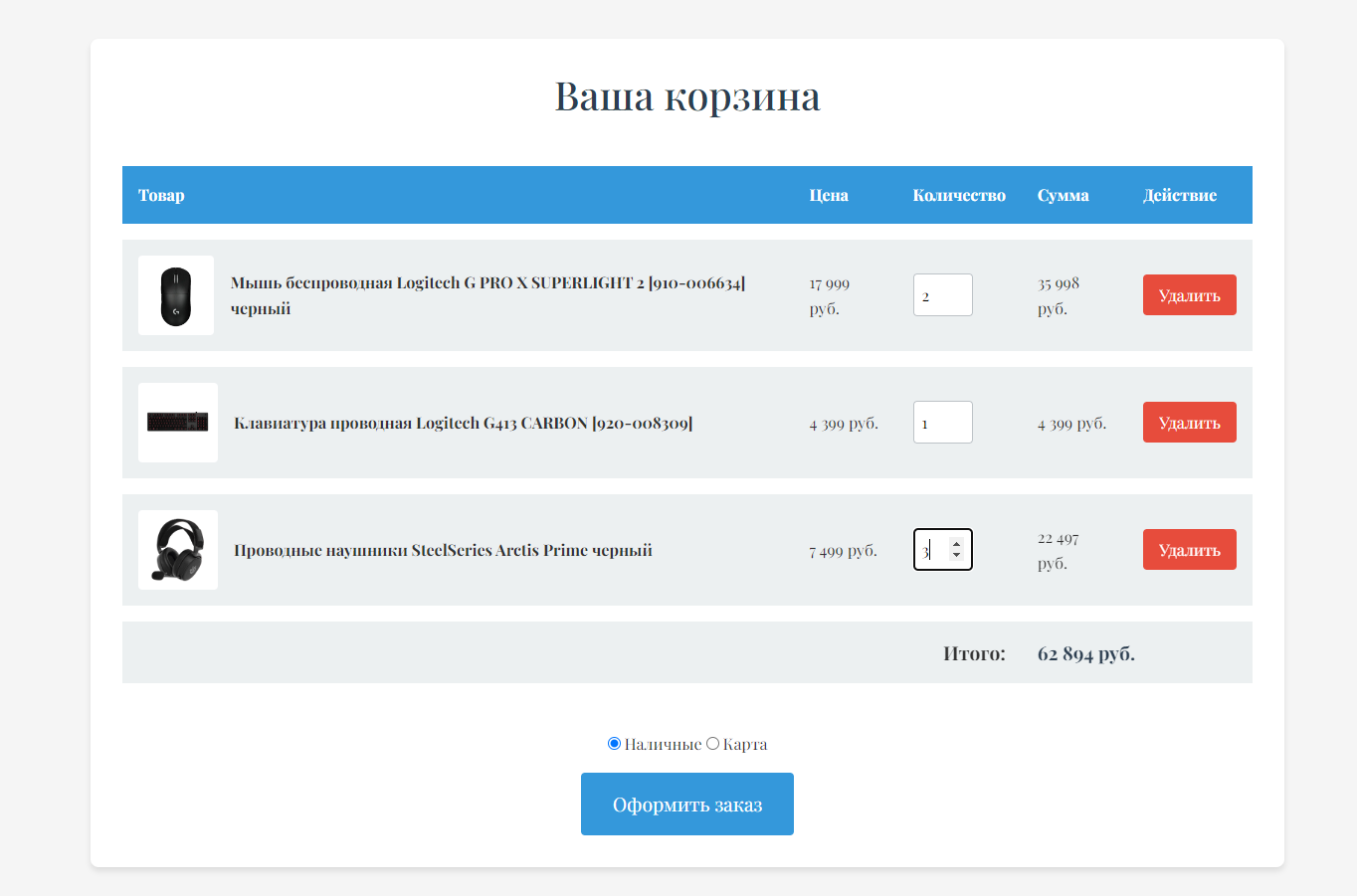


Рисунок 35 – Страница корзины

Если пользователь нажмет на кнопку «Оформить заказ», то его перенаправит на страницу с историей его заказов (рисунок 36).



Рисунок 36 – Страница мои заказы

Так же пользователь может редактировать свои данные профиля. Страница профиля показана на рисунке 37.



