**МИНОБРНАУКИ РОССИИ**

**Федеральное государственное бюджетное**

**образовательное учреждение высшего образования**

**«Тульский государственный университет»**

Подразделение кафедра вычислительной техники

*(наименование подразделения)*

**ОТЧЕТ ПО ПРАКТИКЕ**

|  |  |
| --- | --- |
| Вид (тип) практики | Учебная (технологическая (проектно-технологическая))\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Курс | 2\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Направление подготовки  /специальность | 09.03.01. Информатика и вычислительная техника \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Ф.И.О. обучающегося | Картаков Вячеслав Валерьевич\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Место прохождения практики | г. Тула, ООО «Максимастер»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Период прохождения практики | 20.02.2023 – 07.06.2023\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

Руководитель практической подготовки

(руководитель практики) от профильной организации

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(Ф.И.О., должность) (подпись)*

М. П.

Руководитель практической подготовки

(руководитель практики) от подразделения

Антонов Д. М., доц. каф. ВТ\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(Ф.И.О., должность) (подпись)*

Тула 2023 г.

**РЕФЕРАТ**

Отчет 1 кн., 24 с., 12 рис., 6 ист.

САЙТ, ФРЕЙМВОРК, АДАПТИВНАЯ ВЁРСТКА, ЯЗЫК ПРОГРАММИРОВАНИЯ, БЫЗА ДАННЫХ, HTML, CSS, DJANGO, МАКЕТ.

Объект исследования: веб-сайт.

Предмет исследования: интернет-магазин.

Цель работы - сверстать адаптивный шаблон сайта (интернет-магазина) из PSD-макета. В макетах должно быть минимум четыре страницы:

* главная страница;
* страница со списком товаров (каталог);
* страница с подробной информацией о товаре;
* страница с веб-формой (например форма обратной связи);

В ходе выполнения практической работы получены следующие результаты:

Изучены теоретические сведения о сайтах;

Разработан макет сайта;

Рассмотрены и изучены HTML, CSS;

Изучен фреймворк Django.

**СОДЕРЖАНИЕ**

[ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ 4](#_Toc136268923)

[ВВЕДЕНИЕ 5](#_Toc136268924)

[1 Основы создания сайта. 6](#_Toc136268925)

[2 Выбор макета сайта. 8](#_Toc136268939)

[3 Вёрстка макета. 8](#_Toc136268944)

[3.1 Работа с макетом 9](#_Toc136268948)

[3.2 Написание HTML-кода 9](#_Toc136268955)

[3.3 Стилизация 12](#_Toc136268972)

[4 Реализация серверной части. 14](#_Toc136268988)

[4.1 Базы данных 15](#_Toc136268991)

[4.2 Генерация URL-адресов 16](#_Toc136269001)

[4.3 Шаблонизация 17](#_Toc136269009)

[ЗАКЛЮЧЕНИЕ 19](#_Toc136269013)

[СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 20](#_Toc136269014)

[ПРИЛОЖЕНИЯ 21](#_Toc136269015)

[ПРИЛОЖЕНИЕ А 21](#_Toc136269016)

# **ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ**

Веб - система доступа к связанным между собой документам на различных компьютерах, подключённых к Интернету.

Тег - идентификатор для категоризации, описания, поиска данных и задания внутренней структуры.

Сервер - выделенный или [специализированный](https://ru.wikipedia.org/wiki/%D0%A1%D0%BF%D0%B5%D1%86%D0%B8%D0%B0%D0%BB%D0%B8%D0%B7%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D0%BC%D0%B0%D1%88%D0%B8%D0%BD%D0%B0) [компьютер](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80) для выполнения сервисного программного обеспечения (в том числе [серверов](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%B5%D1%80_(%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5)) тех или иных задач).

Трафик – количество посетителей, посещающих сайт за определённый период времени.

Медиа запрос - это функция CSS 3, позволяющая рендерингу контента адаптироваться к различным условиям, таким как разрешение экрана.

Фреймворк - программная платформа, определяющая структуру программной системы; [программное обеспечение](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5), облегчающее разработку и объединение разных компонентов большого программного проекта.

Ссылка - запись, которая [идентифицирует](https://ru.wikipedia.org/wiki/%D0%98%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F_(%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D1%8B%D0%B5_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D1%8B)) [документ](https://ru.wikipedia.org/wiki/%D0%94%D0%BE%D0%BA%D1%83%D0%BC%D0%B5%D0%BD%D1%82) или его часть.

# **ВВЕДЕНИЕ**

Актуальность темы:

В наше время, когда все больше людей обращаются к интернету для получения информации и услуг, создание сайтов становится необходимостью для многих компаний и организаций. Сайты позволяют увеличить видимость бренда, привлечь новых клиентов и улучшить коммуникацию с уже существующими. Кроме того, сайты могут быть использованы для продажи товаров и услуг, что делает их важным инструментом для бизнеса.

Для веб-разработчиков и дизайнеров создание сайтов также является выгодным бизнесом, поскольку спрос на услуги по созданию сайтов по-прежнему остается высоким. Кроме того, создание сайтов может быть интересным и творческим процессом, который требует разнообразных навыков и знаний.

В целом, создание сайтов остается важным и актуальным процессом в нашей цифровой эпохе. Он предоставляет множество возможностей для бизнеса, творчества и социальной жизни, и требует постоянного развития и улучшения.

