**Содержание**

Введение 3

1. Теоретические основы веб-программирования 4

1.1. HTML – язык гипертекстовой разметки 4

1.2. CSS – каскадные таблицы стилей 5

1.3. Скриптовый язык общего назначения PHP 6

1.4. Проприетарный текстовый редактор Sublime Text 7

1.5. Свободная реляционная система управления базами данных MySQL и веб-интерфейс для администрирования СУБД MySQL phpMyAdmin 8

2. Разработка веб-сайта для магазина канцелярских товаров 10

2.1. Вёрстка сайта на HTML с использованием CSS 10

2.2. Работа с PHP и базой данных 18

Заключение 26

Список использованных источников 27

Приложения 28

**Введение**

Во время стремительного прогресса просто красиво оформленный текст и картинки на веб-сайте уже никого не удивят. Требования к сайтам, изменились - теперь для успешного представительства компании в Интернет необходимо предоставить своим посетителям различные возможности: обратную связь, форум, голосование, интернет-магазин, различные веб-тесты, поиск по сайту, счётчик посещений и многое другое. При помощи обычного html этого не достичь, ведь html – это язык гипертекстовой разметки – инструмент для создания гиперссылок, вставки изображений, таблиц и др. С помощью языка html легко и быстро можно сделать форму для отправки какого-либо запроса. Такая форма будет иметь все необходимые атрибуты: и поле для ввода текста, и кнопку отправки. Однако, при нажатии на такую кнопку в большинстве случаев не произойдет ровным счетом ничего, поскольку не был задан сценарий действий, которые следует выполнить, чтобы получить результат. Поэтому, создание интерактивных компонентов – это задача для веб-программирования.

Актуальность создания веб-сайтов заключается в том, что это намного упрощает взаимодействие между клиентом и какой-нибудь условной компанией. Если клиент захочет заказать какой-либо продукт, то ему не придется даже звонить, достаточно будет только заполнить форму и заказ оформлен.

Целью курсовой является разработка сайта для магазина канцтоваров.

Для достижения поставленной цели необходимо выполнить следующие задачи:

* ознакомиться с литературой и интернет источниками по веб-разработке;
* сверстать сайт с использованием HTML и CSS;
* подключить сайт к СУБД MySQL;
* добавить к сайту код PHP;
* сделать вывод по проделанной работе.

**1. Теоретические основы веб-программирования**

**1.1. HTML – язык гипертекстовой разметки**

HTML – это язык разметки, который представляет простые правила оформления и компактный набор структурных и семантических элементов разметки (тегов). HTML позволяет описывать способ представления логических частей документа (заголовки, абзацы, списки и т.д.) и создавать веб-страницы разной сложности.

HTML не является языком программирования, но веб-страницы могут содержать встроенные или загружаемые программы на скриптовых языках (в первую очередь Javascript) и программы-апплеты на языке Java [5].

В языке гипертекстовой разметки HTML используются простые команды, именуемые тегами, для определения различных частей -фрагментов. Ниже приведен код HTML простой веб-страницы:

<!DOCTYPE>

<html>

<head>

<meta charset=”UTF-8”>

<title>Это заголовок веб-страницы</title>

</head>

<body>

<p>А это абзац этой страницы</p>

</body>

</html>

Конечно, пример очень простой, но демонстрирует все основные элементы, необходимые обычной веб-странице. В нем вы заметите то, что называется объявлением типа документа — doctype, за ним следует тег (со скобками), потом head (голова, заголовок), следом body (тело, тело документа), а в нем непосредственно содержимое веб-страницы. Все это завершается закрывающим тегом [2].

**1.2. CSS – каскадные таблицы стилей**

CSS означает каскадные таблицы стилей. Это язык для указания того, как HTML документ отображается на веб-странице. Без CSS каждый веб-сайт был бы просто Times New Roman с крошечными кнопками. Однако он способен на гораздо большее, чем стилизация текста. CSS позволяет нам определить целые макеты и элементы положения и даже выполнять анимацию. Таблицы стилей говорят сами за себя, но что такое каскадная таблица стилей? Потому что больше чем одно правило стиля может применяться к данному элементу HTML, должен быть какой-то способ определить, какое правило должно применяться в случае конфликта. Стили «каскадируют» из менее конкретных к более конкретным селекторам, и наиболее конкретное правило побеждает. Специфика – это важное понятие в CSS [4].

CSS работает с HTML, но не имеет к HTML никакого отношения. Это совершенно другой язык. HTML структурирует документ, упорядочивая информацию в заголовки, абзацы, маркированные списки и т.д., в то время как CSS тесно взаимодействует с браузером, чтобы оформление HTML-документа имело совершенный вид.

Например, вы могли бы использовать HTML, чтобы превратить фразу в заголовок, отделяя его от содержания страницы, но лучше применять CSS для форматирования заголовка, скажем, большим полужирным красным шрифтом с позиционированием на 50 пикселов от левого края окна. В CSS это форматирование текста включает в себя стиль – правило, описывающее внешний вид конкретной части веб-страницы. А таблица стилей (stylesheet) является набором таких стилей.