Рисунок 37 – Страница мои данные

Также пользователь может просмотреть весь каталог товаров (рисунок 38). Страница товаров представлена на рисунке 39. Эти товары можно добавить в корзину и просмотреть карточку товара (рисунок 40).

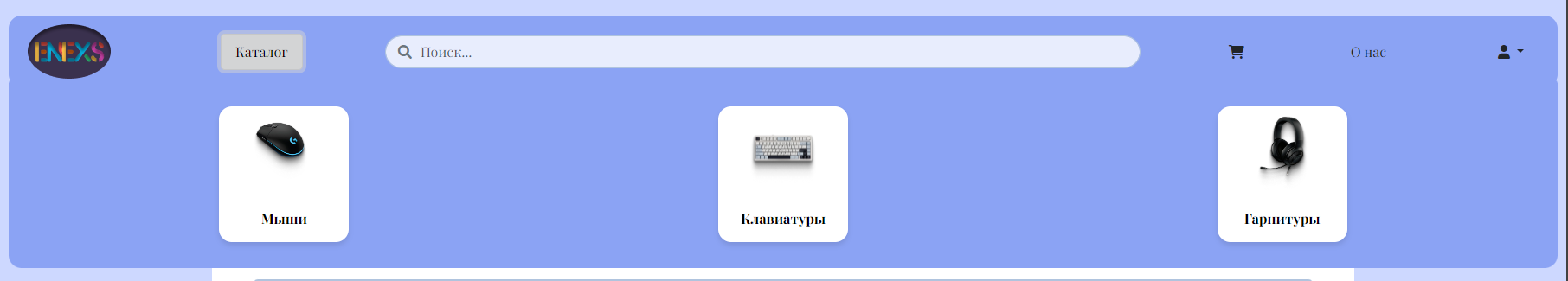


Рисунок 38 – Страница каталога

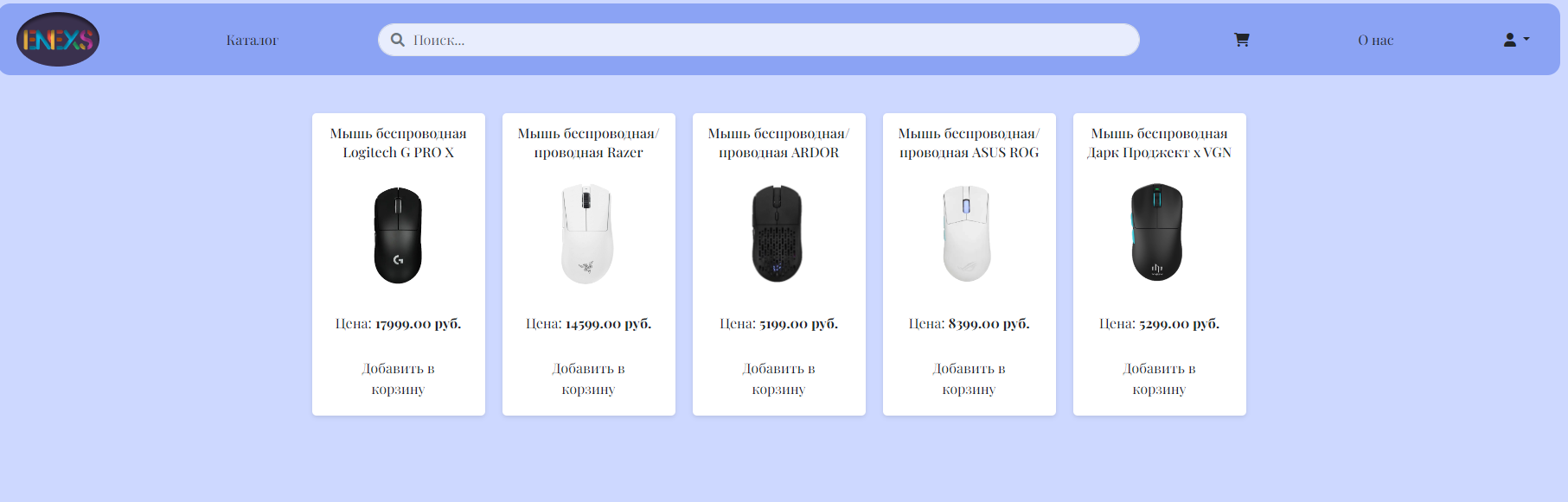


Рисунок 39 – Страница представленных товаров

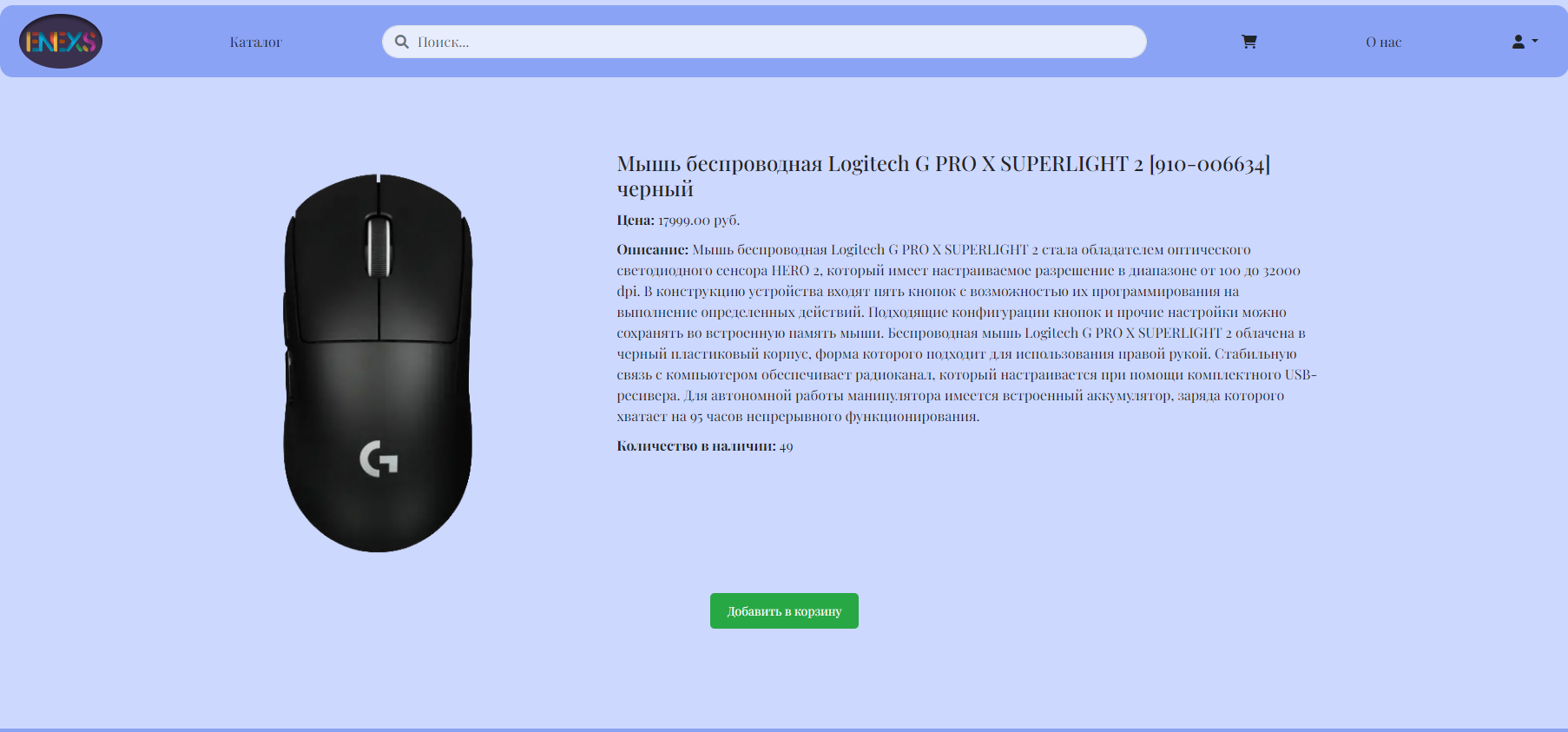


Рисунок 40 – Страница карточка товара

# **Заключение**

В ходе выполнения курсовой работы была достигнута цель разработки веб-приложения «Магазин компьютерной периферии». В рамках работы были решены все поставленные задачи:

* Описание предметной области.
* Анализ инструментальных средств разработки.
* Техническое задание.
* Проектирование веб-приложение.
* Разработка веб-приложение.
* Документирование программного продукта.