Цель работы - сверстать адаптивный шаблон сайта (интернет-магазина) из PSD-макета. В макетах должно быть минимум четыре страницы:

* главная страница;
* страница со списком товаров (каталог);
* страница с подробной информацией о товаре;
* страница с веб-формой (например форма обратной связи);

Задачи работы:

1. изучение теоретической части по разработке сайтов;
2. разработка макета сайта;
3. вёрстка шаблона сайта;
4. разработка серверной части сайта.
5. **Основы создания сайта.**

Сайт - это набор веб-страниц, связанных между собой гиперссылками и размещенных на одном домене в интернете. Сайт может содержать информацию о компании, продуктах и услугах, а также предоставлять пользователю возможность взаимодействия с компанией, например, заказывать товары или услуги.

Основной язык программирования, который используется при создании сайта – HTML (это язык разметки, который используется для создания веб-страниц. Он состоит из набора тегов, которые определяют структуру и содержание веб-страницы. HTML позволяет создавать текст, изображения, ссылки, таблицы, формы и другие элементы веб-страницы. Браузеры интерпретируют HTML-код и отображают его в виде веб-страницы для пользователя.).

Данный язык является довольно самодостаточным и сайт, написанный только на нём, уже может функционировать. В соответствии с рисунком 1



Рисунок 1 – Сайт на чистом HTML

Минус сайта на чистом HTML заключается в том, что он не обладает динамическими функциями, такими как интерактивность и возможность обработки данных на стороне сервера. Также, сайт на чистом HTML может быть более сложным для поддержки и обновления, особенно если он содержит много страниц.

Помимо HTML используется такой язык как CSS - это язык стилей, который используется для оформления веб-страниц. Он позволяет определять внешний вид элементов HTML, таких как цвет, шрифт, размер и расположение. CSS позволяет разделять содержание и оформление веб-страницы, что упрощает ее редактирование и обновление. Браузеры интерпретируют CSS-код и применяют его к HTML-элементам для создания красивого и современного дизайна веб-страницы.

Сайт в котором, помимо HTML, присутствует ещё и CSS (в соответствии с рисунком 2), является более визуально привлекательным и интуитивно понятным для пользователя, благодаря чему трафик сайта будет увеличиваться.

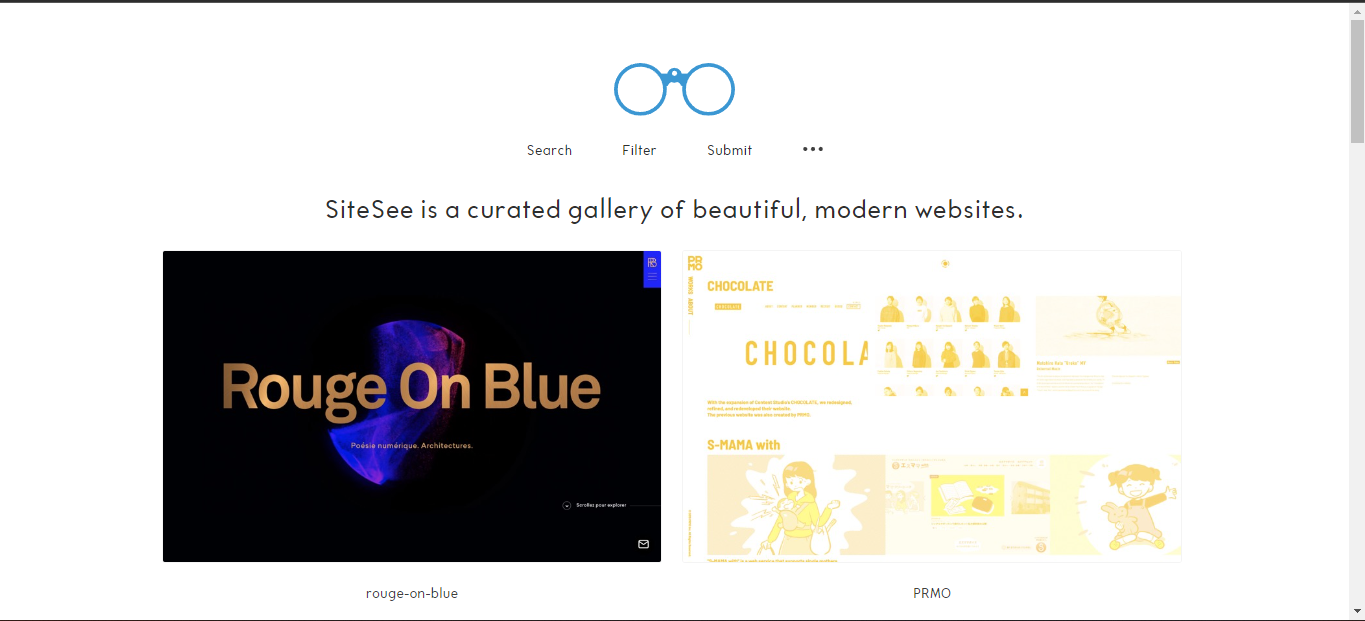


Рисунок 2 – Сайт с использованием CSS

При использовании данных языков программирования можно уже создать довольно красивый и функциональный сайт, который может выполнять информационную функцию, но для полноценной работоспособности ему будет не хватать серверной части, которая обеспечит взаимодействие с клиентом.

