Автономная некоммерческая организация Дополнительного профессионального образования

Компьютерная Академия «ТОП» филиал «Академия ТОП Уфа»

Дипломная работа

по курсу «Веб-разработка на Python»

«Разработка WEB-приложения интернет – магазина тканей»

Выполнила студент Компьютерной Академии ТОП
Ахметова Алия Шамилевна/_____

Дипломная работа допущена к защите и проверена на объем заимствования:

Директор филиала
АНО ДПО «Академия ТОП»
Игнатьева Азалия Фаритовна\ _____

Рук. учебной части филиала
АНО ДПО «Академия ТОП»
Фатхинурова Светлана Форагатовна_____

Уфа, 2024г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Век информационных технологий и цифровизации охватывает все сферы нашей жизни. Мы живем в эпоху, когда цифровые технологии проникают в каждую отрасль, от образования и здравоохранения до бизнеса и развлечений. Цифровизация позволяет нам оптимизировать процессы, повышать эффективность и улучшать качество наших дней.

С каждым годом мы становимся свидетелями стремительного развития технологий: искусственный интеллект, большие данные, облачные вычисления и интернет вещей — все это уже стало неотъемлемой частью нашего повседневного существования. Мы можем работать удаленно, общаться с людьми по всему миру в реальном времени и получать доступ к информации в любое время и в любом месте.

Однако с этими возможностями приходят и новые вызовы. Вопросы безопасности данных, конфиденциальности и цифрового неравенства становятся все более актуальными. Важно не только адаптироваться к новым условиям, но и осознанно подходить к использованию технологий, чтобы они служили на благо общества.

Таким образом, цифровизация — это не просто тренд, а необходимость, которая формирует наше будущее. Мы находимся на пороге новой эры, где технологии будут продолжать трансформировать наш мир, открывая новые горизонты и возможности.

В данной работе пойдёт речь о разработке WEB-приложения интернет – магазина тканей. Целью данной работы является создание WEB-приложения тематики Интернет-магазина, который будет иметь следующий функционал:

1. Навигация на странице
2. Собирать данные из HTML-страницы

ГЛАВА 1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

Веб-приложение позволяет рассказать о своем бизнесе или новом продукте большему количеству потенциальных клиентов. Он помогает удобно взаимодействовать с существующими покупателями, партнерами, сотрудниками. Около 90% бизнеса в России представлены онлайн. Это витрина бизнеса в интернете, которую можно оформить как угодно.

На сайте можно продемонстрировать компанию с лучшей стороны и подробно. Отобразить процессы, работников, сертификаты, технологии, локации, а также любой каталог товаров и услуг.

Сайт помогает рассказать о бизнесе максимально понятно. На сайте информация располагается последовательно, и ее всегда можно найти. Обычно при разработке выясняют, что важно потенциальным клиентам и партнерам. А потом это учитывается при проектировании.

От дизайна и удобства сайта во многом зависит пользовательский опыт — понравится ли клиенту на сайте, захочет ли он воспользоваться сервисом снова. Этот опыт можно постоянно анализировать и улучшать.

Привлекать новых клиентов и партнеров. Сайты рекламируют через контекстную рекламу, баннеры, продвижение в поисковиках и даже через традиционные каналы. Реклама в интернете может привлечь не только покупателей, но и возможных инвесторов, партнеров и работников.

Продемонстрировать результаты работы. Сайт позволяет разместить портфолио: описание решенных задач с фото, видео, текстом и любой графикой. Например, потенциальные клиенты и партнеры могут убедиться в опыте исполнителя.

Автоматизировать процессы и снизить затраты. Сайт дает возможность автоматизировать оплату и доставку, настроить самостоятельные заказы, а также оформление документов. Еще сайты можно интегрировать с другими приложениями, например, с 1С и CRM-системами, чтобы облегчить учет продаж.

Таким образом, веб-приложение создается для привлечения новых клиентов и заказчиков. Не нужно скачивать на устройство и поэтому не занимает много пространства. Не требует обновления — в браузере пользователь всегда работает с самой актуальной версией. Можно использовать на любых устройствах, где есть браузер и интернет. Это удобно и эффективно.