Также реализован следующий функционал:

* Администратор может удалять, редактировать и создавать новый товар и категории на странице через специальную панель администратора.
* Регистрация пользователей в веб-приложении.
* Пользователь может просматривать сайта и товаров.
* Создание поисковой строки.

Был проанализирован и разработан программный продукт с интуитивным пользовательским интерфейсом.

В процессе создания курсового проекта, были получены и применены новые знания. Основная сложность, с которыми пришлось столкнуться во время разработки, отсутствие опыта работы с laravel. Проблемы были решены путём подробного изучения принципов работы laravel.

В дальнейшем ВЕБ можно сделать более функциональным.

# **Список используемых источников**

1. laravel.su – документация по Laravel – URL: https://laravel.su/docs/11.x/installation (дата обращения 11.10.2024). – Текст: электронный.
2. laravel.su – frontend – URL: https://laravel.su/docs/11.x/frontend (дата обращения 11.10.2024). – Текст электронный.
3. sinonim.org ­­– Онлайн словарь помогает быстро, удобно и бесплатно искать определения нужных слов – URL: https://sinonim.org/t (22.09.2024) – Текст: электронный.
4. fontawesome – это библиотека иконок и набор инструментов используемый миллионами дизайнеров, разработчиков – URL: https://fontawesome.com (15.11.2024) – Текст: электронный.
5. htmlacademy.ru – Понятие и особенности языка PHP. – URL: https://htmlacademy.ru (дата обращения: 08.11.2024). – Текст: электронный.
6. htmlacademy.ru – Определение и особенности языка PHP – URL: https://htmlacademy.ru/tutorial/php/basics. – Определение и особенности языка PHP. – (Дата обращения 09.11.2024). – Текст: электронный.

# **Приложение А – Техническое задание**

**Министерство образования Иркутской области**

Государственное бюджетное профессиональное

образовательное учреждение Иркутской области

«Иркутский авиационный техникум»

(ГБПОУИО «ИАТ»)

**Техническое задание**

**МАГАЗИН КОМПЬЮТЕРНОЙ ПЕРИФЕРИИ**

|  |  |  |
| --- | --- | --- |
| Руководитель: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | (Н.Р. Карпова) |
|  | (подпись, дата) |  |
| Студент: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | (Е.Ю. Лиханова) |
|  | (подпись, дата) |  |

Иркутск 2024

1. **Введение**
   1. **Общие сведения**

Документ представляет собой техническое задание на создание веб-приложения «Магазин компьютерной периферии.

* 1. **Цели и задачи**

Целью создания веб-приложения «Магазин компьютерной периферии» является автоматизация работы магазина. Задачи веб-приложения включают:

* Регистрация и авторизация пользователя.
* Добавление и удаление товара в корзине.
* Оформления заказа.
* Авторизация администратора.
* Создание админ панели, где администратор может удалять, добавлять и редактировать товар, так же добавлять и удалять каталог.

1. **Основания для разработки**
   1. **Нормативные документы**

Документ основывается на следующих нормативных документах:

* ГОСТ 34.602-2020 "Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы".
* ГОСТ Р 56477-2015 "Проектирование и внедрение информационных систем. Общие требования".
* ГОСТ Р 51304-2009 "Услуги торговли. Общие требования".
* ГОСТ Р 52292-2004 "Информационная технология. Электронный обмен информацией. Термины и определения"
* ГОСТ Р 51303-2013 "Торговля. Термины и определения".
  1. **Проектные документы**

Проектные документы включают:

* Пояснительную записку.
* Руководство и эксплуатация.
* Техническое задание.

1. **Назначение веб-приложения**
   1. **Общее описание**

Автоматизация веб-приложения магазин компьютерной периферии предназначена для упрощения процесса покупок.

* 1. **Преимущества и новизна**

Автоматизация веб-приложения магазин компьютерной периферии будет предоставлять:

* Преимущества:

Интуитивно понятный интерфейс.

* Новизна:

В любое время будет возможность заказать товар.

24/7 работы пункта выдачи.