Существует довольно обильное количество языков программирования, позволяющих реализовать это, среди которых – Ruby, PHP, Node.JS, Python.

С их использованием появляется возможность внедрения баз данных, что упростит работу по изменению содержимого страниц сайта и не будет требовать вмешательства в код проекта, а также появится возможность организации системы пользователя, что для многих типов сайтов является неотъемлемой функцией.

1. **Выбор макета сайта.**

Макет сайта - это графическое представление внешнего вида и структуры веб-страницы. Он содержит информацию о расположении элементов на странице, их размерах, цветах и шрифтах. Макет сайта может быть создан в графических редакторах, таких как Adobe Photoshop или Sketch, и является основой для верстки веб-страницы с использованием HTML и CSS. Макет сайта позволяет визуализировать конечный результат и упрощает процесс разработки и дизайна веб-сайта.

В настоящее время существует огромное количество ресурсов, предоставляющих макеты бесплатно, но в связи с тем, что данная тема (интернет-магазин) не является распространённой, то количество предлагаемых макетов не велико, а в предлагаемых не имеются требуемые в задании страницы, было принято решение разработать макет самостоятельно.

Для его создания был использован графический редактор, ориентированный конкретно под сайты – Mobirise.

Конечный макет сайта приведён в Приложении А.

1. **Вёрстка макета.**

Верстка сайта - это процесс создания веб-страницы с использованием HTML и CSS.

Она включает в себя размещение контента на странице, установку шрифтов, цветов, изображений и других элементов дизайна. Верстка также отвечает за адаптивность сайта под различные устройства и экраны, такие как компьютеры, планшеты и мобильные телефоны.

Цель верстки - создать эффективный и красивый дизайн, который будет удобен для пользователей и соответствовать бренду компании.

1. **Работа с макетом**

Подготовка макета к верстке происходит разбиением основной «картины» на отдельные элементы, таким образом, чтобы все компоненты были доступны для скачивания и последующего размещения на сайте, данный пункт касается различных фонов, картинок, иконок.

Структуру сайта можно разбить на 3 основных компонента: header, main, footer. В соответствии с рисунком 3.

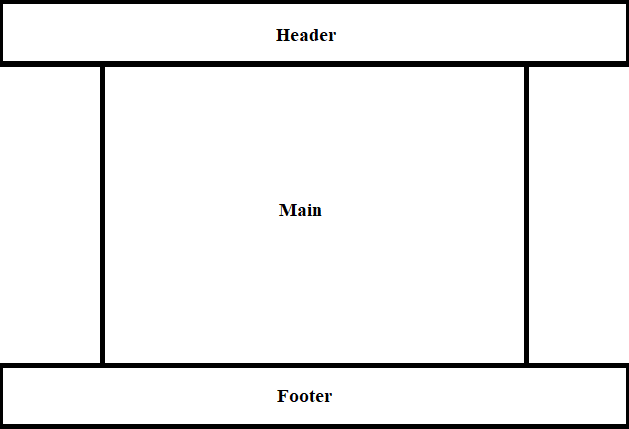


Рисунок 3 - Структура сайта

Секции header и footer для каждой страницы являются одинаковыми и зачастую содержат следующие компоненты:

* логотип и название компании;
* ссылки на другие страницы сайта;
* информация о дате создании сайта и авторе.

А вот секция main отвечает за содержание страницы и для каждой является индивидуальной. Она так же имеет собственную структуру, может разбиваться на отдельные строки и столбы, но единой структуры не имеет.

1. **Написание HTML-кода**

Как было сказано в пункте 1, в качестве «скелета» страницы используется язык HTML, который в свою очередь так же обладает собственной структурой. В соответствии с рисунком 4.

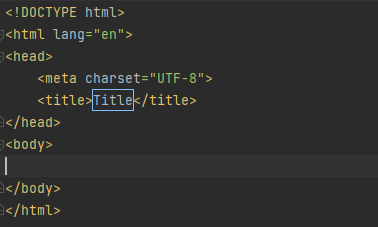


Рисунок 4 – Структура HTML.

Элемент <!DOCTYPE> предназначен для указания типа текущего документа, в нашем случае HTML.

Тег <html> определяет начало HTML-файла, внутри него хранится заголовок (<head>) и тело документа (<body>).

Заголовок документа, как еще называют блок <head>, может содержать текст и теги, но содержимое этого раздела не показывается напрямую на странице, за исключением контейнера <title>.

Тег <meta> является универсальным и добавляет целый класс возможностей, в частности, с помощью метатегов, как обобщенно называют этот тег, можно изменять кодировку страницы, добавлять ключевые слова, описание документа и многое другое.

Тег <title> определяет заголовок веб-страницы, это один из важных элементов предназначенный для решения множества задач.

Тело документа <body> предназначено для размещения тегов и содержательной части веб-страницы.

Содержимое тега head на данном этапе не заполняется, поскольку в основном он отвечает за стилизацию компонентов сайта.