Можно также создавать стили специально для работы с изображениями. Например, с помощью стилей можно выровнять изображение по правому краю веб-страницы, поместить его в цветную рамку, отделить от окружающего текста на 50 пикселов.

Создав стиль один раз, можно применять его к текстовым фрагментам, изображениям, заголовкам и любым другим элементам страницы сколько угодно. Например, вы можете выбрать абзац текста и применить к нему стиль, тут же изменяющий размер, цвет и шрифт текста. Можно также сделать стили для определенных HTML-тегов так, чтобы, например, все заголовки первого уровня (теги) на вашем сайте были отображены в одинаковом стиле, независимо от того, где они размещены [2].

Таблица стилей CSS состоит из правил. Правила CSS нацелены на элементы HTML с помощью селекторов, которые описывают элементы, к которым должны применяться стили.

Правило состоит из селектора, за которым следует блок свойств CSS, содержащийся внутри фигурных скобок. Свойства состоят из значения, разделенных двоеточием и разделяются точкой с запятой. Значение может быть одним или набором нескольких значений в зависимости от свойства. Селектор прописывается через точку. Например: .header.

Каждый элемент в документе, который соответствует селектору, имеет свойства, к которому применено правило CSS [4].

**1.3. Скриптовый язык общего назначения PHP**

PHP – это язык программирования общего назначения, используемый в основном для веб-разработки. Созданный Расмусом Лердорфом в 1994 году, он позволяет разработчикам с легкостью создавать динамические веб-страницы.

Например, разработчики могут создать форму в HTML и обработать ее с помощью PHP. В зависимости от входных данных, введенных в форму, разработчики могут использовать PHP для отображения различных выходных данных для пользователей. Большую часть времени PHP используется в качестве серверного языка. Это означает, что код PHP не обрабатывается на компьютере пользователя (также известном как клиент).

Другими словами, когда вы открываете страницу PHP в своем браузере, код не обрабатывается на вашем компьютере. Вместо этого ваш браузер отправляет запрос на веб-сервер, который затем обрабатывает код и возвращает результат в браузер в виде веб-страницы.

Чаще всего этот веб-сервер представляет собой удаленный компьютер, на котором хранятся файлы PHP. Чтобы веб-сервер мог обрабатывать код PHP, необходимо установить специальное программное обеспечение, известное как интерпретатор PHP [3].

Типичный PHP-сценарий — это набор выражений. Каждое выражение начинается с новой строчки и заканчивается точкой с запятой.

Выражение — это инструкция, которая приказывает PHP-интерпретатору выполнить одно действие, например, сложить два числа или вывести на экран информацию. Давайте посмотрим на самый простой сценарий. Он будет выводить на экран одну строку: «Привет, Мир!»:

<?php print("Привет, мир");

Важно отметить, что любой PHP-сценарий обязательно начинается с такой строки: <?php - так мы сообщаем веб-серверу, что дальше в файле находится код на языке PHP.

В нашем примере сценарий состоит из всего одной инструкции: print("Привет, мир");

Здесь мы просим функцию print вывести на экран наш текст — «Привет, Мир!». Это можно рассматривать как подлежащее и сказуемое в русском языке, где функция print — это сказуемое, а «Привет, Мир!» — подлежащее.

Дополнительный фрагмент ?> в конце сообщает серверу, где заканчивается наш PHP-сценарий. В случае, если код последний в документе, и после него ничего не следует, этот фрагмент не обязателен [7].

**1.4. Проприетарный текстовый редактор Sublime Text**

Sublime Text – проприетарный текстовый редактор. Поддерживает плагины на языке программирования Python.

Проект начался в ноябре 2007 года с целью создать «лучший текстовый редактор», первая версия программы вышла 18 января 2008 года и была доступна только под Windows.

Разработчик позволяет бесплатно и без ограничений ознакомиться с продуктом, однако программа уведомляет о необходимости приобретения лицензии.

Sublime Text поддерживает большое количество языков программированияи имеет возможность подсветки синтаксиса для C, C++, C#, CSS, D, Dylan, Erlang, HTML, Groovy, Haskell, Java, JavaScript, LaTeX, Lisp, Lua, Markdown, MATLAB, OCaml, Perl, PHP, Python, R, Ruby, Rust, SQL, TCL и XML.

В дополнение к тем языкам программирования, которые включены по умолчанию, пользователи имеют возможность загружать плагины для поддержки других языков.

Sublime Text может быть оснащён менеджером пакетов, который позволяет пользователю находить, устанавливать, обновлять и удалять пакеты без перезагрузки программы. Менеджер поддерживает установленные пакеты в актуальном состоянии, загружая новые версии из репозиториев. Кроме того, он предоставляет команды для активации и деактивации установленных пакетов [12].

**1.5. Свободная реляционная система управления базами данных MySQL и веб-интерфейс для администрирования СУБД MySQL phpMyAdmin**

MySQL – одна из наиболее используемых систем управления базами данных. MySQL управляет реляционными базами данных, то есть такими, в которых таблицы связаны между собой.

