

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(ДГТУ)**

Факультет «Информатика и вычислительная техника»

КАФЕДРА «Кибербезопасность информационных систем»

**КУРСОВАЯ РАБОТА**

Тема: «Разработка WEB-приложения для продажи компьютерных девайсов»

Дисциплина: «WEB-программирование»

Направление подготовки: 09.03.01 Информатика и вычислительная техника

Направленность: Информатика и вычислительная техника

Обозначение курсового проекта WП.490000.000 Группа ВИ-43

Обучающийся \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Думановский Т.С.

(подпись, дата)

Курсовая работа защищена с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ доцент, И.А. Сосновский

(подпись, дата)

Ростов-на-Дону

2024



МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(ДГТУ)**

Факультет «Информатика и вычислительная техника»

КАФЕДРА «Кибербезопасность информационных систем»

**ЗАДАНИЕ**

на выполнение курсовой работы

Тема «Разработка WEB-приложения для продажи компьютерных девайсов»

Дисциплина: WEB-программирование

Обучающийся: Думановский Тихон Сергеевич

Обозначение курсовой работы WП.490000.000 Группа: ВИ-43

Срок представления проекта (работы) к защите «\_\_\_» \_\_\_\_\_\_\_\_ 20\_\_г.

Исходные данные для курсового проекта:

1. Учебные материалы, электронные ресурсы, периодические материалы.

2. Официальный сайт организации, для которой разрабатывается web-приложение, сайты аналогичных компаний.

3. Рекомендации по выполнению работы.

4. Рекомендации по оформлению работы.

**Содержание пояснительной записки**

|  |
| --- |
| Введение. |
| 1. Теоретические основы разработки сайтов. |
| 2. Сбор информации об объекте. |
| 3. Создание базы данных. |
| 3. Создание карты сайта или макета сайта. |
| 4. Создание шаблон дизайна страницы. |
| 5. Формирование контента. |
| 6. Вёрстка сайта. |
| 7. Тестирование и размещение сайта в сети. |
| Заключение. |
| Перечень использованных информационных ресурсов. |
| Приложение А. |

Руководитель проекта \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ И.А. Сосновский

(подпись, дата)

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Т.П. Думановский

(подпись, дата)

Содержание

|  |  |
| --- | --- |
| Введение……………………………………………………………………. | 4 |
| 1. Теоретические основы разработки сайтов…………………………….. | 5 |
| 1.1 Язык гипертекстовой разметки HTML………………………… | 5 |
| 1.2 Язык программирования JavaScript……………………………. | 6 |
| 1.3 Язык программирования PHP…………………………………... | 7 |
| 1.4 Сервер баз данных MySQL……………………………………...  1.5 Среда разработки PHPShtorm…………………………………... | 8  10 |
| 2. Сбор информации об объекте…………………………………………... | 12 |
| 3. Создание базы данных………………………………………………….. | 13 |
| 4. Создание карты сайта или макета сайта………………………………. | 16 |
| 5. Создание шаблон дизайна страницы………………………………….. | 19 |
| 6. Формирование контента………………………………………………… | 21 |
| 7. Вёрстка сайта……………………………………………………………. | 22 |
| 8. Тестирование и размещение сайта в сети …...………………………… | 29 |
| Заключение…………………………………………………………………. | 34 |
| Перечень использованных информационных ресурсов…………………. | 35 |
|  |  |

**Введение**

В современном цифровом бизнесе, где онлайн-торговля и взаимодействие с клиентами становятся все более распространенными, возникает необходимость в эффективных инструментах для организации работы интернет-магазина и обмена информацией с потребителями. В этом контексте веб-приложение для управления продуктами и взаимодействия с клиентами играет ключевую роль, предоставляя пользователям возможность создавать различные страницы товаров, на которых можно просматривать, выбирать и приобретать товары, а также получать обратную связь и делиться мнениями. Позволяя легко добавлять описания товаров, изображения, видеообзоры и другие мультимедийные элементы, веб-приложение становится неотъемлемым инструментом для организации эффективной работы интернет-магазина и удовлетворения потребностей клиентов.

В данном исследовании будут рассмотрены функциональные возможности веб-приложения для управления интернет-магазином и его потенциал в организации эффективного взаимодействия с клиентами и увеличения объемов продаж в различных отраслях розничной торговли.

**1 Теоретические основы разработки сайтов**

* 1. **Язык гипертекстовой разметки HTML**

HTML (HyperText Markup Language) – это стандартизированный язык разметки, применяемый для создания веб-страниц. Он обеспечивает укладку и семантику контента веб-страницы, определяя, как она должна быть организована и представлена в браузере. HTML представляет собой набор элементов (тегов), которые определяют различные части содержимого страницы. Каждый элемент выполняет свою функцию и может включать в себя текст, изображения, ссылки, формы и другие элементы. Элементы HTML структурированы в древовидную систему, известную как DOM (Document Object Model), которая определяет иерархию элементов на странице.

Рассмотрим возможность применения данного языка:

* структурирование содержимого c помощью HTML позволяет организовать содержимое веб-страницы с помощью различных элементов, таких как заголовки, абзацы, списки и таблицы, делая страницу более удобной для чтения и понимания.
* HTML поддерживает создание интерактивных элементов, таких как формы и кнопки, позволяя пользователям взаимодействовать с веб-страницей.
* визуальное форматирование: HTML используется для определения внешнего вида содержимого страницы с помощью CSS (Cascading Style Sheets) и добавления интерактивности с помощью JavaScript.

Вместе с тем, HTML является лишь одной из составляющих технологического стека веб-разработки, и взаимодействует с CSS и JavaScript для обеспечения визуального оформления и интерактивности веб-страниц.

**1.2 Язык программирования JavaScript**

JavaScript (JS) — это высокоуровневый, интерпретируемый язык программирования, который широко применяется для создания динамических веб-страниц. С течением времени JavaScript стал популярным языком программирования, который сейчас используется в различных областях, включая веб-разработку, разработку мобильных приложений, создание игр и многое другое. Основные особенности javascript:

* синтаксис и типы данных – это javascript имеет синтаксис, подобный другим языкам программирования, таким как C++ и Java, что делает его относительно простым для изучения. Он поддерживает различные типы данных, включая числа, строки, массивы, объекты и булевы значения.
* динамическая типизация – javascript не требуется явное указание типов данных переменных, так как он использует динамическую типизацию. это означает, что тип данных переменной определяется автоматически во время выполнения программы.
* функции в javascript поддерживает функции как объекты первого класса, что означает, что функции могут быть присвоены переменным, переданы в качестве аргументов другим функциям и возвращены из других функций.
* объектно-ориентированное программирование (ооп): javascript поддерживает объектно-ориентированное программирование, позволяя создавать объекты, определять их свойства и методы, а также наследовать свойства и методы от других объектов.
* событийное программирование в javascript используется для обработки событий, таких как клики мыши, нажатия клавиш и загрузки страницы. С помощью обработчиков событий можно создавать интерактивные элементы на веб-страницах.
* интеграция с html и css javascript используется для взаимодействия с html и css, позволяя изменять содержимое и стили веб-страницы динамически в зависимости от действий пользователя или других событий.

JavaScript является ключевым инструментом веб-разработки и играет важную роль в создании современных интерактивных и динамических веб-приложений. Он широко применяется как в браузере, так и на сервере.

**1.3 Язык программирования PHP**

PHP (Hypertext Preprocessor) — это широко используемый язык программирования, который применяется преимущественно для создания динамических веб-сайтов и веб-приложений. Он является одним из наиболее популярных языков программирования на вебе и широко применяется для разработки сайтов различного уровня сложности, от небольших блогов до крупных интернет-порталов и онлайн-магазинов. Основные особенности PHP:

* динамическая генерация контента, одним из ключевых преимуществ PHP является его способность генерировать динамический контент на веб-страницах. Это позволяет создавать интерактивные элементы, обрабатывать формы, работать с базами данных и создавать персонализированные страницы для пользователей;
* взаимодействие с базами данных, PHP обладает инструментами для работы с различными системами управления базами данных (СУБД), такими как mysql, postgresql, sqlite и другими. Он позволяет выполнять запросы к базам данных, получать, изменять и удалять данные, что делает его идеальным инструментом для создания сайтов с динамическим содержимым.
* универсальность и гибкость, PHP может быть использован для широкого спектра задач, начиная от создания скриптов обратной связи и заканчивая разработкой сложных веб-приложений.
* обширные библиотеки и фреймворки. Существует множество библиотек и фреймворков для PHP, которые упрощают разработку веб-приложений и предоставляют готовые решения для типичных задач, таких как аутентификация пользователей, маршрутизация URL-адресов, обработка форм и другие.
* PHP является кроссплатформенным языком программирования и может быть запущен на различных операционных системах, включая Windows, Linux, macos и другие.
* большое сообщество и обширная документация, PHP имеет большое сообщество разработчиков и обширную документацию, что облегчает изучение языка и получение поддержки при разработке проектов;
* безопасность в PHP может быть слабым местом при неправильном использовании, существует множество методов и рекомендаций для обеспечения безопасности веб-приложений на этом языке, таких как защита от SQL-инъекций, кросс-сайтового скриптинга и других угроз.

PHP продолжает оставаться одним из наиболее популярных и востребованных языков программирования для веб-разработки благодаря своей мощности, гибкости и широкому применению.

**1.4 Сервер баз данных MySQL**

MySQL — это одна из самых популярных открытых реляционных систем управления базами данных (СУБД), широко используемая для хранения и управления данными во многих веб-приложениях. Основные особенности MySQL:

* MySQL предназначен для хранения и организации данных в структурированном формате. Он обеспечивает механизмы для создания баз данных, таблиц и индексов для эффективного хранения и обработки информации.
* MySQL использует язык структурированных запросов (SQL) для взаимодействия с базой данных. С помощью SQL можно выполнять различные операции с данными, такие как вставка, выборка, обновление и удаление.
* MySQL предоставляет механизмы управления доступом к данным и базе данных. Администраторы могут настраивать права доступа пользователей к конкретным таблицам или операциям.
* MySQL обеспечивает механизмы безопасности для защиты данных от несанкционированного доступа. Это включает в себя установку паролей, шифрование данных и другие меры защиты.
* MySQL обладает хорошей масштабируемостью и может работать как на небольших, так и на крупных проектах. Он поддерживает горизонтальное и вертикальное масштабирование для увеличения производительности и обработки больших объемов данных.
* Работа с MySQL через PhpMyAdmin:
* PhpMyAdmin предоставляет удобный веб-интерфейс для управления базами данных и таблицами MySQL. С его помощью можно создавать, изменять и удалять базы данных и таблицы, а также просматривать и редактировать структуру таблиц.
* PhpMyAdmin позволяет выполнять SQL-запросы прямо из веб-интерфейса. Это позволяет администраторам и разработчикам выполнять различные операции с данными и управлять базой данных.
* PhpMyAdmin поддерживает импорт и экспорт данных из и в базу данных MySQL из различных форматов файлов, таких как CSV, SQL и другие.
* PhpMyAdmin позволяет администраторам управлять пользователями и их привилегиями доступа к базам данных и таблицам. Это включает в себя создание новых пользователей, установку паролей и назначение различных уровней доступа.