Для добавления содержимого на страницу сайта, нужно прописать определённые теги в теге body, каждый тег, в свою очередь, обладает собственной функцией и отвечает за различные типы элементов, зачастую при создании используются следующие:

* [<div>](https://developer.mozilla.org/ru/docs/Web/HTML/Element/div) - является универсальным контейнером для [потокового контента](https://developer.mozilla.org/ru/docs/Web/HTML/Content_categories#%d0%9f%d0%be%d1%82%d0%be%d0%ba%d0%be%d0%b2%d1%8b%d0%b9_%d0%ba%d0%be%d0%bd%d1%82%d0%b5%d0%bd%d1%82). Он не влияет на контент или макет до тех пор, пока не будет стилизован;
* [<hr>](https://developer.mozilla.org/ru/docs/Web/HTML/Element/hr) - служит для тематического разделения абзацев. Рисует горизонтальную прямую;
* [<p>](https://developer.mozilla.org/ru/docs/Web/HTML/Element/p) - представляет собой абзац;
* [<li>](https://developer.mozilla.org/ru/docs/Web/HTML/Element/li) - используется для создания элементов списка;
* [<ul>](https://developer.mozilla.org/ru/docs/Web/HTML/Element/ul) - используется для неупорядоченного списка - в частности для маркированного списка;
* [<a>](https://developer.mozilla.org/ru/docs/Web/HTML/Element/a) - определяет гиперссылку для перехода на определённое место на странице или на другую страницу в Интернете;
* [<img>](https://developer.mozilla.org/ru/docs/Web/HTML/Element/img) - встраивает изображение в документ;
* [<script>](https://developer.mozilla.org/ru/docs/Web/HTML/Element/script) - используется для встраивания или подключения исполняемого JavaScript кода;
* [<button>](https://developer.mozilla.org/ru/docs/Web/HTML/Element/button) - создаёт интерактивную кнопку, которая может быть использована в формах или в любом другом месте документа;
* [<form>](https://developer.mozilla.org/ru/docs/Web/HTML/Element/form) - представляет (собой) раздел документа, содержащий интерактивные элементы управления, которые позволяют пользователю отправлять информацию на веб-сервер;
* [<input>](https://developer.mozilla.org/ru/docs/Web/HTML/Element/input) - используется для создания интерактивных элементов управления в веб-формах;

В соответствии с данными тегами, уже можно сделать «набросок» сайта, прописав присутствующие элементы макета в коде.

После заполнения тега body, сайт выглядит следующим образом, в соответствии с рисунком 5.

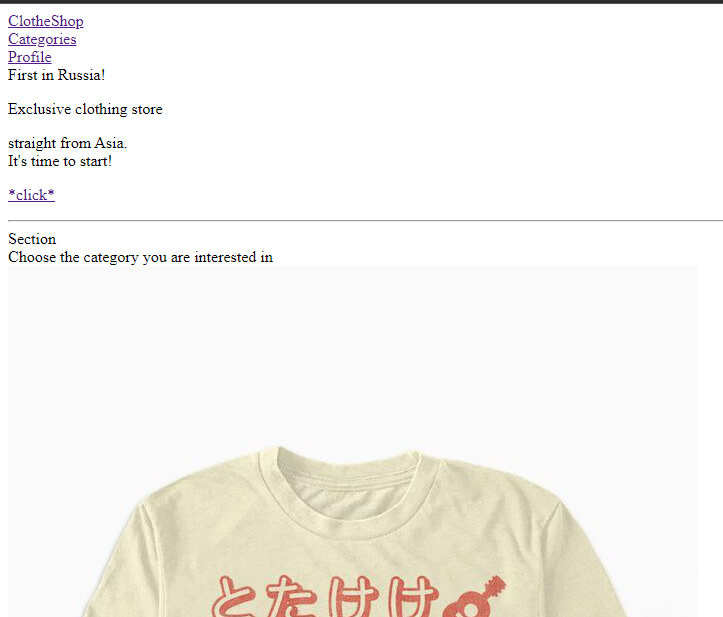


Рисунок 5 – Версия сайта на HTML.

HTML-код страниц приведён в Приложении Б.

1. Стилизация

Для взаимодействия CSS кода с основной страницей, требуется «подключить» его. Выполнить это можно, прописав в теге head, следующий код: <link rel="stylesheet" href="">, где в поле href следует прописать путь к файлу.

После подключения CSS файла, для стилизации конкретного элемента требует к нему обратиться.

Для этого можно:

Обратиться к имени элементу (body{}, div{})

Обратиться к классу элемента (.container{}, .name{}). Класс элемента прописывается в его теге, в виде: <div class=”container”>…</div>.

Обратиться по id элемента (#container{}, #name{}). Id элемента прописывается в его теге, в виде: <div id=”container”>…</div>.

Для некоторых элементов, в ходе работы будут использоваться готовые стили Bootstap.

При стилизации «контейнеров» в основном используются следующие стили:

* width – отвечает за ширину «контейнера»;
* height – отвечает за высоту «контейнера»;
* margin – задаёт внешние отступы;
* padding – задаёт внутренние отступы;
* background-color – задаёт цвет фона;
* background-image – замещает фон картинкой.

Для стилизации текста:

* text-align – расположение текста (по левому краю, по центру, по правому краю);
* color – задаёт цвет текста;
* font-size – задаёт размер текста;
* font-family – задаёт шрифт;
* text-decoration – отвечает за стиль текста (подчёркнутый, курсивный, «жирный»).