MySQL работает по принципу клиент-сервер. Компьютер пользователя (клиент) отправляет запрос. Сервер баз данных его обрабатывает и предоставляет ответ. Именно поэтому часто можно услышать понятие MySQL-сервер. Это сервер, на котором хранится база данных.

Система MySQL написана на языках программирования C и C++. Для работы MySQL используется язык структурированных запросов SQL [11].

SQL или Structured Query Language (язык структурированных запросов) — язык программирования, предназначенный для управления данными в СУБД. Все современные СУБД поддерживают SQL.

На языке SQL выражаются все действия, которые можно провести с данными: от записи и чтения данных, до администрирования самого сервера СУБД.

Для повседневной работы совсем не обязательно знать весь этот язык; достаточно ознакомиться лишь с основными понятиями синтаксиса и ключевыми словами. Кроме того, SQL очень простой язык по своей структуре, поэтому его освоение не составит большого труда.

Язык SQL – это в первую очередь язык запросов, а кроме того он очень похож на естественный язык.

Каждый раз, когда требуется прочитать или записать любую информацию в БД, требуется составить корректный запрос. Такой запрос должен быть выражен в терминах SQL [6].

phpMyAdmin – это приложение написанное на PHP и обеспечивающее полноценную, в том числе удаленную, работу с базами данных MySQL через браузер. Так как phpMyAdmin позволяет во многих случаях обойтись без непосредственного ввода команд SQL, то работа с базами данных становится вполне посильной задачей даже для человека весьма поверхностно знакомого с MySQL.

Активное использование MySQL в веб-программировании обусловило его актуальность, а интуитивно понятный интерфейс в совокупности с широкой функциональностью и поддержкой более 60 языков (в т. ч. и русского) обеспечило ему популярность среди веб-разработчиков [10].

**2. Разработка веб-сайта для магазина канцелярских товаров**

**2.1. Вёрстка сайта на HTML с использованием CSS**

В папке OpenServer требуется перейти в папку domains, а из нее в localhost. В этой папке создается папка для сайта. Называем ее «курсовая 2 курс». Первым делом будем писать код для главной страницы. Но для начала стоит подготовить папки и материалы, которые понадобятся при написании. А именно: папка для CSS стилей, изображений, сами изображения и шрифты. После всего создаем файл с index.php. Файл с таким расширением, поскольку в дальнейшем будет добавляться код php.

Далее задаем структуру документ, как показано в таблице 1. Делается это просто вводом тега <html>, а дальше программа Sublime Text самостоятельно вводит нужные значения.

Таблица 1 – Структура HTML документа

|  |
| --- |
| HTML |
| <!DOCTYPE html>  <html>  <head>  <meta charset="utf-8">  <title> </title>  </head>  <body>  </body> |

Первым делом всегда заполняется «голова», код которой приведен в таблице 2. В теге <head> находятся теги <title>, <meta> и <link>. <title> позволяет задать название для страницы. В данном случае это будет «Ножницы - интернет-магазин канцелярских товаров». Тег <meta> с атрибутом charset задает кодировку документа. При помощи тега <link> с атрибутом href устанавливается связь со внешним документом, которым будет являться файл со стилями. Также, создаем еще один тег <link> для подключения иконок font-awesome и тег <meta> с именем viewport. Это пригодится для адаптивной верстки. Первый параметр width дает команду браузеру адаптировать по ширине устройства страницу. Масштабирование ширины страницы определяется автоматически, если указано значение «device-width». Еще один параметр – initial-scale – означает коэффициент масштабирования. Если указано значение «1», разрешение CSS и устройства приравниваются друг к другу независимо от размера экрана. Можно указать любое значение параметра от 0.1 до 10. В случае, показанном в таблице 2, это значение равно 1.

Таблица 2 – «Голова»

|  |
| --- |
| HTML |
| <!DOCTYPE html>  <html>  <head>  <link rel="stylesheet" type="text/css" href="css/styles.css">  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/font-awesome/4.5.0/css/font-awesome.min.css">  <meta name="viewport" content="width=device-width, initial-scale=1">  <meta charset="utf-8">  <title>Ножницы - интернет-магазин канцелярских товаров </title>  </head>  <body>  </body> |

Чтобы задать шрифт нужно их сперва подключить. Для этого, в файле CSS прописываем правило @font-face. В свойстве font-family указывается имя для шрифта, а в src – его путь. Весь процесс подключения показан в таблице 3.

Таблица 3 – Подключение шрифтов

|  |
| --- |
| CSS |
| @font-face{font-family: Vetka; src: url(../fonts/Vetka.otf);}  @font-face{font-family: Roboto; src: url(../fonts/Roboto.ttf);} |

После всего вышеперечисленного приступаем к обозначению блоков на странице, как показано в приложении 1. Для этого, используем теги <header>, <main> и <footer>. В теге <header> будут находится блоки для верхней части страницы – «шапки». В селекторах с псевдоклассом :hover задаем свойство transition для более плавного перехода при наведении. Для селектора header указывается ширина в процентах, для того, чтобы при уменьшении он плавно изменялся.

Вторым этапом прописывается «тело» сайта согласно приложению 2. В теге <main>, так же как и в шапке, размещаем нужные блоки. Для создания перемещения на слайдерах воспользовались псевдоклассом :target и указали id, на который будет переходить при нажатии. Вывод товаров и брендов будет осуществлен с помощью php.