В целом, MySQL предоставляет надежную и эффективную платформу для хранения и управления данными, а PHPMyAdmin облегчает администрирование баз данных MySQL через удобный веб-интерфейс. С их помощью разработчики и администраторы могут эффективно управлять базами данных и обеспечивать их безопасность и производительность.

**1.5 Среда разработки PhpStorm**

PhpStorm — это интегрированная среда разработки (IDE) для PHP, разработанная компанией JetBrains. Вот некоторые основные особенности и возможности PhpStorm. PhpStorm предоставляет мощный редактор кода с подсветкой синтаксиса, автозавершением, подсказками и другими инструментами для увеличения производительности:

* IDE предоставляет широкие возможности для работы с PHP, включая подсветку синтаксиса, структурное обнаружение ошибок, автодополнение, рефакторинг и т.д.
* PhpStorm включает в себя мощный отладчик для PHP, который позволяет запускать и отслеживать выполнение кода, устанавливать точки останова, анализировать значения переменных и многое другое.
* PhpStorm интегрируется с популярными системами контроля версий, такими как Git, SVN и Mercurial, что облегчает работу в команде и управление изменениями.
* PhpStorm предоставляет поддержку для множества PHP-фреймворков, таких как Symfony, Laravel, Yii, Zend Framework и других, включая функции автодополнения, шаблонов и рефакторинга.
* PhpStorm предоставляет возможности для работы с веб-технологиями, такими как HTML, CSS и JavaScript, включая поддержку автодополнения, проверки кода и отладки JavaScript.
* IDE интегрируется с различными базами данных, что позволяет работать с SQL запросами, просматривать данные и управлять структурой баз данных без покидания среды разработки.
* PhpStorm поддерживает широкий спектр плагинов, которые позволяют расширить его функциональность, а также предоставляет множество настроек для адаптации среды разработки под ваши потребности.

**2 Сбор информации об объекте**

Платформа по продажи товаров может быть очень полезной в целях бизнеса. Пользователи могут просматривать товары и приобретать товары через такую платформу. Такое WEB – приложение нужно для:

* увеличения целевой аудитории покупателей магазина.
* увеличение прибыли бизнеса по продаже товаров.

Важные детали в реализации WEB – приложения:

* приложение будет реализовано для администратора и пользователей;
* регистрация и аутентификация пользователей, пользователи должны иметь возможность создавать учетные записи и входить в систему.
* добавление контента администратор и руководитель должны иметь возможность добавлять новые товары, описания товара и изображения товаров.
* должна быть обеспечена безопасность для данных пользователей и безопасность их учетных записей.

**3 Создание базы данных**

Для создания WEB-приложения, база данных играет важную роль, поскольку она служит для хранения всех данных, необходимых для функционирования приложения и обеспечения пользовательского опыта. Вот несколько причин, почему база данных необходима для такого приложения:

* хранение пользовательских данных. База данных позволяет хранить информацию о пользователях, включая их учетные записи и товары.
* хранение категорий девайсов, это позволяет облегчить поиск товара на главной странице через поисковик, так как поиск товара производится не только по описанию товара, но и по категории.
* управление правами доступа. База данных позволяет определять права доступа пользователей к различным функциям приложения и контенту. Например, определение администраторов, которые в свою очередь могут добавлять или удалять товар из web-магазина.
* обеспечение целостности данных. Использование базы данных позволяет обеспечить целостность данных путем предотвращения потери или искажения информации.

Для создания базы данных для такого приложения необходимо определить её структуру и используемые таблицы, а также установить отношения между ними. Даталогическая модель для информационного сопровождения задачи выполняемой группой работников представлена на рисунке 1.

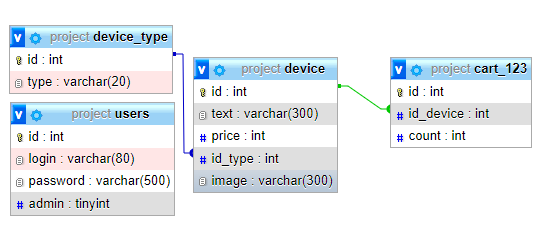


Рисунок 1 – Даталогическая модель

Рассмотрим каждую таблицу:

Таблица 1 - Пользователи

|  |  |  |
| --- | --- | --- |
| Столбец | Тип данных | Описание |
| id | Int | Уникальный идентификатор пользователя |
| name | varchar (255) | Уникальный логин пользователя |
| password | varchar (255) | Хэшированный пароль пользователя |
| admin | int | Уровень доступа пользователя |

Таблица 2 – Тип товара

|  |  |  |
| --- | --- | --- |
| Столбец | Тип данных | Описание |
| id | int | Уникальный идентификатор типа товара |
| type | varchar (255) | Уникальное наименование типа товара |

В таблице тип товара содержится информация о типе товара, название категории, тип товара облегчит поиск товара через поисковую строку, также тип товара можно будет использовать для получения статистических данных.

Таблица 3 - Девайсы

|  |  |  |
| --- | --- | --- |
| Столбец | Тип данных | Описание |
| id | Int | Уникальный идентификатор товара |
| title | varchar (300) | Описание товара |
| price | int | Цена товара |
| id\_type | Int | Уникальный идентификатор типа товара |
| Image | varchar (255) | Ссылка на фотографию товара |

В таблице девайсы, будет информация о всех товарах, которые есть в web-магазине.

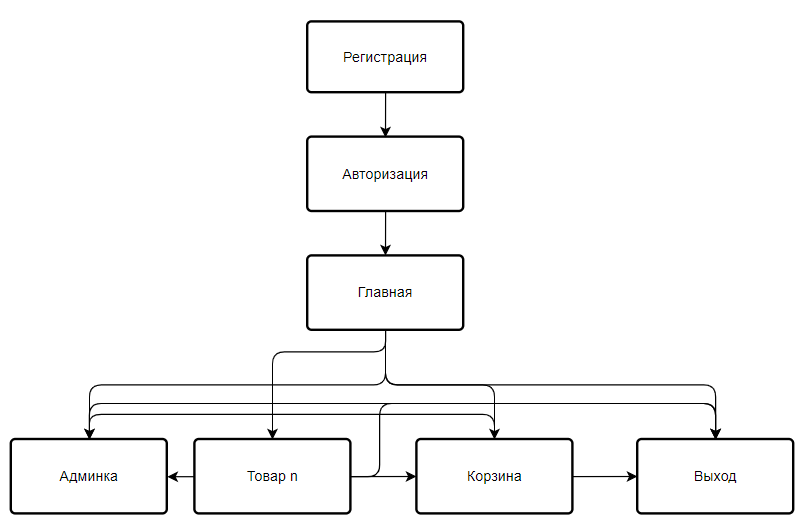
Таблица 4 – Корзина пользователя

|  |  |  |
| --- | --- | --- |
| Столбец | Тип данных | Описание |
| Id | int | Уникальный идентификатор |
| Id\_device | int | Уникальный идентификатор товара |
| Count | int | Количество товара в пользовательской корзине |

В таблице корзина будет представлена информация о добавленных в корзину товаров пользователем.

**4 Создание карты сайта или макета сайта**

Карта сайта содержит список всех доступных на веб-сайте страниц или разделов, обычно представленных в иерархическом порядке. Карта сайта облегчает поиск информации по сайту для поисковых роботов и пользователей. Рассмотрим карту сайта на рисунке 2.

 Рисунок 2 – Карта сайта

Основные разделы включают главную страницу, которая включает в себя ссылки на основные разделы сайта, ссылки на товары и выход со страницы.

Каждый раздел сайта логически связан с другими, обеспечивая удобную навигацию и удобный интерфейс для пользователей. Рассмотрим макет страницы приложения на рисунке 3.



Рисунок 3 – Макет страницы

На данном макете представлена структура WEB-приложения, которая состоит из следующих разделов:

* шапка – это верхняя часть страницы, обычно содержащая логотип, название сайта и основное меню навигации. Может также включать контактные данные, кнопки для входа в систему, поиск и другие элементы.
* поисковая строка облегчает пользователям поиск товаров на сайте. Обычно размещается в шапке страницы, но также может быть расположено в боковой панели.
* основное содержимое – это основная область страницы, содержащая основной контент, такой как товары, цены на товары, изображения товаров и описание товаров.
* подвал – это нижняя часть страницы, содержащая дополнительные ссылки, контактную информацию и другие вспомогательные элементы.

Макет страницы определяет расположение и структуру контента на веб-сайте. Он позволяет разработчикам и дизайнерам организовать информацию на странице таким образом, чтобы пользователи могли легко ориентироваться и быстро находить нужную им информацию.

**5 Создание шаблон дизайна страницы**

Шаблон страницы определяет общий стиль и внешний вид веб-сайта. основные элементы, шаблона дизайна страницы.

Пример шаблона дизайна главной страницы обычного пользователя сайта можно увидеть на рисунке 4.