После добавления стилей, главная страница стала выглядеть следующим образом, в соответствии с рисунком 6.



Рисунок 6 – Готовая вёрстка.

Адаптивный сайт - это сайт, который автоматически подстраивается под различные размеры экранов устройств, на которых он просматривается (например, компьютер, планшет, смартфон). Адаптивный дизайн позволяет улучшить пользовательский опыт и удобство использования сайта на любом устройстве, а также повысить его поисковую видимость.

Для того чтобы адаптировать сайт под различные устройства существуют медиа-запросы: @media (min-width: 490px) and (max-width: 990px){}

В качестве параметров подаётся максимальный или минимальный размер страницы, в случае если экран устройства попадает под заданные размеры, то стили страницы отображаются в соответствии тех, которые указаны внутри запроса.

CSS-код страниц приведён в Приложении В.

1. **Реализация серверной части.**

В качестве серверного языка был выбран – Python, в связи с его простотой и универсальностью, а именно его фреймворк – Django.

Этот высокоуровневый веб-фреймворк позволяет быстро и эффективно создавать веб-приложения. Он включает в себя множество инструментов и библиотек для работы с базами данных, обработки форм, аутентификации пользователей, генерации URL-адресов и многое другое. Django также обеспечивает безопасность приложения и управление сессиями. Он широко используется в различных проектах, от простых блогов до сложных корпоративных систем.

1. **Базы данных**

В данном фреймворке реализована основная функция – взаимодействие с базами данных. Одно из удобств именно этого языка – встроенные базы данных в проект.

Работы с ними происходит при помощи встроенной панели администрации в соответствии с рисунком 7.

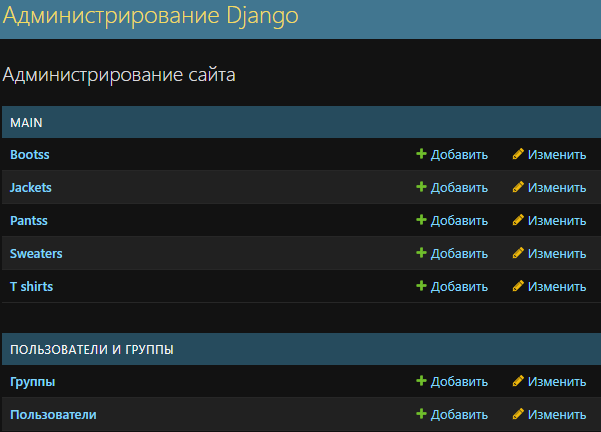


Рисунок 7 – Панель администратора

В секции MAIN располагаются созданные таблицы, в данном случае – таблицы, содержащие информацию о товарах. Структура таблиц в соответствии с рисунком 8.



Рисунок 8 – Структура таблицы

Секция Пользователи и Группы отвечает за посетителей сайта. Группы позволяют разбивать их по правам доступа, например: Администратор, Редактор, Посетитель.

Базы данных во многом упрощают работу с содержимым страницы. Появляется возможность хранения всех товаров, а также данных о пользователях в удобно для чтения и изменения данных.

1. **Генерация URL-адресов**

При создании многостраничных сайтов, на которых на одних и тех же страницах может отображаться различный контент, в зависимости от типа, используются параметры в ссылках.

При переходе на определённую страницу, браузеру передаётся помимо ссылки, ещё и её параметры, взаимодействую с данными параметрами появляется возможность работы с отображаемым контентом

Прежде всего требуется задать пути к файлам, для корректного отображения страниц. Сделать это можно, прописав следующий код в файле urls.py:

urlpatterns = [  
 path('', views.index),  
 path('prodList/', views.prodList),  
 path('product/', views.product),  
 path('profile/', views.profile),  
]

Данная секция отвечает за пути к страницам присутствующим на сайте, данные для этого берутся уже из другого файла – views.py:

def index(request):  
 return render(request, 'main/index.html')  
 Данная функция передаёт путь к файлу страницы, который будет после отображаться.  
def product(request):  
 type = request.GET.get("type")  
 id\_ = request.GET.get("id")  
 if int(type) == 1:  
 prod = TShirt.objects.filter(id=id\_)  
 elif int(type) == 2:  
 prod = Sweater.objects.filter(id=id\_)  
 elif int(type) == 3:  
 prod = Jacket.objects.filter(id=id\_)  
 elif int(type) == 4:  
 prod = Pants.objects.filter(id=id\_)  
 else:  
 prod = Boots.objects.filter(id=id\_)  
 return render(request, 'main/product.html', {'prod': prod})

В данном примере помимо ссылки, так же передаются и её параметры, при помощи которых в последствии происходит работа с БД, что упрощает создание сайта, и позволяет не создавать отдельные страницы под каждый товар.

1. **Шаблонизация**

Данный фреймворк позволяет создавать определённые шаблоны, которые можно использовать в других различных фалах, не прописывая один и тот же код множество раз.

Данная функция позволяет реализовать такие компоненты как: header и footer, поскольку на различных страницах их содержания будет одинаковым.

Сделать это можно, поместив HTML код в следующий код: {% block content %} … {% endblock %}, а вызвать его на нужной странице при помощи команды - {% extends 'main/layout.html' %}.