ГЛАВА 2. ИСПОЛЬЗОВАНИЕ ИНСТРУМЕНТОВ ДЛЯ РАЗРАБОТКИ ВЕБ-ПРИЛОЖЕНИЯ

Для создания веб-приложения использовались Django, HTML, CSS, JavaScript, база данных SQLite. Вместе эти технологии образуют мощный стек для разработки современных веб-приложений, где Django управляет серверной логикой и базами данных, HTML структурирует контент, CSS отвечает за стиль, а JavaScript добавляет интерактивность.

2.1 Django

Django - Это высокоуровневый веб-фреймворк на Python, который упрощает создание веб-приложений. Он следует принципу "не повторяйся" (DRY) и предоставляет множество встроенных функций, таких как аутентификация, админ-панель, ORM (Object-Relational Mapping) для работы с базами данных и маршрутизация URL. Django позволяет быстро разрабатывать безопасные и масштабируемые веб-приложения.

Архитектура Django похожа на «Модель-Представление-Контроллер» (MVC). Контроллер классической модели MVC примерно соответствует уровню, который в Django называется Представление (англ. View), а презентационная логика Представления реализуется в Django уровнем Шаблонов (англ. Template). Из-за этого уровневую архитектуру Django часто называют «Модель-Шаблон-Представление» (MTV).

Первоначальная разработка Django как средства для работы новостных ресурсов достаточно сильно отразилась на его архитектуре: он предоставляет ряд средств, которые помогают в быстрой разработке веб-сайтов информационного характера. Так, например, разработчику не требуется создавать контроллеры и страницы для административной части сайта, в Django есть встроенное приложение для управления содержимым, которое можно включить в любой сайт, сделанный на Django, и которое может

управлять сразу несколькими сайтами на одном сервере. Административное приложение позволяет создавать, изменять и удалять любые объекты наполнения сайта, протоколируя все совершённые действия, и предоставляет интерфейс для управления пользователями и группами (с пообъектным назначением прав).

В дистрибутив Django также включены приложения для системы комментариев, синдикации RSS и Atom, «статических страниц» (которыми можно управлять без необходимости писать контроллеры и представления), перенаправления URL и другое.

Джанго также предлагает множество инструментов и утилит, которые облегчают разработку веб-приложений, такие как встроенная система аутентификации и авторизации, обработка форм, валидация данных, механизмы кэширования, миграции баз данных и многое другое.

Система URL-маршрутизации Django позволяет легко определять структуру URL-адресов приложения и их соответствующие обработчики представлений (views), что способствует лучшей организации кода и повышает его читаемость.

Благодаря использованию ORM (Object-Relational Mapping), Django обеспечивает абстракцию от деталей взаимодействия с базой данных, что упрощает работу с данными и делает код более переносимым и поддерживаемым.

2.2 HTML

HTML (HyperText Markup Language) - это стандартный язык разметки для создания веб-страниц. HTML используется для структурирования контента на веб-странице, включая заголовки, абзацы, списки, изображения и ссылки. Он определяет, как элементы отображаются в браузере.

Язык гипертекстовой разметки HTML был разработан британским учёным Тимом Бернерсом-Ли приблизительно в 1986—1991 годах в

стенах ЦЕРНа в Женеве в Швейцарии. HTML создавался как язык для обмена научной и технической документацией, пригодный для использования людьми, не являющимися специалистами в области вёрстки. HTML успешно справлялся с проблемой сложности SGML путём определения небольшого набора структурных и семантических элементов — дескрипторов. Дескрипторы также часто называют «тегами». С помощью HTML можно легко создать относительно простой, но красиво оформленный документ. Помимо упрощения структуры документа, в HTML внесена поддержка гипертекста. Мультимедийные возможности были добавлены позже.

Первым общедоступным описанием HTML был документ «Теги HTML», впервые упомянутый в Интернете Тимом Бернерсом-Ли в конце 1991 года. В нём описываются 18 элементов, составляющих первоначальный, относительно простой дизайн HTML. За исключением тега гиперссылки, на них сильно повлиял SGMLguid, внутренний формат документации, основанный на стандартном обобщенном языке разметки (SGML), в CERN. Одиннадцать из этих элементов всё ещё существуют в HTML 4.