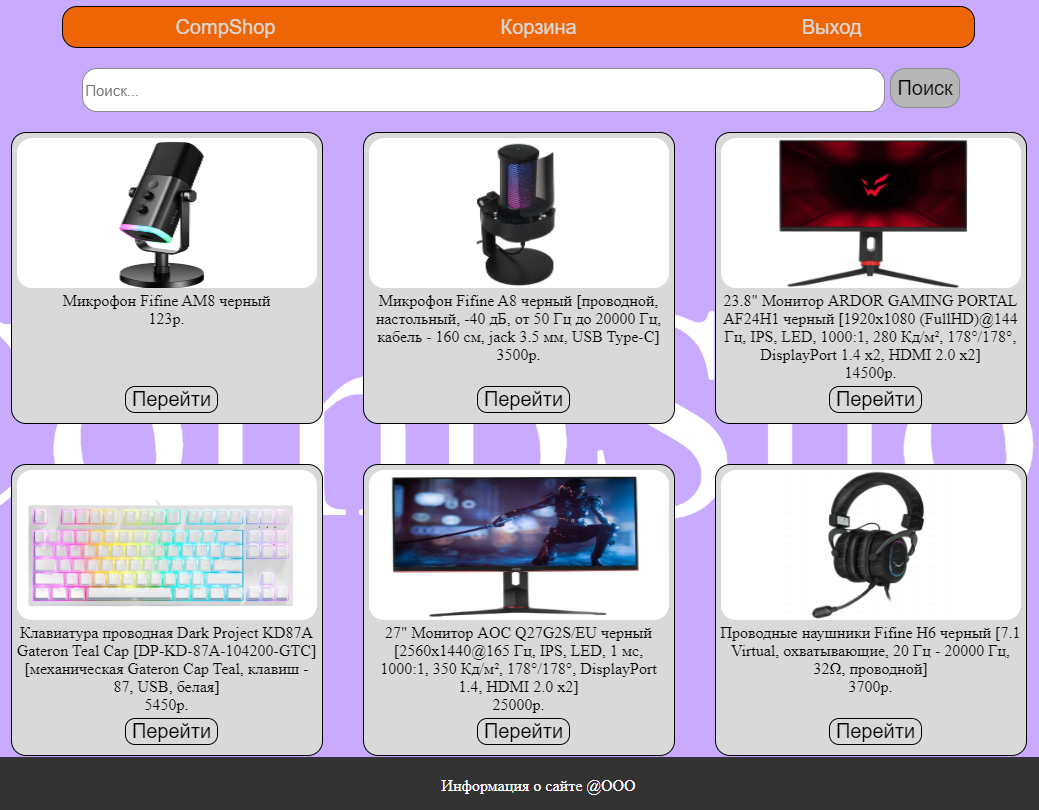


Рисунок 4 – Пример дизайна страницы

Цветовая палитра WEB - приложения выполнена в оранжевых, белых и сиреневых тонах оттенках. Цвета должны гармонировать друг с другом, чтоб не портить внешний вид и не создавать диссонанс. Более яркие цвета будут использоваться для выделения важной информации на странице. Для удобства восприятия информации рекомендуется использовать читаемые шрифты. Страницы будут иметь четкую и легко воспринимаемую структуру для удобной навигации пользователей. Ясное разделение различных элементов, таких как меню навигации, контент и подвал, делает навигацию по сайту интуитивно понятной и удобной. Использование высококачественных изображений, отражающих деятельность вашей организации, чтобы создать привлекательное визуальное впечатление.

Дизайн страницы должен быть привлекательным, функциональным и удобным для пользователей.

**6 Формирование контента**

Страница регистрации – это страница, на которой пользователи могут создать учетную запись для доступа к WEB-приложению или сервису. На странице представлены поля для ввода логина пользователя и пароля. Пользователи должны заполнить все обязательные поля и нажать кнопку "Регистрироваться", чтобы завершить процесс регистрации. Также на странице реализована авторизация, если пользователь уже имеет учетную запись. Для авторизации требуется нажать кнопку "Авторизация"

Главная страница – это основная страница WEB-приложения, на которой пользователи оказываются после успешной авторизации. На главной странице представлено навигационное меню, основная информация о товарах, которая автоматически загружается из базы данных, также представлен подвал. В зависимости от уровня доступа пользователя. Если пользователь является администратором, то на главной странице у него появляется возможность удалять товар из магазина и возможность перейти на страницу администратора.

Страница по каждому товару – на данной странице представлена основная информация, для выбранного пользователем товара и изображения товара. С этой страницы пользователь может добавить товар к себе в корзину.

Страница корзины – на данной странице представлена основная информация о содержании корзины пользователя WEB-приложения. Также возможность удалять товары из корзины и ввод персональной информации для оплаты товара и его доставки.

Страница администратора – на данной странице администратор может добавлять новые товары, указывая их основную информацию и загружать фото товара. Добавленные товары автоматически отобразятся на главной странице WEB-приложения.

**7 Вёрстка сайта**

Для разработки сайта использовался локальный сервер Open Server, обеспечивающий среду для работы с PHP и базой данных MySQL.

Содержание папки «CompShop» с проектом включает следующие файлы:

* registration.php этот файл представляет собой модуль HTML-страницы для регистрации и авторизации пользователя на веб-сайте. В нем используется PHP для проверки статуса пользователя и отображения сообщений об ошибках валидации. Форма регистрации и авторизации содержит поля для ввода логина и пароля. Пользователь должен заполнить обязательные поля и отправить форму, чтобы зарегистрироваться или авторизоваться на сайте. Если введенные данные не проходят валидацию, пользователю отображаются сообщения об ошибках. Если введенные данные проходят валидацию, произойдёт переход на главную страницу веб-сайте shop.php.
* index.php этот файл представляет собой HTML-страницу для авторизации пользователей на веб-сайте "CompShop". В этой странице модульно подгружается файл registration.php.
* shop.php этот файл является главной страницей веб-приложения "CompShop". Он предоставляет пользователю интерфейс для просмотра товара и поиска на стайте определённого товара. В файле присутствуют формы для перехода на страницу определённого товара. Страница также содержит шапку с логотипом и навигационным меню, поисковую строку, а также с информацией об авторском праве.
* admin.php этот файл является страницей администратора веб-приложения "CompShop". В файле также присутствуют формы для добавления новых товаров.
* protect.php этот PHP-скрипт является подключаемым модулем страниц. Функция файла в том, чтобы удостовериться что по ссылкам переходит авторизированный пользователь. Сначала он начинает сессию и проверяет, установлен ли идентификатор пользователя в сессии. Если не установлен, скрипт перенаправляет пользователя на страницу входа.
* cart.php этот файл представляет собой страницу корзины авторизованного пользователя. Также предоставляет содержимоего корзины пользователя и возможность взаимодействия с корзиной пользователя. Также есть возможность выбора типа оплаты покупки, ввода данных пользователя для связи и ввод адреса доставки.
* config.php этот файл устанавливает соединение с базой данных MySQL на локальном сервере. Если соединение не установлено, выводится сообщение об ошибке.
* product.php этот представляет собой страницу продукта, на это страницу можно попасть только с главной страницы WEB-приложения. Наполнение страницы происходит из базы данных. С этой страницы пользователи могут добавлять товары к себе в корзину.
* upload.php этот файл PHP-скрипт отвечает за обработку данных, полученных из формы на клиентской стороне и их добавление нового товара в базу данных. Он также устанавливает соединение с базой данных, проверяет наличие ошибок при подключении, получает данные из POST-запроса, обрабатывает изображение (если оно было загружено), генерирует SQL-запрос для добавления данных в базу данных, выполняет. После обработки данных он закрывает соединение с базой данных.
* top.php этот файл PHP является подключаемым модулем шапки, здесь находятся основные функции меню навигации.
* face.php этот PHP-скрипт отвечает за подключение файла style.css, который отвечает за дизайн страниц WEB-приложения.
* bottom.php этот файл PHP является подключаемым модулем подвала. Он загружается на всех страницах и содержит информацию об авторских правах WEB-приложения.
* styles.css этот файл для всех страниц, обеспечивающий их внешний вид и структуру.
* buy.js этот файл является клиентским скриптом Vue.js, который используется для создания интерактивного интерфейса на веб-странице. Он загружает список новостей с сервера при загрузке страницы и отображает их на странице.
* func.js этот скрипт файл отвечает за остановку анимации загрузки и скрытие загрузочного блока после полной загрузки содержимого страницы.

Рассмотрим реализацию WEB – приложения и его взаимодействие с базой данных. На рисунке 5 представлена форма авторизации и регистрации. В форме регистрации пользователь вводит свои уникальные данные. Для отправки в базу данных, они все должны быть заполнены. Когда пользователь нажимает на кнопку продолжить, происходит проверка на заполнение данных и запрос отправляется на сервер, в базе данных появляется информация о пользователе, если запрос был корректно сформирован.

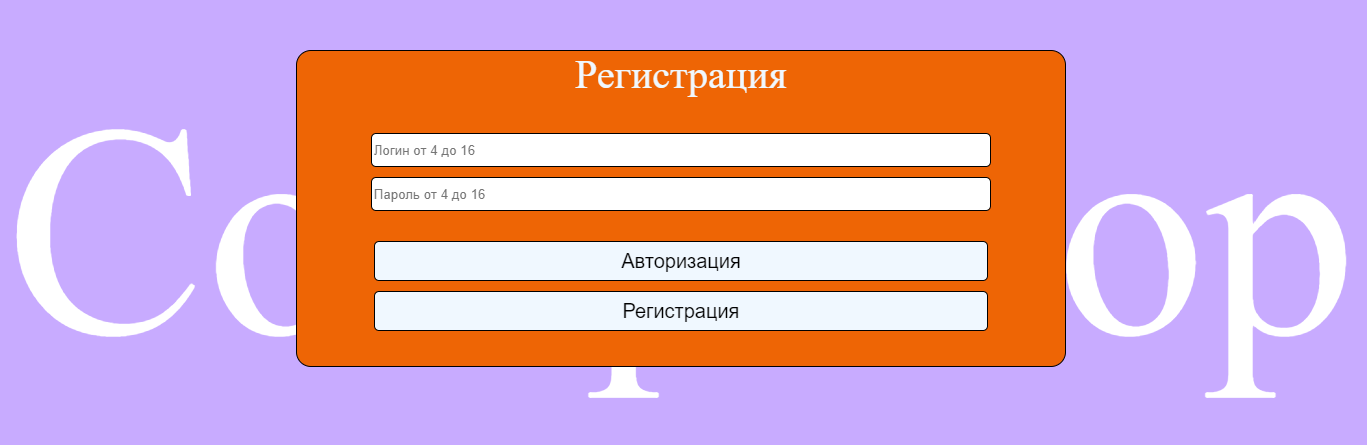


Рисунок 5 – Форма регистрации

В форму авторизации и регистрации пользователь, администратор вводят данные, которые были указаны при регистрации. Они отправляются на сервер, где происходит проверка данных, если данных не будет в базе данных, то вернется ошибка. Во время отправки данных устанавливается идентификатор пользователя, по сессии происходит проверка уровня доступа пользователя и открывается страница в зависимости от доступа. Рассмотрим WEB – приложение со стороны работы администратора на рисунке 9.