# **ЗАКЛЮЧЕНИЕ**

Таким образом, в данной работе был создан адаптивный дизайн, который позволяет пользователям удобно просматривать сайт на любом устройстве. Это значительно повышает удобство использования сайта и улучшает пользовательский опыт. Кроме того, адаптивность сайта положительно влияет на его поисковую видимость, что является важным фактором для привлечения новых пользователей.

# **СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ**

1. Основы CSS: [Электронный ресурс]. URL: https://developer.mozilla.org/ru/docs/Learn/Getting\_started\_with\_the\_web/CSS\_basics. (Дата обращения 25.02.2023).
2. Основы HTML: [Электронный ресурс]. URL: https://developer.mozilla.org/ru/docs/Learn/Getting\_started\_with\_the\_web/HTML\_basics. (Дата обращения: 27.02.2023).
3. Python: [Электронный ресурс]. URL: https://www.python.org/. (Дата обращения: 17.03.2023).
4. Веб-фреймворк Django (Python): [Электронный ресурс]. URL: https://developer.mozilla.org/ru/docs/Learn/Server-side/Django. (Дата обращения 12.03.2023).
5. Django: [Электронный ресурс]. URL: https://www.djangoproject.com/ (Дата обращения 12.03.2023).
6. Адаптивная вёрстка: что это и как использовать: [Электронный ресурс]. URL: https://tproger.ru/translations/responsive-web-design-tips/ (Дата обращения 2.04.2023).