И под конец, «подвал» сайта, усмотренный в приложении 3. Тег <footer> так же заполняется нужными нам блоками.

В шапке при нажатии на иконку с изображением упрощенного бюста человека всплывает модальное окно. Для него прописывается код в соответствии с таблицей 3. В дальнейшем, оно будет подключаться на страницу с помощью php. В изначальной позиции модальное окно не должно быть видно, поэтому в селекторе popup прописывается видимость hidden. Чтобы при нажатии он становился видимым, задаем псевдокласс :target со значением visible.

Таблица 3 – Модальное окно

|  |  |
| --- | --- |
| HTML | CSS |
| <div class="popup" id="reg\_form">  <div class="popup\_body">  <div class="popup\_content">  <a href="#" class="popup\_close">X</a>  <h4>АВТОРИЗАЦИЯ</h4>  <form method="POST" action="controllers/auth.php">  <input type="text" name="login\_auth" placeholder="Логин (эл.почта или телефон)" class="reg\_form\_input"> | .popup{position: fixed;  width: 100%;  height: 100%;  background-image: url(../images/tigers.jpg);  background-size: cover;  top: 0;  left: 0;  z-index: 2;  visibility: hidden; |

Продолжение таблицы 3

|  |  |
| --- | --- |
| <input type="password" name="password\_auth" placeholder="Пароль" class="reg\_form\_input">  <input type="submit" name="auth" value="ВОЙТИ" class="reg\_form\_input auth">  </form>  <div class="or\_reg">Или <a href="#registration">зарегестрируйтесь</a></div>  </div></div></div>  <div class="popup" id="registration">  <div class="popup\_body">  <div class="popup\_content reg">  <a href="#" class="popup\_close">X</a>  <h4>РЕГИСТРАЦИЯ</h4>  <form method="POST">  <input type="text" name="name" placeholder="Имя" class="reg\_form\_input">  <input type="text" name="surname" placeholder="Фамилия" class="reg\_form\_input">  <input type="text" name="login" placeholder="Логин (эл.почта или телефон)" class="reg\_form\_input">  <input type="password" name="password" placeholder="Пароль" class="reg\_form\_input">  <input type="password" name="r\_password" placeholder="Подтверждение пароля" class="reg\_form\_input">  <input type="submit" name="reg" value="ЗАРЕГЕСТРИРОВАТЬСЯ" class="reg\_form\_input auth">  </form>  <div class="or\_reg">Уже есть аккаунт?<a href="#reg\_form">Войдите</a></div>  </div>  </div></div> | opacity: 0%;  transition: ease-in-out 0.5s;}  .auth{width: 200px;  background-color: #63a5cf;}  .reg\_form\_input{  border-radius: 0px;  margin-top: 15px;  border: solid grey 1px;}  .or\_reg{margin-top: 15px;  transition: ease-in-out 0s;}  popup\_body{  display: flex;  min-height: 100%;  align-items: center;  justify-content: center;}  .popup\_content{  background-color: white;  width: 589px;  height: 293px;  text-align: center;  position: relative;  padding-top: 50px;}  .reg{  height: 500px;  }  .popup a:hover{  color: #63a5cf;}  .popup\_close{  position: absolute;  right: 25px;  top: 20px;}  .popup:target{visibility: visible;  opacity: 100%;} |

На модальном окне заканчивается этап написание кода HTML и стилей CSS для главной страницы и можно приступить к страницам из меню, личному кабинету, корзине и избранному.

Для меню создаем отдельную папку menu, в которых находятся папки с названиями соответствующие названиям меню на сайте, но на английском. В этих папках находятся по одному файлу с именем index.php. Также, в папке menu нет категорий и брендов, поскольку они находятся отдельно.

В разделе меню «Контакты» находится в основном только текстовая информация, а шапка и подвал взяты из главной, поэтому будет расписан только контент согласно приложению 4. В самом верху находятся так называемые хлебные крошки. Они упрощают переход между страницами.

Также, как и в контактах, в пункте меню «Доставка» будет прописан только контент (как и в последующих действиях), как показано в приложении 5. В тегах <li> указаны другие стили, т.к. они будут конфликтовать с свойством list-style-type: none в CSS.

В «Акциях» находятся изображения, а внизу страницы расположена пагинация в соответствии приложению 6, но она не будет рабочей. Ее задают при большом количестве информации. Чтобы сделать ее рабочей нужно указать ссылки на страницы.

Далее идут «Бренды». Весь основной контент брендов будет прописан с помощью php. Код страницы показан в таблице 5.