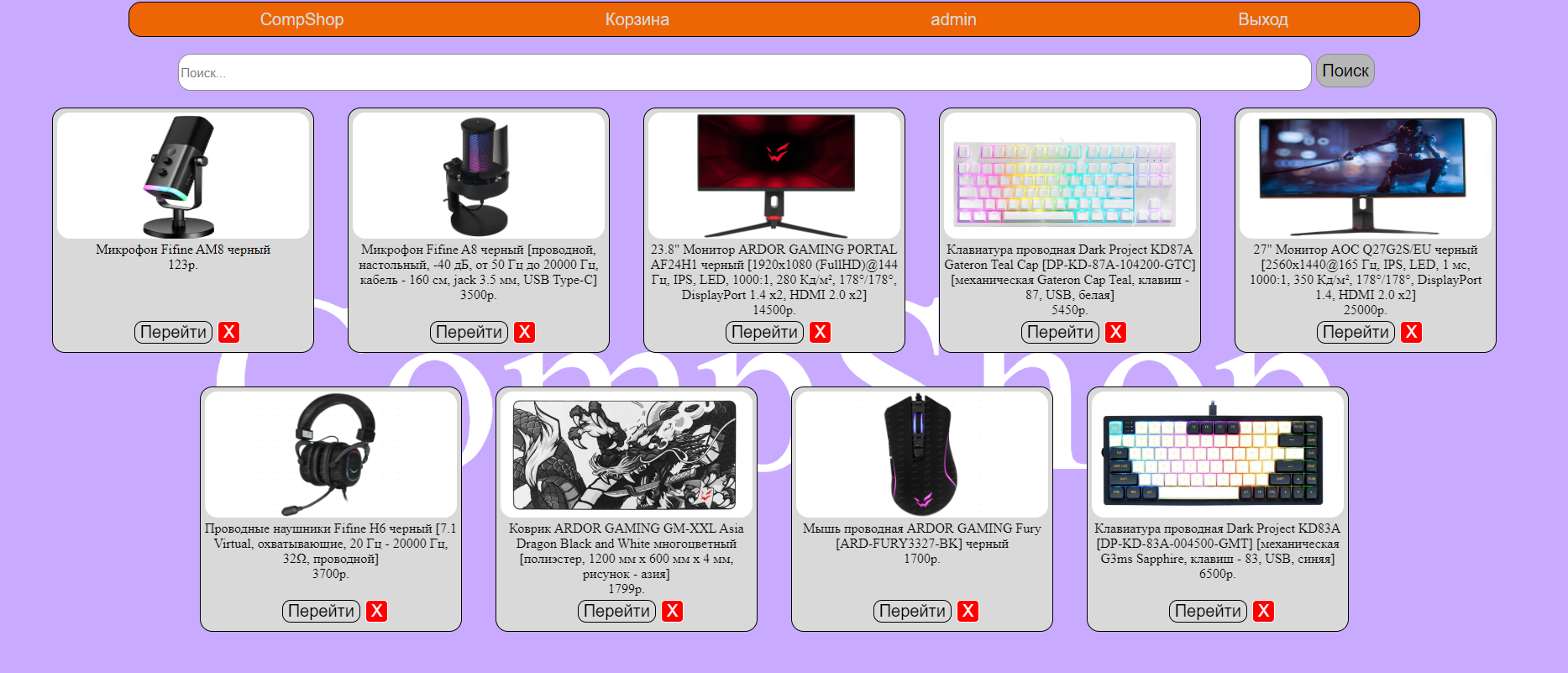


Рисунок 6 – Главная страница

На этой странице администратор с помощью форм «товар» может удалять данные о товаре из базы данных.

На станице админестратора администратор заполняет форму данными и отправляет их на сервер, если данные указаны корректно они добавляются в базу данных и затем отображаются на главной странице рисунок 10 и рисунок 11.

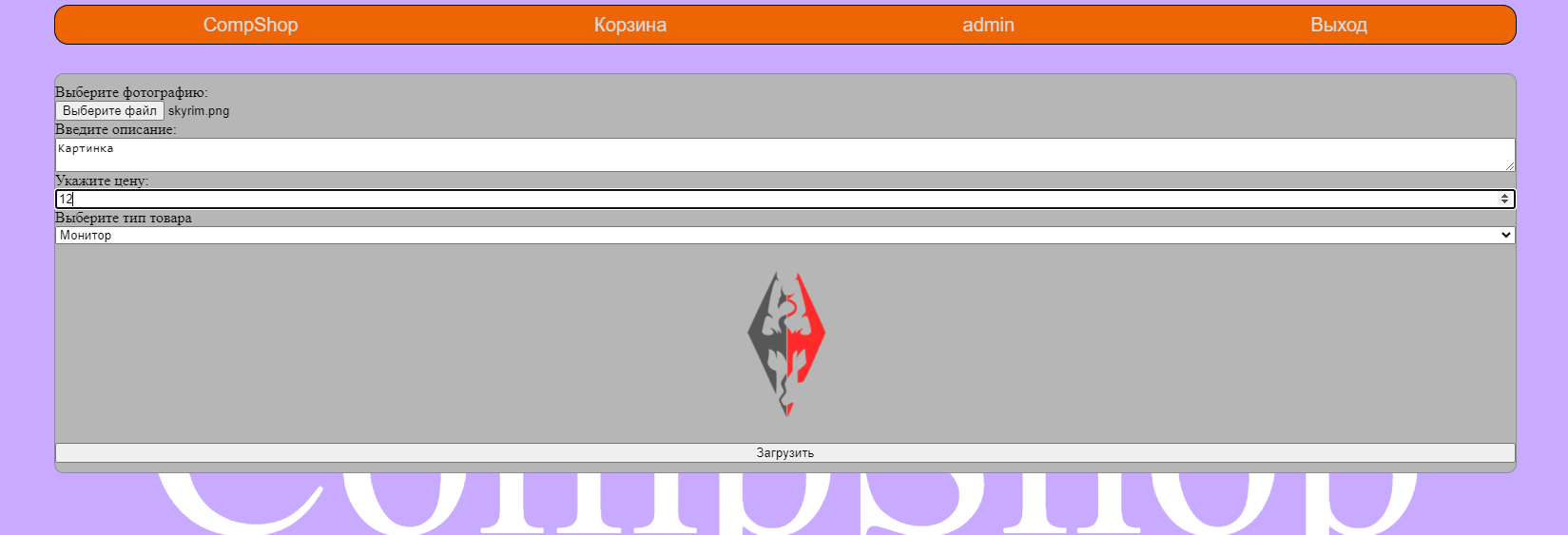


Рисунок 7 – Страница администратора



Рисунок 8 – Новый товар на главной странице

Работа WEB – приложения со стороны пользователя представлена на рисунке 9.

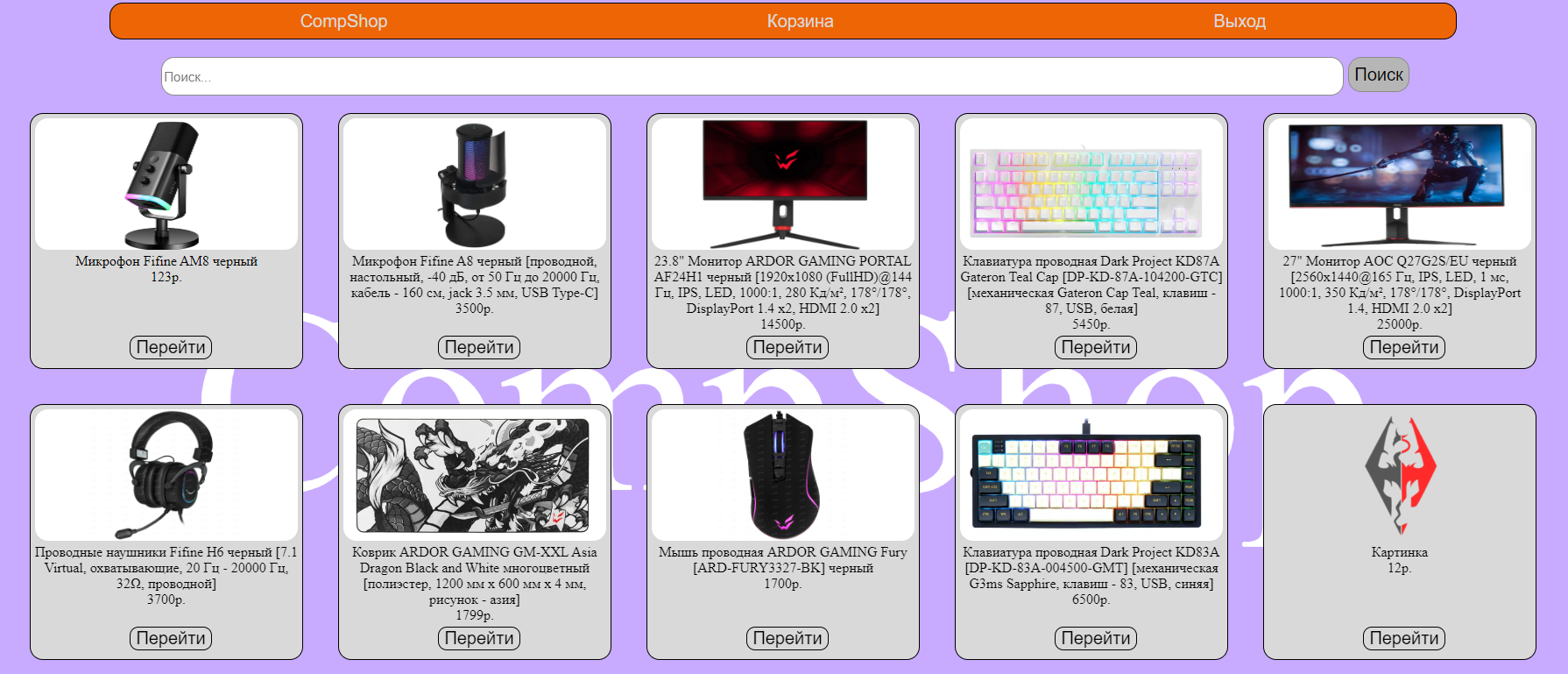


Рисунок 9 – Страница пользователя

Пользователь может просматривать товары, переходить на страницу товаров и добавлять их в свою корзину. Работа представлена на рисунке 10 и рисунке 11.