**ПРИЛОЖЕНИЯ**

**ПРИЛОЖЕНИЕ А**

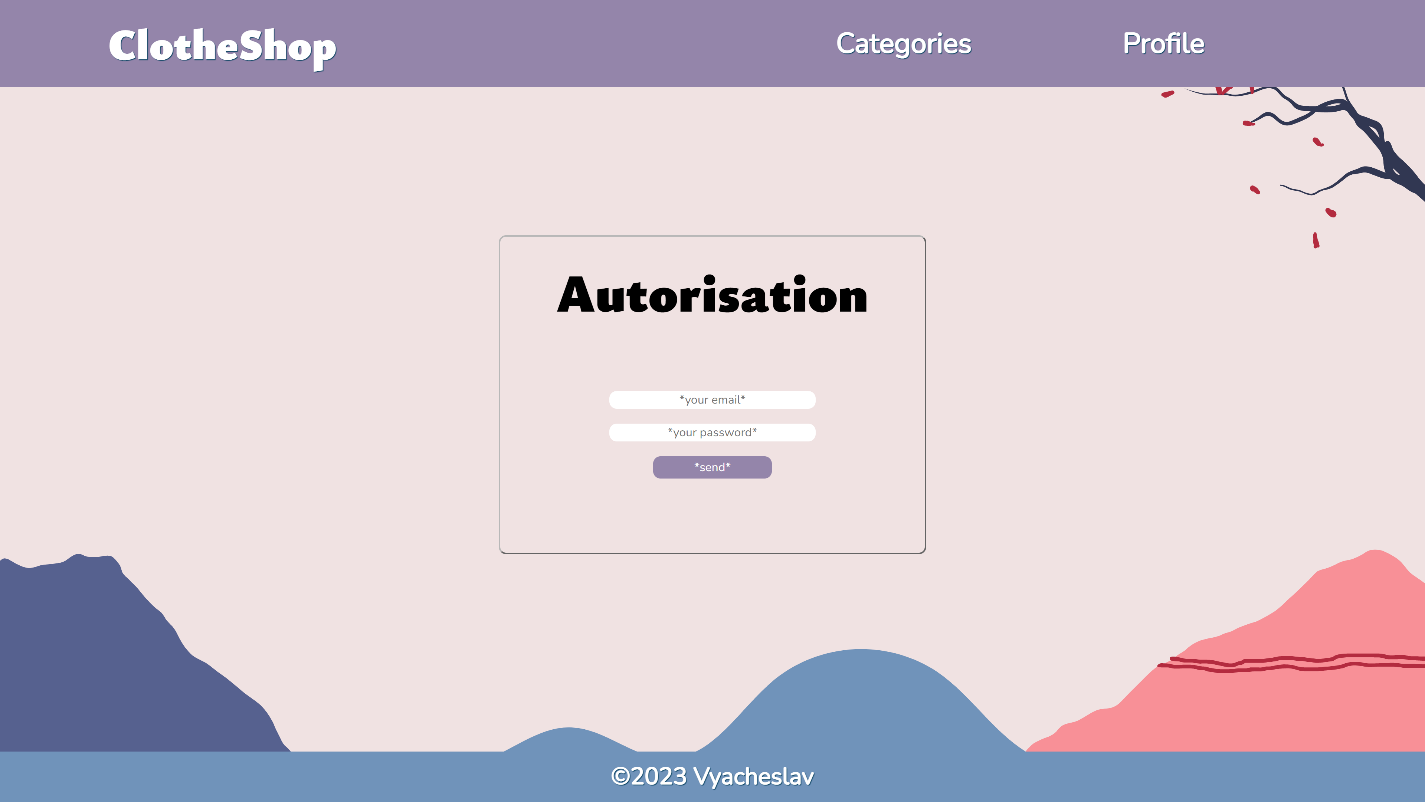


Рисунок 9 – Страница авторизации

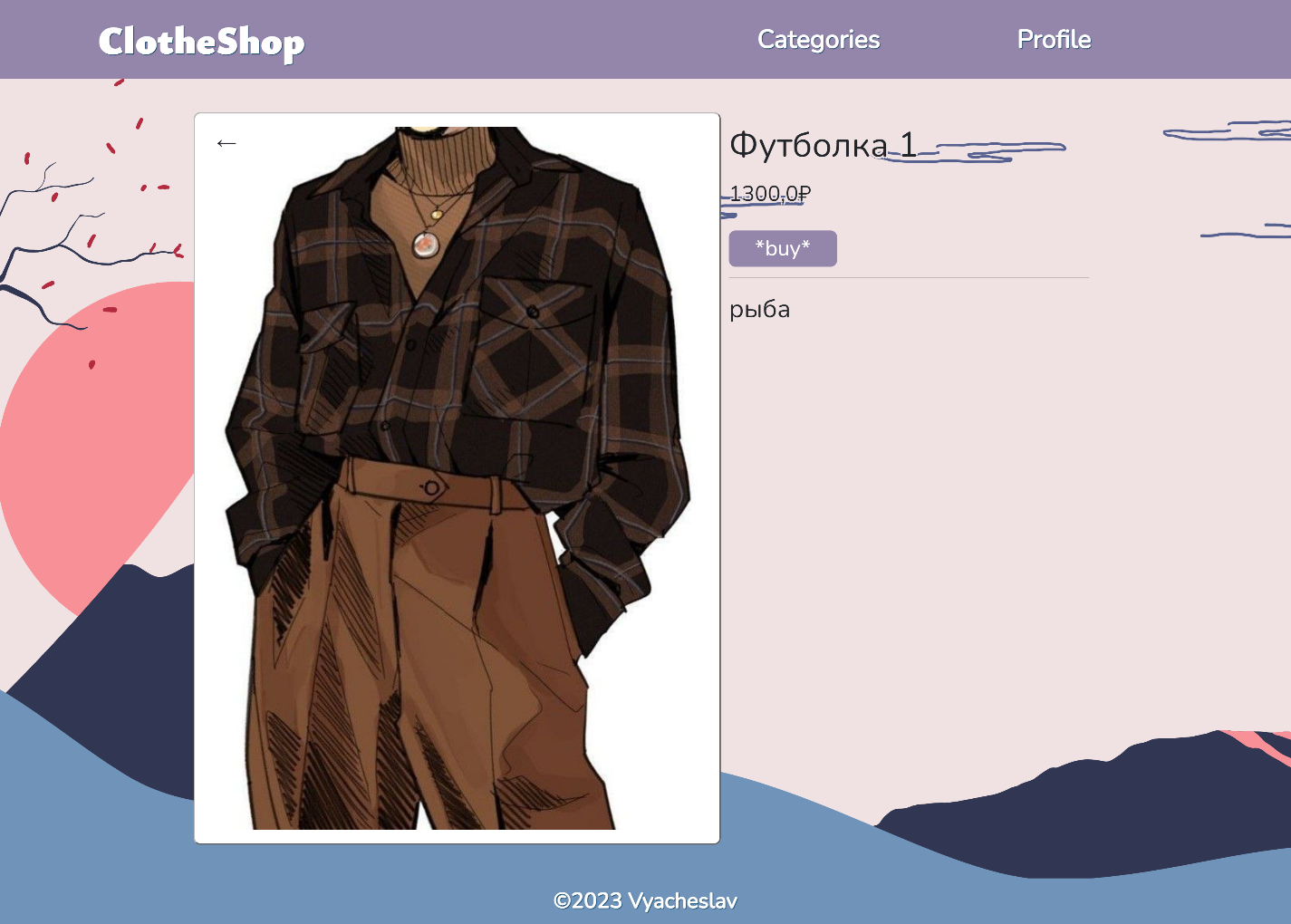