Изначально язык HTML был задуман и создан как средство структурирования и форматирования документов без их привязки к средствам воспроизведения (отображения). В идеале, текст с разметкой HTML должен был без стилистических и структурных искажений воспроизводиться на оборудовании с различной технической оснащённостью (цветной экран современного компьютера, монохромный экран органайзера, ограниченный по размерам экран мобильного телефона или устройства и программы голосового воспроизведения текстов). Однако современное применение HTML очень далеко от его изначальной задачи. Например, тег <table> предназначен для создания в документах таблиц, но иногда используется и для оформления размещения элементов на странице. С течением времени основная идея платформонезависимости языка HTML была принесена в жертву современным потребностям в мультимедийном и графическом оформлении. Сейчас используется версия HTML 5.

2.3 CSS

CSS (Cascading Style Sheets) – это язык стилей, который используется для описания внешнего вида HTML-документов. CSS позволяет задавать цвета, шрифты, отступы, размеры и другие визуальные аспекты элементов на странице. Он помогает сделать веб-страницы более привлекательными и удобными для пользователей.

CSS используется создателями веб-страниц для задания цветов, шрифтов, стилей, расположения отдельных блоков и других аспектов представления внешнего вида этих веб-страниц. Основной целью разработки CSS является ограждение и отделение описания логической структуры веб-страницы (которое производится с помощью HTML или других языков разметки) от описания внешнего вида этой веб-страницы (которое теперь производится с помощью формального языка CSS). Такое разделение может увеличить доступность документа, предоставить большую гибкость и возможность управления его представлением, а также уменьшить сложность и повторяемость в структурном содержимом.

2.4 JavaScript

JavaScript - это язык программирования, который позволяет добавлять интерактивность и динамическое поведение на веб-страницы. JavaScript может изменять HTML и CSS в реальном времени, обрабатывать события (например, клики мыши), взаимодействовать с сервером и выполнять множество других задач, что делает веб-приложения более интерактивными и отзывчивыми. Он мультипарадигменный язык программирования. Поддерживает объектно-ориентированный, императивный и функциональный стили. Является реализацией спецификации ECMAScript.

JavaScript обычно используется как встраиваемый язык для программного доступа к объектам приложений. Наиболее широкое применение находит в браузерах как язык сценариев для придания интерактивности веб-страницам

Основные архитектурные черты: динамическая типизация, слабая типизация, автоматическое управление памятью, прототипное программирование, функции как объекты первого класса.

2.5 SQLite 3

Слово «встраиваемый» (англ. embedded) означает, что SQLite не использует парадигмы клиент-сервер, то есть движок SQLite не является отдельно работающим процессом, с которым взаимодействует программа, а представляет собой библиотеку, с которой программа компонуется, и движок становится составной частью программы. Таким образом, в качестве протокола обмена используются вызовы функций (API) библиотеки SQLite. Такой подход уменьшает накладные расходы, время отклика и упрощает программу. SQLite хранит всю базу данных (включая определения, таблицы, индексы и данные) в единственном стандартном файле на том компьютере, на котором исполняется программа. Простота реализации достигается за счёт того, что перед началом исполнения транзакции записи весь файл, хранящий базу данных, блокируется; ACID-функции достигаются в том числе за счёт создания файла журнала.

Несколько процессов или потоков могут одновременно без каких-либо проблем читать данные из одной базы. Запись в базу можно осуществить только в том случае, если никаких других запросов в данный момент не обслуживается; в противном случае попытка записи оканчивается неудачей, и в программу возвращается код ошибки. Другим вариантом развития событий является автоматическое повторение попыток записи в течение заданного интервала времени.

В комплекте поставки идёт также функциональная клиентская часть в виде исполняемого файла sqlite3, с помощью которого демонстрируется реализация функций основной библиотеки. Клиентская часть является кроссплатформенной утилитой командной строки.

SQLite возможно использовать как на встраиваемых системах, так и на выделенных машинах с гигабайтными массивами данных.