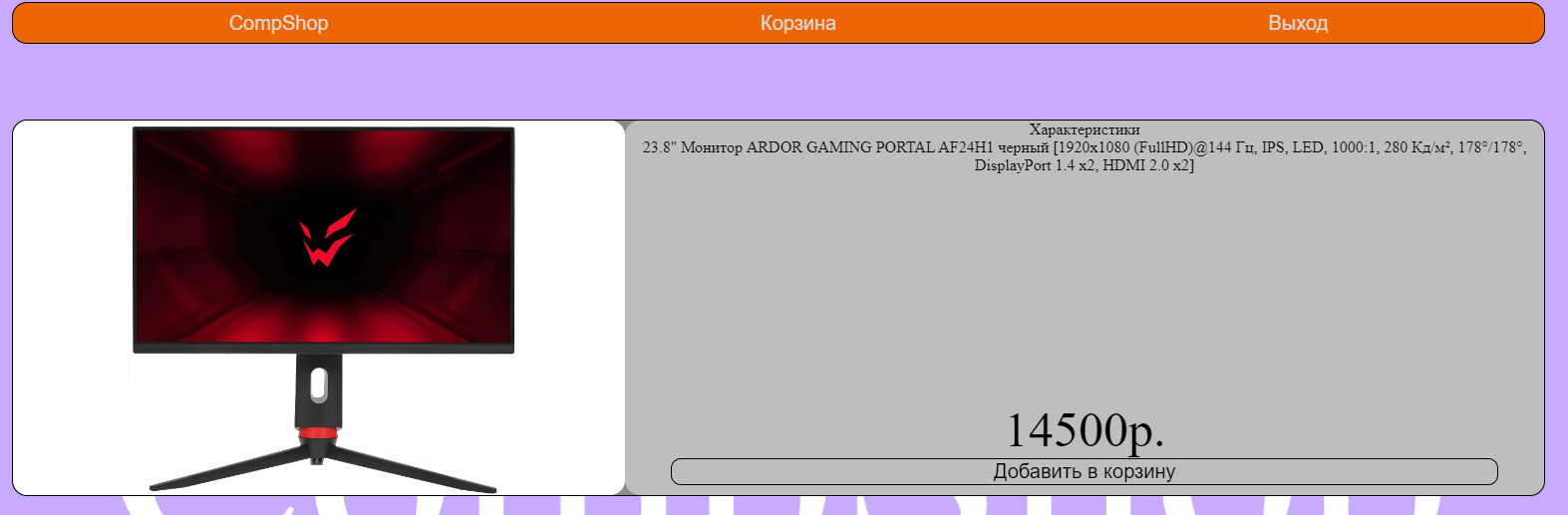


Рисунок 10 – Страница товара

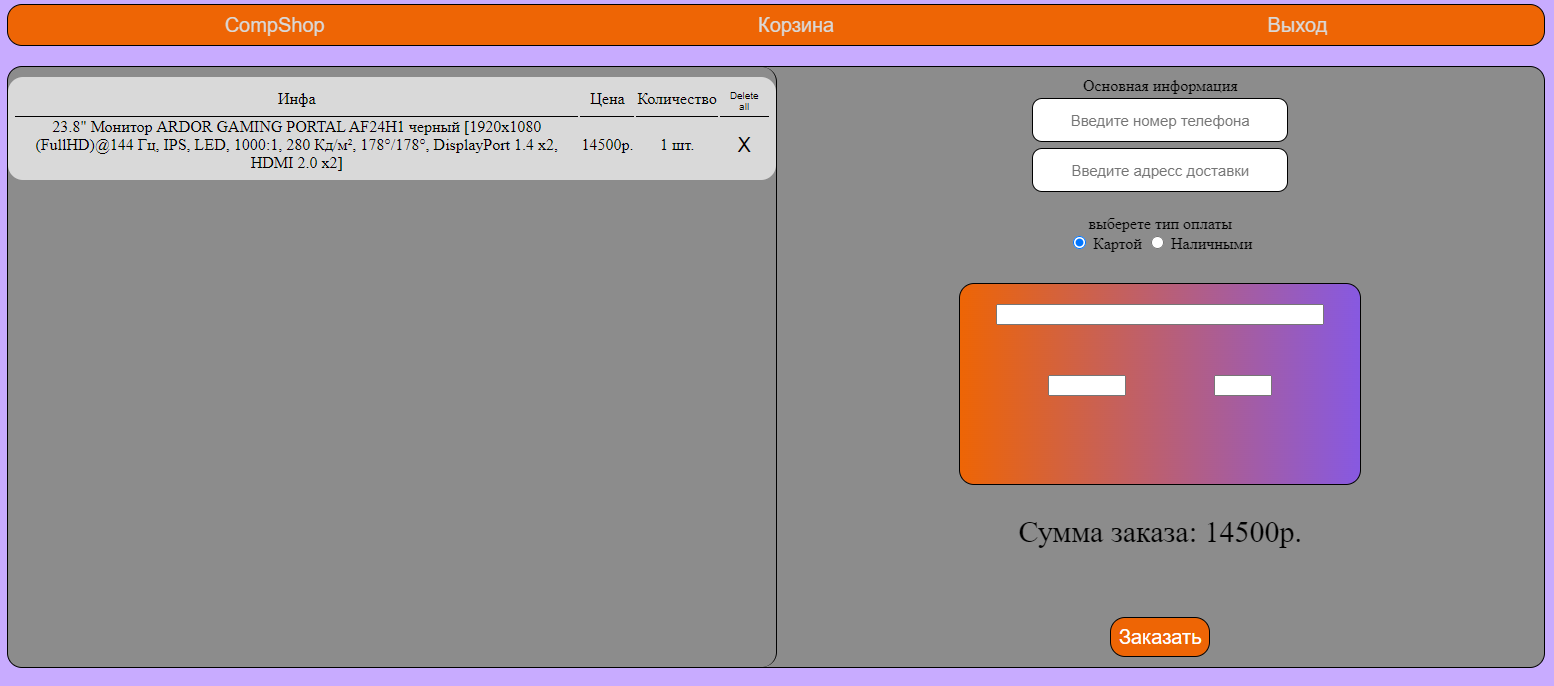


Рисунок 11 – Корзина пользователя

Все описание работы серверной части можно более детально описаны ниже.

Работа северной части начинается с процесса регистрации или авторизации. Данные, которые вводит пользователь, проходят проверку на валидность. Если проверка успешна пройдена, пользователь попадает на главную страницу web-приложения. Так же при авторизации из базы данных загружается информация об уровне доступа пользователя. Всего существует 2 уровня доступа: администратор и пользователь. После успешной авторизации на каждой странице проверяется факт того, что пользователь или администратор авторизован. В случае отсутствия факта авторизации, пользователь или администратор попадает на страницу авторизации.

Администраторы имеют доступ к взаимодействию с содержимым главной таблицы в базе данных, в которой хранятся товары. Таким образом администратор может управлять наполнением главной страницы сайта.

Пользователи, как и администраторы, могут взаимодействовать со своей корзиной, добавлять или удалять товары из своей корзины.

Все пользователи и администраторы имею доступ к поисковой строке главной страницы, с помощью которой производится запрос на выборку данных из главной таблицы базы данных.

Все взаимодействия связаны с отправкой sql запросов или использованием локальных сессий.

Таким образом обеспечивается надёжность и отказоустойчивость сайта, разграничение доступа и, как безопасность пользовательских данных, так и данных web-приложения.

**8 Тестирование и размещение сайта в сети**

Для обеспечения работоспособности сайта необходимо пройти через целый ряд тестовых сценариев, чтобы убедиться, что все функции работают правильно. Вот общий список тестовых сценариев для проверки работоспособности сайта:

Регистрация нового пользователя не успешная: ввести некорректные данные нового пользователя (например, неуникальный логин, или неподходящее по формату изображение) в форму регистрации и убедиться, что при попытке регистрации с некорректными данными выдается соответствующая ошибка и данные не добавляются в базу данных.

Авторизация пользователя не успешная: ввести учетные данные незарегистрированного пользователя (логин и пароль) в форму авторизации. Нажать кнопку "Авторизация" и убедиться, что система выдает ошибку входа на сайт.

Эти шаги помогут убедиться в корректной работе функций регистрации и авторизации. В случае возникновения ошибок необходимо проанализировать причины и внести соответствующие исправления.

Добавление нового товара на главную страницу не успешно:

* ввести некорректные данные для добавления нового товара (например, пустой текст). Нажать кнопку "Загрузить" и убедиться, что приложение выдает ошибку и не добавляет товар с некорректными данными.
* ввести корректные данные для создания нового блока (например, заполните текст и добавьте изображение). Нажать кнопку "Загрузить" и убедитесь, что новый товар успешно добавлен на главную страницу.

В всех случаях важно обеспечить корректное информирование пользователя о возникших проблемах и предложить понятные инструкции о дальнейших действиях. Также важно регистрировать и сохранять информацию об ошибках для последующего анализа и устранения проблемы.

Редактирование и удаление блока, успешная работа:

Добавление товара в корзину успешно:

* выбрать товар.
* нажать кнопку "Добавить".
* проверить наличие записи о товаре в корзине.
* удалить товар из корзины.
* уведомить пользователя об ошибке и предложить повторить попытку.

Удаление товара с главной страницы успешно:

* выбрать товар для удаления.
* нажать кнопку, отвечающую за удаление.
* проверить удаление товара из пользовательских корзин для предотвращения ошибок наличия уже не существующего товара.
* уведомить пользователя об ошибке и предложить повторить попытку.

Регистрация нового пользователя успешная включает в себя введение корректные данные нового пользователя (например, неуникальный логин, или неподходящее по формату изображение) в форму регистрации и убедиться, что при попытке регистрации с корректными данными не выдается ошибка и данные добавляются в базу данных. Также убедиться, что пользователь успешно добавлен в базу данных. Проверить таблицу пользователей в базе данных, чтобы убедиться, что новая запись была добавлена с корректными данными.

Авторизация пользователя успешная:

* ввести учетные данные незарегистрированного пользователя (логин и пароль) в форму авторизации. нажать кнопку "авторизация" и убедиться, что система не выдает ошибку входа на сайт.
* убедиться, что после успешной авторизации пользователь перенаправляется на главную страницу. проверить, что доступные функции, такие как поиск товара, добавление или удаление товара из корзины. Проверить доступ ко всем страницам только авторизованным пользователям.