Таблица 5 – Бренды

|  |  |
| --- | --- |
| HTML | CSS |
| <main >  <div class="breadcrumbs-width">  <ul class="breadcrumb">  <li class="breadcrumb-item"><a href="../">Главная</a></li>  <li class="breadcrumb-item active\_brdcrmb">Бренды</li>  </ul>  </div> | Стили для данных элементов уже были упомянуты в Приложении 2 и Приложении 4 |

Продолжение таблицы 5

|  |  |
| --- | --- |
| <div class="top\_section">  <h1>БРЕНДЫ</h1>  <div class="our\_brands">Наш интернет-магазин является официальным дилером представленных торговых марок. Это означает, что вся продукция действительно фирменная, никакого «серого импорта», на все товары распространяется гарантия производителя, цены в нашем магазине соответствуют, рекомендованным производителем.</div>  </div>  <div class="brands\_page\_wrapper">  <div class="our\_partners">  <img src='../images/бренды/$out\_pic[picture]' style=’images’>  </div>  </div>  </main> |  |

В категориях, показанные в приложении 7, и разделах, так же как и в брендах и слайдерах, основной контент, а именно товары, будут реализованы с помощью php. Для разделов создается отдельная папка sections в папке category. Поскольку разделы и категории различаются только названиями, то соответственно их стили и наполнение похожи между собой и можно посмотреть также в приложение 7.

Нажимая на какой-либо товар мы переходим на страницу с товаром соответственно приложению 8. Для файла index.php создаем отдельную папку products в папке category.

Для добавления товара в корзину или избранное, нам нужно, чтобы это куда-то выводилось. В папке курсовая 2 курс создаем папку shopping\_cart для корзины и favorites для избранного. В них добавляем файлы index.php. На стадии HTML и CSS ничего добавляться и считаться не будет. Код корзины и избранного, усмотренный в таблице 6, будет похож между собой и отличается только текстом, а весь основной функционал реализован на php.

Таблица 6 – Избранное и корзина

|  |  |
| --- | --- |
| HTML | CSS |
| <main >  <div class="breadcrumbs-width">  <ul class="breadcrumb">  <li class="breadcrumb-item"><a href="../">Главная</a></li>  <li class="breadcrumb-item active\_brdcrmb">Корзина</li>  </ul>  </div><div class="top\_section">  <h1>КОРЗИНА</h1>  </div>  <div class="shopping\_cart\_wrapper">  <div class="your\_cart\_is\_empty">Ваша корзина пуста :(</div>  <div class="your\_cart\_is\_empty continue\_search">Продолжайте искать и может вам что-то приглянется!</div>  </div>  </div>  </main> | .your\_cart\_is\_empty{  font-size: 40px;  text-align: center;  margin-top: 50px;  padding-bottom: 70px;}  .continue\_search{font-size: 30px;}  .shopping\_cart\_wrapper{padding-bottom: 70px;  } |

После регистрации или входа пользователь будет переходить на личный кабинет, который показан в таблице 7, и в дальнейшем взаимодействовать с ним. Создаем папку personal\_account, а в ней файл index.php, папки для изменения настроек личного кабинета и вывода бонусов у пользователя (не рабочее). Как и все, что связано с взаимодействием между пользователем на сайте, основной функционал реализован на php. Раздел изменения информации, бонусы и главная похожи между собой и различаются только наполнением. Они показаны в приложении 9 и приложении 10 соответственно.

Таблица 7 – Главная страница в личном кабинете

|  |  |
| --- | --- |
| HTML | CSS |
| <main >  <div class="breadcrumbs-width">  <ul class="breadcrumb">  <li class="breadcrumb-item"><a href="../">Главная</a></li>  <li class="breadcrumb-item active\_brdcrmb">Личный кабинет</li></ul></div>  <div class="top\_section"><h1>ЛИЧНЫЙ КАБИНЕТ</h1></div>  <div class="personal\_account\_wrapper">  <ul><li class="active">Мои заказы</li>  <li><a href="personal\_account\_settings/">Настройки</a></li>  <li><a href="personal\_account\_bonuses/">Бонусы</a></li>  <li><a href="../controllers/exit.php">Выйти</a></li>  </ul>  <div class="personal\_account\_data"><h1>ПОСЛЕДНИЕ ЗАКАЗЫ</h1><div class="last\_orders\_wrapper">  <div class="last\_orders">Вы пока ничего не заказали</div></div>  <h1>ИЗБРАННОЕ</h1><div class="last\_orders\_wrapper">  <ul>  <div class="shopping\_cart\_wrapper">  <div class="shopping\_cart\_filled">  <div class="order\_info">  </div>  <div class="last\_orders">Вы пока не добавили ни одного товара в избранное</div>  </div>  </div>  </div>  </main> | .personal\_account\_wrapper{width: 1920px;  padding-bottom: 50px;}  .personal\_account\_wrapper li {font-size: 23px;  margin-top: 20px;  margin-right: 52px;}  .personal\_account\_wrapper ul{display: flex;  padding-right: 1310px;}  .personal\_account\_wrapper ul a:hover{color: black;}  .personal\_account\_data{margin-top: 40px;}  .personal\_account\_data h1{color: #63a5cf;}  .personal\_account\_wrapper .active{color: #63a5cf;  text-decoration: underline;  background-color: transparent;}  .last\_orders{font-size: 23px;  margin-top: 20px;  margin-bottom: 45px;  } |

**2.2. Работа с PHP и базой данных**

Для работы с базой данных будем использовать программу Open Server Panel.