ГЛАВА 3. РЕАЛИЗАЦИЯ

Реализацию в работе проведена в среде PyCharm. Создали Вертуальное окружение, приложение blogapp, проект blogject и учётную запись администратора, установили базу данных SQLite3 и установили библиотеку Pillow для добавления изображений в базу данных. А так же в settings.py проекта добавили настройки

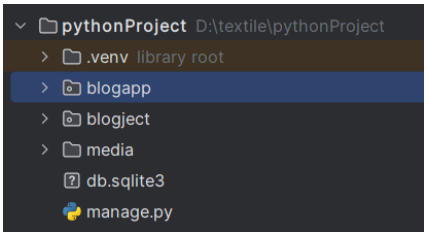

Рис. 1 – Архитектура проекта

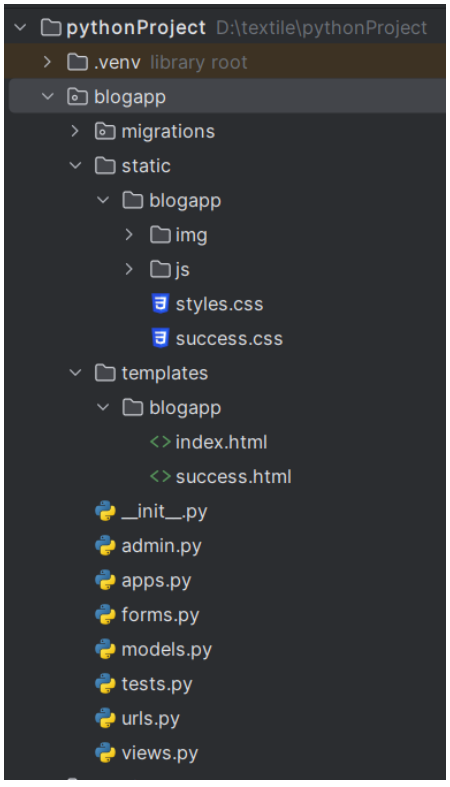

Рис. 2 – Архитектура приложения

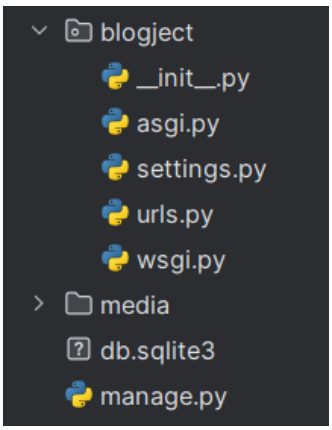

Рис. 3 – Архитектура проекта

Приложение Blogapp (Рис. 2) использовались файлы:

Index.html – Главная страница

```
{% load static %}
<!DOCTYPE html>
<html lang="ru" xmlns="http://www.w3.org/1999/html">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Магазин Тканей "Идеал"</title>
   <link rel="stylesheet" href="{% static 'blogapp/styles.css' %}">
    <script src="{% static 'blogapp/js/script.js' %}"></script>
</head>
<body>
   <header>
      <nav>
         <div class="logo">Магазин Тканей</div>
         <ul>
            <li><a href="#home">Главная</a></li>
            <li><a href="#about">О нас</a></li>
            <li><a href="#products">Ткани</a></li>
            <li><a href="#contact">Контакты</a></li>
         </ul>
      </nav>
   </header>

   <section id="home" class="hero-section">
      <h1>Добро пожаловать в Магазин Тканей</h1>
      <p>Высококачественные ткани для ваших лучших проектов</p>
```

```
    <a href="#products" class="btn">Каталог Тканей</a>
  </section>

  <section id="about">
    <h2>О нас</h2>
    <p>Мы предлагаем широкий ассортимент тканей для любых нужд. Качество и разнообразие нашей продукции удовлетворит самые изысканные предпочтения.</p>
  </section>

  <section id="products">
    <h2>Наши Ткани</h2>
      <div class="product-grid">
       <div class="product-card">
          <img src="{% static 'blogapp/img/s_1.jpg' %}" alt="Ткань 1">
          <h3>Шёлк премиум</h3>
          <p>Мягкий и гладкий шёлк для элегантных изделий.</p>
        </div>
        <div class="product-card">
          <img src="{% static 'blogapp/img/cotton.jpg' %}" alt="Ткань 2">
          <h3>Хлопок натуральный</h3>
          <p>Дышащая ткань идеальна для повседневной одежды.</p>
        </div>
        <div class="product-card">
          <img src="{% static 'blogapp/img/lon.jpg' %}" alt="Ткань 3">
          <h3>Лён экологичный</h3>
          <p>Стиль и комфорт в каждом метре ткани.</p>
        </div>
    </div>
  </section>

  <section id="contact">
    <h2>Свяжитесь с нами</h2>
    <form id="contact-form"  method="post">
      {% csrf_token %}
      {{ form.name }}
      {{ form.email }}
      {{ form.message }}
      <button type="submit">Отправить</button>
    </form>
  </section>

  <footer>
    <p>&copy; 2024 Магазин Тканей. Все права защищены.</p>
  </footer>
</body>
</html>
```