Нефункциональное тестирование:

* производительность – это оценка скорость загрузки страниц и выполнения запросов к базе данных. убедиться, что сайт работает быстро и отзывчиво.
* безопасность включает в себя проверку наличие уязвимостей безопасности. обеспечить защиту данных пользователей и аутентификацию.
* совместимость требует проверить работоспособность сайта на различных браузерах (chrome, firefox, safari, edge) и устройствах (компьютеры, планшеты, мобильные телефоны).
* отзывчивый дизайн – убедиться, что сайт корректно отображается и работает на разных разрешениях экрана и устройствах.
* проверка корректного отображения сайта и его функций на различных мобильных устройствах с разными размерами экранов.
* убедиться, что элементы управления и контент легко доступны и пригодны для использования на мобильных устройствах.
* проверка корректности управления сеансами пользователей, чтобы убедиться, что авторизованные пользователи не могут получить доступ к защищенным страницам или функциям без входа в систему.
* проверка производительности базы данных при обработке запросов на чтение и запись.
* оценка времени ответа базы данных при выполнении различных операций, таких как создание, обновление и удаление записей.

После проведения тестирования необходимо внимательно проанализировать результаты, исправить обнаруженные ошибки и улучшить функциональность и удобство использования виртуальной доски.

Подготовка базы данных на Handy Host:

* cоздание базы данных postgresql через панель управления Handy Host.
* cоздание пользователя базы данных с необходимыми правами доступа.
* Тестирование соединения:
* написание тестового скрипта для подключения к базе данных postgresql на Handy Host и выполнения простого запроса.
* запуск тестового скрипта для проверки успешного подключения и выполнения запросов.
* Импорт данных:
* подготовка данных для импорта, если они есть.
* использование инструментов Handy Host или командной строки для импорта данных в созданную базу данных.

Настройка безопасности:

* проверка и обновление настроек безопасности базы данных postgresql на Handy Host, включая права доступа к таблицам и пользователям.
* регулярное обновление паролей и других настроек безопасности.

Настройка резервного копирования базы данных postgresql реализована на Handy Host через инструменты, предоставленные хостинг-провайдером.

Масштабирование и оптимизация:

* оценка производительности и масштабирование базы данных на Handy Host, при необходимости увеличение выделенных ресурсов.
* оптимизация запросов и индексов для повышения производительности.

Мониторинг и обслуживание:

* регулярное мониторинг состояния и производительности базы данных на Handy Host.
* реагирование на любые проблемы или сбои в работе базы данных и веб-приложения.
* обновление веб-приложения и базы данных для поддержания их работоспособности и безопасности.

**Заключение**

При выполнении работы над Web-приложением онлайн магазин были пройдены следующие этапы:

* планирование и проектирование. Эти этапы включают в себя определение функциональных и нефункциональных требований к приложению, создание архитектуры приложения и выбор технологий.
* разработка функциональности – это создание основных функций приложения, таких как добавление и удаление товаров, работа с пользователями, загрузка изображений и другие.
* тестирование и отладка – это проведение тестирования для обнаружения и исправления ошибок, а также улучшение производительности и безопасности приложения.
* доработка и оптимизация включает в себя улучшение интерфейса, добавление дополнительных функций, оптимизация кода и базы данных для повышения производительности и улучшения пользовательского опыта.

В результате был создан онлайн магазин, позволяющий пользователям просматривать, выбирать и покупать товары, а также управлять своими заказами. Реализация магазина включает использование современных технологий веб-разработки, таких как css для создания динамического интерфейса, PHP для обработки серверных запросов и взаимодействия с базой данных, а также AJAX для обмена данными между клиентом и сервером без перезагрузки страницы. Приложение обладает базовым набором функций, необходимых для работы с товарами, и может быть доработано и расширено в будущем в соответствии с потребностями пользователей.

**Перечень использованных информационных ресурсов**

1. Официальная документация языка разметки HTML [Электронный ресурс], URL: https://www.w3.org/2019/04/WHATWG-W3C-MOU.html.

2. Официальная документация каскадных таблиц стилей [Электронный ресурс], URL: https://www.w3.org/Style/CSS/Overview.en.html.

3. Официальная документация языка программирования JS [Электронный ресурс], URL: https://www.javascript.com.

4. Официальная документация языка программирования РНР [Электронный ресурс], URL: <https://www.php.net/manual/ru/index.php>

5. Официальная документация MySQL [Электронный ресурс], URL: <https://www.opennet.ru/docs/RUS/mysqlrus/>

**Приложение А**

Листинг А.1 – файл Admin.php

<?php global $conDB;

require\_once("config.php");

ob\_start();

session\_start();

?>

<!doctype html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport"

content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>admin</title>

<?php

require\_once $\_SERVER['DOCUMENT\_ROOT'] . '/face.php';

if ($\_SESSION['login'] == '' || $\_SESSION['admin'] == 0) {

session\_destroy();

header("Location: index.php");

exit();

}

?>

</head>

<body>

<?php

require\_once $\_SERVER['DOCUMENT\_ROOT'] . '/top.php';

?>

<div class="inputInfo">

<form action="admin.php" class="face\_content\_in\_admin" method="post" enctype="multipart/form-data">

Выберите фотографию: <input type="file" name="fileToUpload" accept="image/\*">

Введите описание: <textarea name="info"></textarea>

Укажите цену: <input type="number" name="price">

<?php

$bd\_info = mysqli\_query($conDB, "SELECT `id`,`type` FROM `device\_type`");

if ($bd\_info) {

echo "Выберите тип товара";

echo "<select name='type\_of\_device'>";

while ($row = mysqli\_fetch\_assoc($bd\_info)) {

echo "<option value='$row[id]'>";

echo "$row[type]";

echo "</option>";

}

echo "</select>";

}

?>

<div id="imageUpload" class="imageUpload">

<img id="previewImage" src="#" alt="Предпросмотр изображения" style="display: none; max-width: 300px;">

</div>

<input type="submit" value="Загрузить" name="upload">

</form>

</div>

<?php

require\_once $\_SERVER['DOCUMENT\_ROOT'] . '/upload.php';

?>

<script src='func.js'></script>

<?php

require\_once $\_SERVER['DOCUMENT\_ROOT'] . '/bottom.php';

?>

</body>

</html>

Листинг А.2 – файл bottom.php

<footer>

Информация о сайте @ООО

</footer>

Листинг А.3 – файл cart.php

<?php global $conDB;

require\_once("config.php");

ob\_start();

?>

<!doctype html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport"

content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Корзина</title>

<script src='buy.js'></script>

<?php

require\_once $\_SERVER['DOCUMENT\_ROOT'] . '/face.php';

?>

</head>

<body class="cartBody">

<?php

session\_start();

if ($\_SESSION['login'] == '') {

header("Location: index.php");

exit();

}

?>

<?php

require\_once $\_SERVER['DOCUMENT\_ROOT'] . '/top.php';

?>

<?php

require\_once $\_SERVER['DOCUMENT\_ROOT'] . '/face.php';

?>

<div class="cart\_main">

<div class="cart\_keep">

<table>

<tr>

<td>Инфа</td>

<td>Цена</td>

<td>Количество</td>

<td>

<form method="get" action="cart.php">

<button type="submit" name="delete\_all" style="color: black; font-size: 10px">

Delete all

</button>

</form>

</td>

</tr>

<?php

if (isset($\_SESSION['login'])) {

if (intval($\_SESSION['login']) != 0) {

$id = $\_SESSION['login'];

$bd\_info = mysqli\_query($conDB, "SELECT `device`.`text`, `device`.`price`, `cart\_$\_SESSION[login]`.`count`, `cart\_$\_SESSION[login]`.`id`

FROM `device`

JOIN `cart\_$\_SESSION[login]` ON `cart\_$\_SESSION[login]`.`id\_device` = `device`.`id`");

static $summ = 0;

if (mysqli\_num\_rows($bd\_info) > 0) {

while ($row = mysqli\_fetch\_assoc($bd\_info)) {

echo "<tr>";

echo "<td style='border-top: black solid 1px'>";

echo "$row[text]";

echo "</td>";

echo "<td style='border-top: black solid 1px'>";

echo "$row[price]р.";

echo "</td>";

echo "<td style='border-top: black solid 1px'>";

echo "$row[count] шт.";

echo "</td>";

echo "<td style='border-top: black solid 1px'>";

echo "<form action='cart.php' method='get'>";

echo "<button type='submit' style='color: black' value=$row[id] name='delete'>";

echo "X";

echo "</button>";

echo "</form>";

echo "</td>";

echo "</tr>";

$summ += $row["price"] \* $row["count"];

}

} else {

echo "<tr>";

echo "<td class='nothing' colspan=4 style='border-top: black solid 1px'>";

echo "У вас нет ничего в корзине";

echo "</td>";

echo "</tr>";

}

}

}

?>

</table>

</div>

<div class="cart\_info">

<div class="infoClient">

<div style="text-align: center">Основная информация</div>

<input type="text" name="phone" pattern="[0-9()-+]\*" placeholder="Введите номер телефона">

<input type="text" name="link" placeholder="Введите адресс доставки">

</div>

<div>

<div style="text-align: center">выберете тип оплаты</div>

<div>

<input type="radio" name="type\_of\_money" value="Картой" checked onclick="toggleCart(this)">

Картой

<input type="radio" name="type\_of\_money" onload="true" value="Наличными" onclick="toggleCart(this)">

Наличными

</div>

</div>

<div class="cartBuy" id="cartBuy" style=" display: flex; flex-direction: column; align-items: center;">

<input class="numberOfCart" type="text" oninput="checkLength2(this)" pattern="[0-9]\*" maxlength="19">

<div class="numberOfCart2" >

<input class="dataOfCart" type="text" oninput="checkLength(this, 4)" pattern="[0-9]\*">

<input class="CVC" type="password" maxlength="3">

</div>

</div>

<div style="margin-top: 30px">

<label style="font-size: 30px;">Сумма заказа:

<?php

echo "$summ"."р.";

?>

</label>

</div>

<button class="button\_buy">

Заказать

</button>

</div>

</div>

<?php

if (isset($\_GET['delete'])) {

mysqli\_query($conDB, "DELETE FROM `cart\_$\_SESSION[login]` WHERE `id` LIKE $\_GET[delete];");

header("Location: cart.php");

exit();

}

if (isset($\_GET['delete\_all'])) {

mysqli\_query($conDB, "DELETE FROM `cart\_$\_SESSION[login]` WHERE `id` LIKE '%';");

header("Location: cart.php");

exit();

}

?>

<?php

require\_once $\_SERVER['DOCUMENT\_ROOT'] . '/bottom.php';

?>

</body>

</html>

Листинг А.4 – файл face.php

<link rel="stylesheet" href="style.css">

<div class="logo">

CompShop

</div>

Листинг А.5 – файл index.php

<?php require\_once("config.php");

ob\_start();