Open Server Panel – это портативная программная среда, созданная специально для веб-разработчиков с учётом их рекомендаций и пожеланий.

Данный программный комплекс включает в себя тщательно подобранный набор серверного программного обеспечения, а также невероятно удобную и продуманную управляющую утилиту, которая обладает мощными возможностями по администрированию и настройке всех доступных компонентов.

OSPanel широко используется с целью разработки, отладки и тестирования веб-проектов, а также для предоставления веб-сервисов в локальных сетях.

Чтобы приступить к работе с базой данных следует зайти на phpMyAdmin и создать базу данных. Для этого запускаем Open Server Panel, переходим во вкладку дополнительно и ищем phpMyAdmin. После, мы заходим в личный кабинет. Оставляем имя пользователя root и пароль пустым. Далее создаем базу данных. Назовем ее scissors.

Создаем таблицы, где будут храниться наши данные: goods – для товаров, category – для категорий, sections – для разделов, brands – для брендов, orders – для заказов, comments – для комментариев, users – для пользователей. Для id в каждой таблице отмечаем чекбокс A\_I или AUTO\_INCREMENT с той целью, чтобы каждый элемент таблицы имел уникальный номер. Тип для такого столбца указывается int – это позволит вносить в столбец id только числовое значение. Для других столбцов тип указывается varchar – строка переменной длины и text для больших текстовых данных по типу описания товара. Также, в местах, где нужно указать время указывается тип timestamp со значением по умолчанию CURRENT\_TIMESTAMP. На рисунках, приведенных ниже, показаны структуры каждой таблицы.

Рисунок 1 – Таблица goods.



Рисунок 2 – Таблица category.

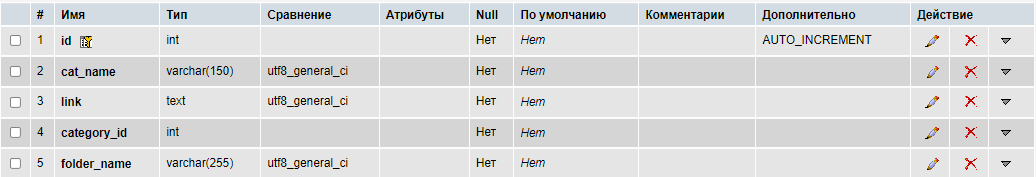


Рисунок 3 – Таблица sections.

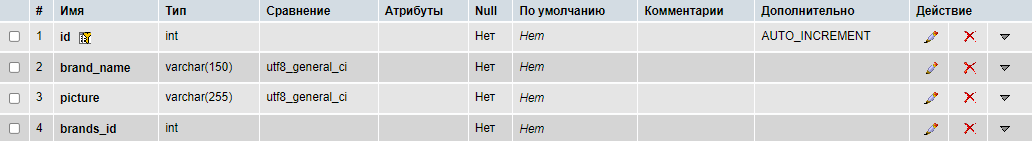
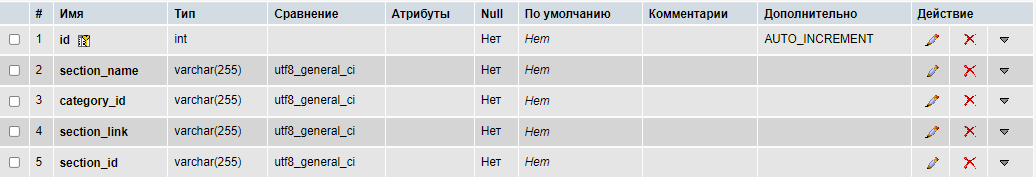


Рисунок 4 – Таблица бренды.



Рисунок 5 – Таблица orders.



Рисунок 6 – Таблица comments.



Рисунок 7 – Таблица users.

Главная страница поделена по файлам. В файле index.php подключаем шапку, контент, подвал и модальное окно с помощью include соответственно таблице 8. Тем же способом подключаем файл с подключением к базе данных, как показано в таблице 9 (далее БД). Для шапки также созданы два файла: header.php, если сессия user не начата и header2.php, если она начата.

Таблица 8 – Главная страница с подключениями к файлам

|  |
| --- |
| PHP |
| <?php  session\_start();  if ($\_SESSION['user']) {include ('header2.php');}  else{include ('header.php');}  include ('content.php');  include ('footer.php');  include\_once ('connect\_db.php');  include ('reg\_auth\_forms.php')  ?> |

Таблица 9 – Подключение к БД

|  |
| --- |
| PHP |
| <?php  $connect=mysqli\_connect('localhost', 'root', '', 'scissors'); ?> |

Одним из функционалов сайта является регистрация и вход. В форме для входа задаем атрибут action со значением пути к обработчику формы auth.php, который находится в папке controllers. В нем сравнивается, существует ли введенный пользователь в таблице users. Код приведен ниже, в таблице 9.