1. **Требования к системе веб-приложения**
   1. **Функциональные требования**

* Управление товарами:

Ведение каталога товаров с возможностью редактирования, удаления и добавления товаров.

Регистрация и авторизация клиента:

Регистрация нового клиента.

Авторизация клиента в веб-приложении.

* Корзина:

Добавление товаров в корзину с возможностью редактирования.

Вывод общей суммы.

Оформления заказа.

* Каталог товаров:

Поиск товара.

Просмотр товаров.

* Оформление заказа:

Оплата наличными или картой.

* 1. **Технические требования**
* Производительность:

Обработка до 3 задач одновременно.

Время отклика системы не более 3 секунд при загрузке данных.

* Надежность:

Доступность системы не менее 99,5% в год.

Раз в 2 дня резервное копирование данных.

* Безопасность:

Аутентификация пользователей через логин и пароль.

Различным пользователям веб-приложения должны предоставляться различные роли.

* 1. **Эксплуатационные требования**
* Удобство использования:

Дружественный пользовательский интерфейс.

Возможность быстрого оформления заказа.

1. **Требования к техническому обеспечению**
   1. **Оборудование**

* Сервер: Серверная платформа с процессором не менее 4 ядер, 16 ГБ ОЗУ, SSD объемом 256 ГБ.
* Клиентские рабочие станции: ПК с ОС Windows 10 или Linux, 4 ГБ ОЗУ, 2 ГБ свободного места на диске.
  1. **Сетевые требования**
* Сеть: Доступ в Интернет со скоростью не менее 50 Мбит/с
* Сетевые протоколы: Поддержка TCP/IP, HTTP/HTTPS.

1. **Требования к программному обеспечению**
   1. **Программные компоненты**

* Операционная система: Серверная версия ОС Linux или Windows Server.
* Базы данных: MySQL.
* Программное обеспечение: Visual Studio Code, веб-сервер Apache или Nginx, интерпретатор PHP.
  1. **Интерфейсы**
* Интерфейс пользователя: Веб-интерфейс с поддержкой браузеров Chrome, Yandex.

1. **Организационно-технические требования**
   1. **Этапы разработки**

В таблице 12 представлены сроки и этапы разработки системы.

Таблица 12 – Сроки и этапы разработки веб-приложения

|  |  |  |
| --- | --- | --- |
| № | Этап | Срок выполнения |
| 1 | Пред проектное исследование предметной области (выбор темы, постановка цели, задач, описание области применения, исследование предметной области) | До 18.09.24 |
| 2 | Разработка технического задания (выбор архитектуры программного обеспечения, выбор типа пользовательского интерфейса, выбор языка и среды программирования) | До 23.09.24 |
| 3 | Проектирование программного обеспечения.  (разработка структурной и функциональной схемы ПО, проектирование базы данных (инфологическое, ER-модель, физическая модель) | До 25.10.24 |
| 4 | Разработка (программирование) и отладка программного продукта | До 14.12.24 |
| 5 | Составление программной документации (оформление ПЗ, написание руководства пользователя, составление презентации и речи) | До 16.12.24 |

# **Приложение Б – Листинг кода контроллера товара**

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use App\Models\Product;

use App\Models\Category;

class ProductController extends Controller

{

//Метод, который обрабатывает запрос для отображения всех категорий продуктов.

public function index()

{

// Получение всех категорий из базы данных.

$categories = Category::all();

return view('shop.product', compact('categories'));

}

// Метод, который показывает продукты в определенной категории по ее идентификатору.

public function showCategory($id)

{

// Проверка, существует ли категория

$category = Category::find($id);

if (!$category)

{

abort(404, 'Категория не найдена.');

}

// Получение продуктов для конкретной категории с пагинацией

$products = Product::where('id\_category', $id)->paginate(10);

// Возвращает представление 'shop.product-category', передавая название категории и список продуктов.

return view('shop.product-category', [

'categoryName' => $category->name\_category,

'products' => $products,

]);

}

// Метод для отображения деталей конкретного продукта по его идентификатору.

public function showProduct($id)

{

$product = Product::findOrFail($id);

// Возвращает представление 'shop.product-detail', передавая информацию о продукте.

return view('shop.product-detail', compact('product'));

}

}