?>

<!doctype html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport"

content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>CompShop</title>

<?php

require\_once $\_SERVER['DOCUMENT\_ROOT'].'/face.php';

?>

</head>

<body style="min-width: 0px">

<?php

require\_once $\_SERVER['DOCUMENT\_ROOT'] . '/registration.php';

?>

</body>

</html>

Листинг А.6 – файл product.php

<?php global $conDB;

require\_once("config.php");

ob\_start();

?>

<!doctype html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport"

content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>product</title>

<?php

require\_once $\_SERVER['DOCUMENT\_ROOT'] . '/face.php';

?>

<?php

session\_start();

if ($\_SESSION['login'] == '') {

header("Location: index.php");

exit();

}

?>

</head>

<body>

<?php

require\_once $\_SERVER['DOCUMENT\_ROOT'] . '/top.php';

?>

<?php

require\_once $\_SERVER['DOCUMENT\_ROOT'] . '/face.php';

?>

<?php

if (isset($\_SESSION['goToProduct'])) {

if (intval($\_SESSION['goToProduct']) != 0) {

$id = $\_SESSION['goToProduct'];

$bd\_info = mysqli\_query($conDB, "SELECT `device`.\*

FROM `device`

JOIN `device\_type` ON `device\_type`.`id` = `device`.`id\_type`

WHERE `device`.`id` LIKE '$id';");

if ($bd\_info) {

while ($row = mysqli\_fetch\_assoc($bd\_info)) {

echo "<br>";

echo "<br>";

echo "<form method='post' action='product.php'>";

echo "<div class='product'>";

echo "<img class='photo\_in\_product' src='$row[image]' alt='photo'>";

echo "<div class='info'>";

echo "Характеристики";

echo "<br>";

echo "$row[text]";

echo "<div class='buy'>";

echo "<div class='price'>";

echo "$row[price]р.";

echo "</div>";

echo "<button name='add\_to\_cart' value=$id class='button\_product' type='submit'>";

echo "Добавить в корзину";

echo "</button>";

echo "</div>";

echo "</div>";

echo "</div>";

echo "</form>";

// echo "cart\_$\_SESSION[login]";

}

}

}

}

?>

<?php

require\_once $\_SERVER['DOCUMENT\_ROOT'] . '/bottom.php';

?>

<?php

if (isset($\_POST['add\_to\_cart'])) {

// Проверяем существует ли запись для данного id\_device

$result = mysqli\_query($conDB, "SELECT \* FROM `cart\_$\_SESSION[login]` WHERE `id\_device` = '$\_POST[add\_to\_cart]'");

if (mysqli\_num\_rows($result) > 0) {

// Если запись существует, обновляем ячейку count

mysqli\_query($conDB, "UPDATE `cart\_$\_SESSION[login]` SET `count` = `count` + 1 WHERE `id\_device` = '$\_POST[add\_to\_cart]'");

} else {

// Если записи не существует, вставляем новую запись

mysqli\_query($conDB, "INSERT INTO `cart\_$\_SESSION[login]` (`id\_device`, `count`) VALUES ('$\_POST[add\_to\_cart]','1')");

}

header("Location: product.php");

exit();

}

?>

</body>

</html>

Листинг А.7 – файл protect.php

<?php

if ($\_SESSION['login'] == '') {

session\_destroy();

header("Location: index.php");

exit();

}

Листинг А.8 – файл registration.php

<?php global $conDB;

require\_once("config.php");

ob\_start();

?>

<div class="content\_rege">

<div class="rege">

<form action="index.php" id="myForm" method="post">

<div class="text-registration">

Регистрация

</div>

<div class="login">

<input class="login\_info" type="text" name="login" placeholder="Логин от 4 до 16">

<input class="login\_info" type="password" name="password" placeholder="Пароль от 4 до 16">

</div>

<div class="buttons">

<button type="submit" name="logging">

Авторизация

</button>

<button type="submit" name="reg">

Регистрация

</button>

</div>

</form>

<?php

if (isset($\_POST['reg'])) {

$login = $\_POST['login'];

$password = $\_POST['password'];

// Если ничего не введено, то

if ($login == "" || $password == "") {

echo "<div class='errors'>вы не ввели логин или пароль</div>";

// Иначе добавляем запись (INSERT INTO)

} else {

$login = mysqli\_real\_escape\_string($conDB, $login);

$password = mysqli\_real\_escape\_string($conDB, $password);

$result = mysqli\_query($conDB, "SELECT \* FROM `users` WHERE `login`='$login';");

if ($result && mysqli\_num\_rows($result) > 0) {

echo "<div class='errors'>Такой пользователь уже существует</div>";

} else {

$hashed\_password = password\_hash($password, PASSWORD\_DEFAULT);

mysqli\_query($conDB, "INSERT INTO `users`

SET `login`='$login',`password`='$hashed\_password', `admin` = 0");

mysqli\_query($conDB, "CREATE TABLE IF NOT EXISTS `cart\_$login` (

`id` INT AUTO\_INCREMENT PRIMARY KEY,

`id\_device` INT NOT NULL,

`count` INT NOT NULL,

FOREIGN KEY (`id\_device`) REFERENCES `device`(`id`) ON DELETE CASCADE

);");

header("Location: index.php");

exit();

}

}

}

?>

<?php

session\_start();

if (isset($\_POST['logging'])) {

$login = $\_POST['login'];

$password = $\_POST['password'];

if ($login == "" or $password == "") {

echo "<div class='errors'>Некорректные данные</div>";

} else {

$login = mysqli\_real\_escape\_string($conDB, $login);

$password = mysqli\_real\_escape\_string($conDB, $password);

// $hashed\_password = password\_hash($password, PASSWORD\_DEFAULT);

$result = mysqli\_query($conDB, "SELECT \* FROM `users` WHERE `login`='$login';");

if ($result && mysqli\_num\_rows($result) > 0) {

$row = mysqli\_fetch\_assoc($result);

if(password\_verify($password, $row['password'])) {

// Успешная авторизация

$\_SESSION['admin'] = $row['admin'];

$\_SESSION['login'] = $login;

header("Location: shop.php");

exit();

}// Важно завершить выполнение скрипта после вызова header

} else {

// Неверный логин или пароль

echo "<div class='errors'>Неверный логин или пароль</div>";

session\_destroy();

}

}

}

?>

</div>

</div>

Листинг А.9 – файл shop.php  
<?php

global $conDB;

require\_once("config.php");

ob\_start();

session\_start();

?>

<!doctype html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport"

content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>CompShop</title>

<?php

require\_once $\_SERVER['DOCUMENT\_ROOT'] . '/face.php';

?>

</head>

<body>

<!--Проверка авторизации через данные сессии-->

<?php

require\_once $\_SERVER['DOCUMENT\_ROOT'] . '/protect.php';

?>

<?php

require\_once $\_SERVER['DOCUMENT\_ROOT'] . '/top.php';

?>

<form action="shop.php" class="search" method="post">

<input class="search\_text" type="text" name="search\_text" placeholder="Поиск...">

<button type="submit" class="search\_text" name="search">Поиск</button>

</form>

<div class="face\_content">

<?php

if (isset($\_POST['search'])) {

if (strval($\_POST['search\_text']) != "") {

$search = $\_POST['search\_text'];

$search = mysqli\_real\_escape\_string($conDB, $search);

$bd\_info = mysqli\_query($conDB, "SELECT `device`.\*

FROM `device`

JOIN `device\_type` ON `device\_type`.`id` = `device`.`id\_type`

WHERE `device`.`text` LIKE '%$search%' OR `device\_type`.`type` LIKE '%$search%' ;");

if ($bd\_info) {

while ($row = mysqli\_fetch\_assoc($bd\_info)) {

show($row);

}

}

} else {

$bd\_info = mysqli\_query($conDB, "SELECT \* FROM `device`");

if ($bd\_info) {

while ($row = mysqli\_fetch\_assoc($bd\_info)) {

show($row);

}

}

}

} else {

$bd\_info = mysqli\_query($conDB, "SELECT \* FROM `device`");

if ($bd\_info) {

while ($row = mysqli\_fetch\_assoc($bd\_info)) {

show($row);

}

}

}

?>

</div>

<?php

function show($row)

{

global $conDB;

echo "<form action='shop.php' method='post'>";

echo "<div class='main\_content'>";

echo "<img src='$row[image]' style='width: 100%; height: 150px' alt='photo'>";

echo "<br>";

echo "$row[text]";

echo "<br>";

echo "$row[price]р.";

echo "<br>";

echo "<div class='selec'>";

echo "<button name='goToProduct' value=$row[id] style='color: black; border: 1px solid black; border-radius: 10px;'>";

echo "Перейти";

echo "</button>";

if ($\_SESSION['admin'] == true) {

echo "<button type='submit' name='delet\_device' value=$row[image] style='color: white; border-radius: 5px; border:solid 1px white; background-color: red'>";

echo "X";

echo "</button>";

}

echo "</div>";

echo "</div>";

echo "</form>";

if (isset($\_POST['goToProduct'])){

$\_SESSION['goToProduct'] = $\_POST['goToProduct'];

header("Location: product.php");

exit();

}

if (isset($\_POST['delet\_device'])){

$fileToDelete = $\_POST['delet\_device'];

// Проверяем, существует ли файл перед удалением

if (file\_exists($fileToDelete)) {

// Попытка удалить файл

if (unlink($fileToDelete)) {

mysqli\_query($conDB, "DELETE FROM `device` WHERE `image` LIKE '$fileToDelete';");

echo 'Файл успешно удален.';

} else {

echo 'Не удалось удалить файл.';

}

header("Location: shop.php");

exit();

} else {

echo 'Файл не существует.';

}

}

}

?>

<?php

require\_once $\_SERVER['DOCUMENT\_ROOT'] . '/bottom.php';

?>

</body>

</html>

Листинг А.10 – файл top.php

<div class="head">

<form id="myForm" method="post">

<button type="submit" name="shop">CompShop</button>

<!-- <button>Тех.поддержка</button>-->

<button type="submit" name="cart">Корзина</button>

<?php

if ($\_SESSION['admin'] == true) {

echo "<button type='submit' name='admin'>";

echo "admin";

echo "</button>";

}

?>

<button type="submit" name="exit">Выход</button>

</form>

<?php

if (isset($\_POST['exit'])){

session\_destroy();

header("Location: index.php");

exit();