Рисунок 10 – Страница товара

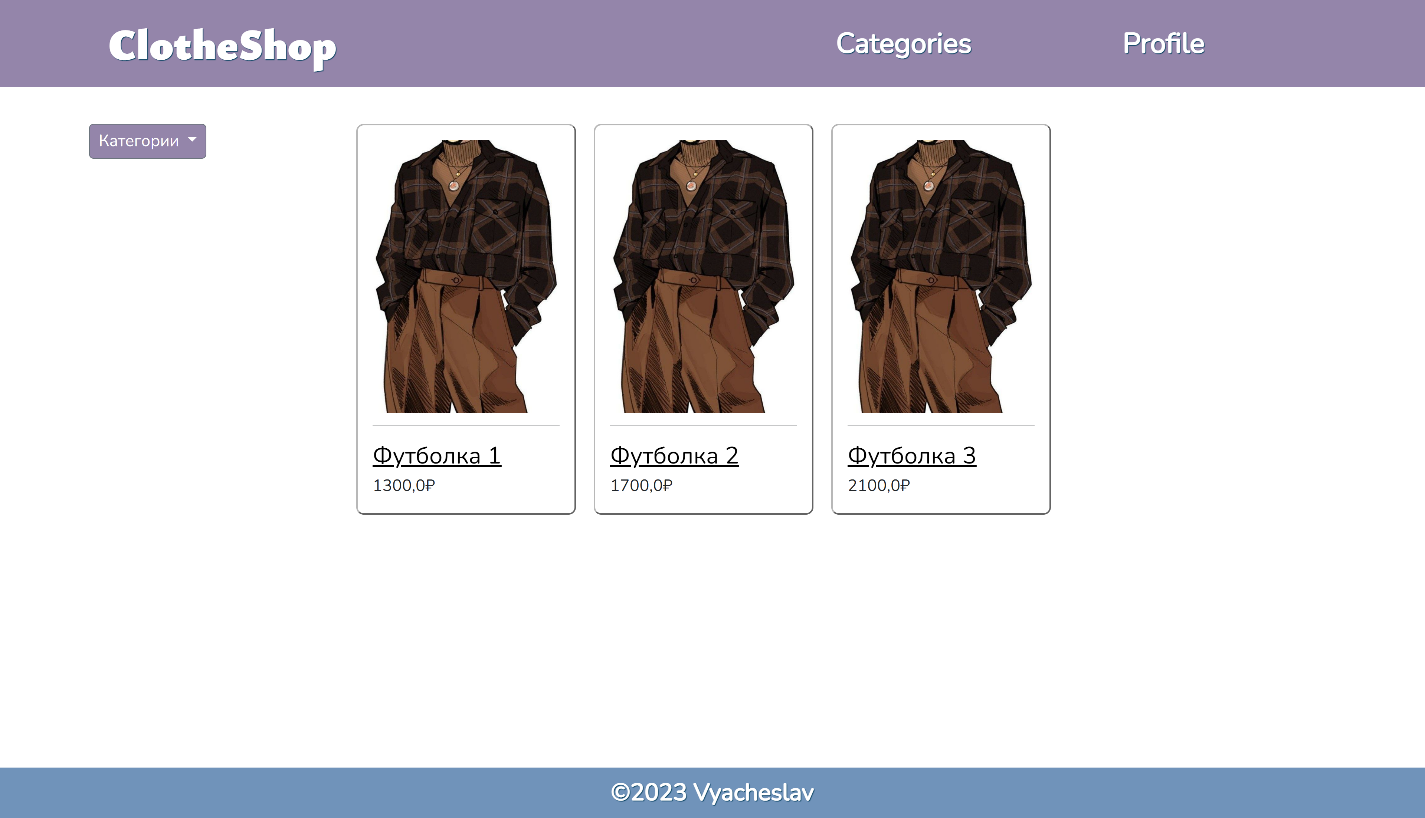


Рисунок 11 – Список товаров

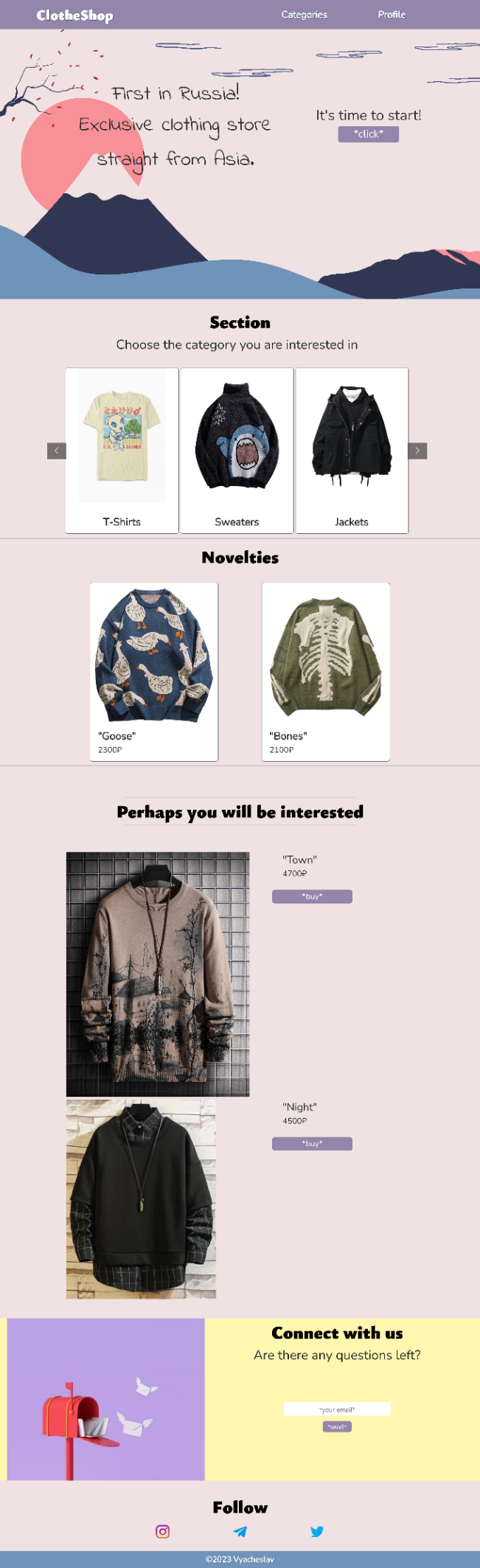


Рисунок 12 – Главная страница