Таблица 9 – Обработчик формы входа

|  |
| --- |
| PHP |
| <?php  session\_start();  include\_once '../connect\_db.php';  $login\_auth=$\_POST['login\_auth'];  $password\_auth=$\_POST['password\_auth'];  $auth=$\_POST['auth']; $password\_auth=md5($password\_auth);  $str\_auth="SELECT \* FROM `users` WHERE login='$login\_auth' && password='$password\_auth'";  $run\_auth=mysqli\_query($connect, $str\_auth);  $count\_user=mysqli\_num\_rows($run\_auth);  $found\_user=mysqli\_fetch\_array($run\_auth);  if ($count\_user>0) {  echo "$found\_user[surname] $found\_user[name], Вы авторизовались";  $\_SESSION['user']=[  "login"=>$found\_user['login'],  "surname"=>$found\_user['surname'], "name"=>$found\_user['name'],  "id"=>$found\_user['id'],  "role"=>$found\_user['role']];  if ($\_SESSION['user']['role']==1) {header("Location:../admin/");  }else{header("Location:../personal\_account/");}  }else{$\_SESSION['auth\_error']="Не верный логин или пароль"; header("Location:../#reg\_form");}  ?> |

Для формы регистрации код представлен в приложении 11.

В меню есть каталог, в котором осуществлен вывод категорий. В ссылке приравниваем category\_id из ассоциативного массива к переменной id\_category, которую выводим на странице для категорий как показано в таблице 10. Переменные в ссылках прописываются через знак «?».

Таблица 10 – Вывод категорий в каталоге

|  |
| --- |
| PHP |
| <?php  $str\_out\_category="SELECT \* FROM `category`";  $run\_out\_category=mysqli\_query($connect, $str\_out\_category);  if ($run\_out\_category) {  while ($out=mysqli\_fetch\_array($run\_out\_category)) {  echo "<li><a href='category/?id\_category=$out[category\_id] '> $out[cat\_name]</a></li>";}  }  ?> |

В начале страницы с категориями создаем ассоциативный массив для вывода названия категории в соответствии с таблицей 11.

Таблица 11 – Вывод названия категории

|  |
| --- |
| PHP |
| <?php  session\_start();  include\_once ('../connect\_db.php');  $id\_category=$\_GET['id\_category'];  $str\_out\_category\_name="SELECT \* FROM `category` WHERE id='$id\_category'";  $run\_out\_category\_name=mysqli\_query($connect, $str\_out\_category\_name);  $out\_name=mysqli\_fetch\_array($run\_out\_category\_name);  ?> |

Пример того, как будет использован этот массив представлен в таблице 12.

Таблица 12 – Пример использования ассоциативного массива

|  |
| --- |
| PHP |
| <title>Товары <?php echo "$out\_name[cat\_name]"; ?> интернет-магазин Ножницы</title> |

Далее, на странице с категориями в контенте выводим из таблицы с товарами, где значение из таблицы с категориями будет равно значению категории в товарах. Устанавливаем лимит на вывод в размере 15 усмотренный в таблице 13. Похожая ситуация с разделами в приложении 12.

Таблица 13 – Вывод товаров, в соответствии с номером категории

|  |
| --- |
| PHP |
| <?php  $str\_out\_goods="SELECT \* FROM `goods` WHERE category='$id\_category' AND id > 5 LIMIT 15";  $run\_out\_goods=mysqli\_query($connect, $str\_out\_goods);  while ($out\_goods=mysqli\_fetch\_array($run\_out\_goods)) {echo "<div class='product on\_main'>  <img src='../images/товары/$out\_goods[folder\_name]/$out\_goods[product\_pic]' class='image image\_on\_main'><div class='product\_info product\_info\_on\_main'>  <a href='../category/products/?id\_goods=$out\_goods[id] $out\_goods[product\_name]' ><div class=product\_title>$out\_goods[product\_name] </div></a><div class=product\_footer>  <div class=product\_price>$out\_goods[price] Р</div>  <a href=../shopping\_cart/?id\_product=$out\_goods[id]><div class='shopping\_cart buy\_product'></div></a>  <a href=../favorites/?id\_product\_fav=$out\_goods[id]><div class='favorites fav\_product'></div><a/></div></div></div>  ";}  ?> |

Так же, как и на странице категорий, на странице с разделом сравниваем название раздела со значениями в таблицах в соответствии с выбранным разделом. Для файла раздела в папке category создается папка section, а в ней index.php.

На странице товаров ситуация аналогична вышесказанному. Создаются ассоциативные массивы в соответствии с выбранным товаром. Для отзывов, которые показаны в приложении 13, как и в случае с личным кабинетом, выводится сообщение о невозможности оставлять отзывы, если $\_SESSION[‘user’] не начата. В условии проверяется наличие данных в текстовом поле и нажатие на кнопку.

Нажатие на кнопку корзины перенаправляет на страницу корзины. Первым делом создадим в отдельном файле functions в папке controllers функцию get\_goods\_by\_id как показано в приложении 14. При вызове этой функции, будет передаваться id товара из массива $\_GET в эту функцию, а уже в самой функции выбираться товары из таблицы и выводиться на страницу. Для вывода товаров на саму страницу начнем сессию cart\_list, к которой приравняем заданную функцию и делаем проверки. При помощи unset удаляем товары из корзины как показано в приложении 15. Действия с избранными товарами аналогичны корзине, меняются только названия переменных.