}

if (isset($\_POST['shop'])){

header("Location: shop.php");

exit();

}

if (isset($\_POST['admin'])){

header("Location: admin.php");

exit();

}

if (isset($\_POST['cart'])){

header("Location: cart.php");

exit();

}

?>

</div>

Листинг А.11 – файл upload.php

<?php

global $conDB;

if (isset($\_POST['upload'])) {

$target\_dir = "images/";

$target\_file = $target\_dir . basename($\_FILES["fileToUpload"]["name"]); // Полный путь к файлу

$check = getimagesize($\_FILES["fileToUpload"]["tmp\_name"]);

if ($check == false) {

echo "Файл не является изображением.";

exit();

}

// Проверка, существует ли уже такой файл

if (file\_exists($target\_file)) {

echo "Извините, файл с таким названием уже существует. Переименуйте файл.";

exit();

}

// Проверка размера файла (не более 10MB)

if ($\_FILES["fileToUpload"]["size"] > 10000000) {

echo "Извините, ваш файл слишком большой.";

exit();

}

if ($\_POST['info'] == "") {

echo "Ошибка, вы не указали описание";

exit();

}

$price = $\_POST['price'];

$type\_of\_device = $\_POST['type\_of\_device'];

$info = $\_POST['info'];

$info = mysqli\_real\_escape\_string($conDB, $info);

$image = $target\_dir . $\_FILES["fileToUpload"]["name"];

if($price == "" || $type\_of\_device == "" || $info == ""){

echo "Ошибка данных";

exit();

}

if (move\_uploaded\_file($\_FILES["fileToUpload"]["tmp\_name"], $target\_file)) {

echo "Файл " . basename($\_FILES["fileToUpload"]["name"]) . " успешно загружен.";

} else {

echo "Произошла ошибка при загрузке файла.";

}

mysqli\_query($conDB, "INSERT INTO `device`

SET `text`='$info',`price`='$price', `id\_type` = '$type\_of\_device', `image` = '$image'");

header("Location: admin.php");

exit();

}

?>

Листинг А.12 – файл buy.js

function toggleCart(radio) {

var cartBuy = document.getElementById('cartBuy');

if (radio.value === 'Картой') {

cartBuy.style.display = 'flex';

} else if (radio.value === 'Наличными') {

cartBuy.style.display = 'none';

}

}

function checkLength(input, maxLength) {

if (input.value.length > maxLength) {

input.value = input.value.slice(0, maxLength);

}

if (input.value.includes("/") & input.value.length == 4){

input.value = input.value.replace("/", "");

}

if (input.value.length == 4) {

input.value = input.value.slice(0, 2) + "/" + input.value.slice(2, 4);

}

}

function checkLength2(input) {

if ((input.value.length === 4 || input.value.length === 9 || input.value.length === 14) && input.value.length < 20) {

input.value = input.value + " ";

}

}

Листинг А.13 – файл func.js

document.querySelector('input[type=file]').addEventListener('change', function() {

let file = this.files[0]; // Получаем выбранный файл

if (file) {

let reader = new FileReader(); // Создаем объект FileReader

// Событие onLoad вызывается, когда файл был полностью загружен

reader.onload = function(e) {

let imagePreview = document.getElementById('previewImage'); // Получаем элемент предварительного просмотра изображения

let imageUpload = document.getElementById('previewImage'); // Получаем элемент предварительного просмотра изображения

imagePreview.src = e.target.result; // Устанавливаем источник изображения для предварительного просмотра

imagePreview.style.display = 'block';// Показываем предварительный просмотр изображения

imageUpload.style.padding = '20px';

};

// Читаем файл как URL-адрес данных (Data URL)

reader.readAsDataURL(file);

}

});

Листинг А.14 – файл style.css

body{

min-width: 750px;

margin: 8px 0 60px 0;

height: 100%;

position: relative;

background-color: #c8abff;

}

.logo{

position: fixed;

z-index: -2;

display: flex;

font-size: 300px;

align-items: center;

justify-content: center;

width: 100%;

height: 100%;

color: white;

}

.cartBody{

min-width: 1000px;}

.content\_rege{

display: flex;

justify-content: center;

align-items: center;

}

.rege{

height: 315px;

width: 40%;

border: solid 1px black;

border-radius: 15px;

background-color: #ee6505;

margin-top: 30vh;

min-width: 305px;

}

.text-registration{

display: flex;

justify-content: center;

font-size: 40px;

color: aliceblue;

}

.errors{text-align: center;}

.login{

margin-top: 30px;

margin-bottom: 20px;

display: flex;

flex-direction: column;

width: 100%;

align-items: center;

}

.login\_info{

margin-top: 5px;

margin-bottom: 5px;

height: 30px;

width: 80%;

border-radius: 5px;

border: solid 1px

}

.buttons{

display: flex;

flex-direction: column;

justify-content: center;

align-items: center;

}

button{

color: #d9d9d9;

font-size: 20px;

border: 0px;

background-color: transparent;

}

button:hover{

cursor: pointer;

}

.buttons button{

margin-top: 5px;

margin-bottom: 5px;

width: 80%;

height: 40px;

background-color: aliceblue;

border-radius: 5px;

border: solid 1px;

color: black;

}

.head form{

height: 40px;

border: black 1px solid;

border-radius: 15px;

margin: auto;

display: flex;

justify-content: space-around;

flex: 0 0.7;

width: 80%;

background-color: #ee6505;

}

.search{

margin-top: 20px;

width: 100%;

display: flex;

justify-content: center;

}

.search input{

width: 70%;

background-color: white;

font-size: 15px;

}

.search\_text{

height: 40px;

border: solid 1px #8c8c8c;

border-radius: 15px;

color: black;

margin-left: 5px;

background-color: #b7b6b6;

}

.face\_content{

display: flex;

flex-wrap: wrap;

flex: 0 0;

justify-content: center;

}

.main\_content{

width: 300px;

height: 280px;

background-color: #d9d9d9;

border: black 1px solid;

border-radius: 15px;

padding: 5px;

margin: 20px;

text-align: center;

position: relative;

}

.main\_content img{

border-radius: 15px;

}

.inputInfo{

background-color: #b7b6b6;

border: solid 1px #8c8c8c;

border-radius: 10px;

color: black;

width: 80%;

margin-left: auto;

margin-right: auto;

margin-top: 30px;

padding-top: 10px;

padding-bottom: 10px;

}

.face\_content\_in\_admin{

display: flex;

flex-wrap: wrap;

flex: 0 0.7;

justify-content: space-between;

flex-direction: column;

}

.imageUpload{

display: flex;

justify-content: center;

align-items: center;

height: 100%; /\* Высота контейнера равна высоте экрана \*/

}

.product{

background-color: #8c8c8c;

border: black solid 1px;

width: 80%;

max-height: 900px;

margin: auto;

margin-top: 40px;

border-radius: 15px;

display: flex;

wrap-option: wrap;

flex: 0 0.3;

}

.photo\_in\_product{

border-radius: 15px;

object-fit: contain;

position: relative;

width: 40%;

}

.info{

background-color: #bebebe;

width: 60%;

border-radius: 15px;

text-align: center;

position: relative;

}

.buy{

position: absolute;

bottom: 10px;

text-align: center;

display: flex;

justify-content: space-around;

width: 100%;

flex-direction: column;

align-items: center;

}

.price{

font-size: 50px;

}

.button\_product{

color: black;

border: 1px solid black;

border-radius: 10px;

width: 90%;

}

.cart\_main{

width: 80%;

margin: auto;

margin-top: 20px;

background-color: #8c8c8c;

border-radius: 15px;

height: 600px;

display: flex;

min-width: 815px;

border: black solid 1px;

}

.cart\_keep, .cart\_info{

width: 50%;

padding-top: 10px;

padding-bottom: 10px;

overflow: auto;

}

.cart\_keep{

border-right: black solid 1px;

text-align: center;

border-radius: 15px;

}

.selec{

width: 100%;

position: absolute;

bottom: 10px;

top: auto;

display: flex;

justify-content: center;

}

.selec button{

margin-left: 3px;

margin-right: 3px;

}

::-webkit-scrollbar {

width: 12px; /\* Ширина скроллбара \*/

}

::-webkit-scrollbar-track {

background: #888; /\* Цвет трека (фоновая область) \*/

}

::-webkit-scrollbar-thumb {

background: #555; /\* Цвет бегунка (ползунка) \*/

border-radius: 6px; /\* Закругление углов бегунка \*/

}

.cart\_info{

position: relative;

display: flex;

flex-wrap: wrap;

flex-direction: column;

align-items: center;

}

.infoClient{

display: flex;

flex-direction: column;

margin-bottom: 20px;

}

.infoClient input{

width: 250px;

height: 40px;

font-size: 15px;

text-align: center;

border-radius: 10px;

border: solid black 1px;

margin: 3px;

}

.cartBuy{

margin-top: 30px;

/\*background-color: white;\*/

border-radius: 15px;

border: 1px solid black;

height: 200px;

width: 400px;

display: none;

background: linear-gradient(to right, #ee6505, #8759e1);

}

.numberOfCart{

width: 80%;

margin-top: 20px;

margin-bottom: 50px;

text-align: center;

}

.numberOfCart2{

width: 100%;

display: flex;

justify-content: space-evenly;

}

.dataOfCart{

width: 70px;

}

.CVC{

width: 50px;

}

.button\_buy{

position: absolute;

bottom: 10px;

margin: auto;

border: black solid 1px;

border-radius: 15px;

height: 40px;

width: 100px;

color: white;

background-color: #ee6505;

}

table{

background-color: #d9d9d9;

width: 100%;

border-radius: 15px;

padding: 5px;

height: 50px;

}

.nothing{

text-align: center;

}

footer{

background-color: #333; /\* Цвет фона подвала \*/

color: #fff; /\* Цвет текста в подвале \*/

text-align: center; /\* Выравнивание текста по центру \*/

padding: 20px; /\* Внутренний отступ \*/

position: fixed; /\* Фиксированное положение \*/

width: 100%; /\* Ширина 100% \*/

bottom: 0; /\* Прижимаем к нижней части окна браузера \*/

}

Листинг А.15 – файл config.php

<?php

$nameDB = "project";

$nameSERVER = "127.0.0.1";

$nameUSER = "root";

$passUSER = "root";

$conDB = mysqli\_connect($nameSERVER, $nameUSER, $passUSER, $nameDB) or die("ERROR".mysqli\_error($nameDB));

mysqli\_query($conDB, "SET NAMES utf8");

?>