Сама корзина оформлена с помощью цикла foreach как в приложении 17. После нажатия на кнопку запускается обработчик order.php, который приведен в таблице 14.

Таблица 14 – Обработчик корзины

|  |
| --- |
| PHP |
| <?php  session\_start();  include\_once('../connect\_db.php');  $ammount=$\_GET['ammount']; $count= $\_POST['count'];  foreach($\_SESSION['cart\_list'] as $goods ){ $product\_name .=$goods['product\_name'];  $\_SESSION['count']=["count"=>$\_POST['count'], "ammount"=>$\_GET['ammount'], "product\_name"=>$product\_name];}  header("Location: ../shopping\_cart/order/");  ?> |

Из обработчика идет перенаправление на страницу с оформлением заказа, код которой представлен в приложении 18, а из него, в зависимости от выполнения условия, на страницу с благодарностью о заказе.

Для добавления товаров в БД создадим отдельную форму add\_products.php в папке admin. В форме будет добавление изображений и для этого в теге form зададим атрибут enctype со значением multipart/form-data как показано в приложении 16. В выпадающее меню выведем названия и в атрибут value их id. Чтобы добавить изображение нужно в тег input прописать тип file. Из изображения занесем только его имя.

В разделе с брендами выводятся изображения из базы данных с помощью цикла. Для их занесения туда также была создана форма как в приложении 20.

Личный кабинет оснащен выходом из аккаунта. Для это создается обработчик выхода exit.php. Он удаляет сессию для авторизации пользователя и перенаправляет на главную страницу как показано в таблице 17.

Таблица 17 – Выход из аккаунта

|  |
| --- |
| PHP |
| <?php  session\_start();  unset($\_SESSION['user']); unset($\_SESSION['auth\_error']); unset($\_SESSION['reg\_error']);  header("Location:../")  ?> |

Для более удобной навигации по сайту на нем был сделан поиск, который усмотрен в приложении 19. Работает он следующим образом: на отдельной странице с выводом результата из таблицы с товарами выбираются значения, похожие на введенное в поле поиска.

**Заключение**

Разные языки веб-программирования позволяют достичь разных целей. Но объединив их мы получим сайт, соответствующий стандартным требованиям клиента. В настоящее время существует множество типов сайтов: от простых персональных сайтов до веб-приложений. Но всех их объединяет одно – взаимодействие с пользователем.

При написании курсовой была изучена различная литература по языкам веб-программирования, а также информационные сайты в сети Интернет. Результатом работы над практической частью стал интернет-магазин с минимальным набором возможностей, написанный с использованием HTML, CSS и PHP.

На основе всей проделанной работе были сформированы следующие выводы:

1. Способ создания слайдера, который применен в этой работе, не является наилучшим, поскольку в адресной строке остается хэш. Также, сам скролл уходит вниз, а элемент слайда уходит наверх;
2. Тестирование посторонними помогает посмотреть на работу под другим углом;
3. При добавлении товара в корзину мы переходим в корзину, но обычно делают так, что выходит модальное окно с успешным добавлением. К сожалению, в силу недостатка знаний, это не было реализовано;
4. Ранняя версия меню категорий имела не слишком удобную систему папок. Достаточно было делать один файл для категорий и разделов;
5. Поскольку изменялись системы папок и файлов, приходилось по несколько раз переписывать пути к файлам на страницу.

Несмотря на все проблемы, которые встретились при создании сайта, работа была успешно выполнена и сайт можно считать вполне рабочей системой для оформления заказа.

**Список использованных источников**

1. Макфарланд Д. Большая книга CSS3 3-е изд. – СПб.: Питер, 2014. – 608 с.: ил. – (Серия «Бестселлеры O’Reilly»).
2. Anne B., Zak R. Murach’s HTML5 and CSS3 3rd ed. Fresno, CA: Mike Murach & Associates, Inc., 2015 – 657 p.
3. Jamie C. Learn PHP in One Day and Learn It Well. PHP for Beginners with Hands-on Project Vol. 6 – Lcf Publishing, 2020. – 209 p.
4. Joe A. Modern CSS: Master the Key Concepts of CSS for Modern Web Development. Billerica, MA, USA: Apress, 2020. – 312 p.
5. Введение в HTML. Информатика. Учебное пособие :URL: http://psk68.ru/files/metod/uchebnik\_Informatika/vvhtml.html
6. Учебник по PHP. HTML Academy. Основы языка запросов SQL :URL: https://htmlacademy.ru/tutorial/php/sql
7. Учебник по PHP. HTML Academy. Определение и особенности языка PHP :URL: https://htmlacademy.ru/tutorial/php/basics
8. Open Server. Локальный веб-сервер для Windows :URL: https://ospanel.io/
9. php.net. Manual. Что такое PHP? :URL: https://www.php.net/manual/ru/intro-whatis.php
10. phpMyAdmin по-русски. Главная страница :URL: https://php-myadmin.ru/
11. REG.RU. Помощь. Что такое MySQL :URL: https://help.reg.ru/hc/ru/articles/440804776321-Что-такое-MySQL
12. Wikipedia – свободная энциклопедия :URL: https://ru.wikipedia.org/wiki/