При заполнении в базу данных пользователями, что данные загружены success.html:

```
{% load static %}
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title> Сохранение </title>
   <link rel="stylesheet" href="{% static 'blogapp/success.css' %}">
</head>
<body>
<h1>Данные сохранены!</h1>
   <a href="{% url 'user_information' %}">
      <button class="back_button" >Обратно</button>
   </a>

</body>
</html>
```

Создано две модели model.py. class UserInformation – для сбора информации от пользователей и class Textile для добавления, редактирование, изменения и удаления карточек товара с панели администратора.

```
from django.db import models


class UserInformation(models.Model):
   name = models.CharField(max_length=100)
   email = models.EmailField()
   message = models.TextField()

   def __str__(self):
      return self.name


class Textile(models.Model):
   image = models.ImageField(upload_to='img')  # Картинка ткани
   name_Textile = models.CharField(max_length=200)  # Название ткани
   description = models.TextField()  # Описание ткани
   price = models.DecimalField(max_digits=10, decimal_places=2)  # Цена ткани за 1 м

   def __str__(self):
      return self.name_Textile
```

Далее модель регистрируется в панели администратора. И в панели администратора появляются базы данных Рис. 4.

```
from django.contrib import admin
from .models import UserInformation
from .models import Textile


admin.site.register(UserInformation)
admin.site.register(Textile)
```

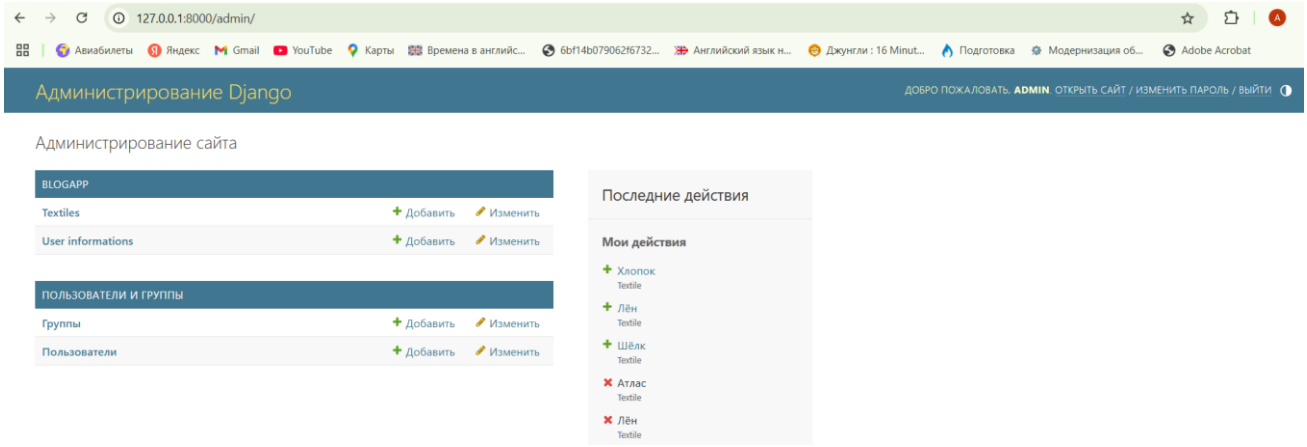

Рис. 4 – Панель администратора

Чтобы была форма для сбора информации от пользователей на сайте, нужно создать файл forms.py

```
from django.forms import ModelForm
from .models import UserInformation


class UserInformationForm(ModelForm):
  class Meta:
    model = UserInformation
    fields = ['name', 'email', 'message']
```

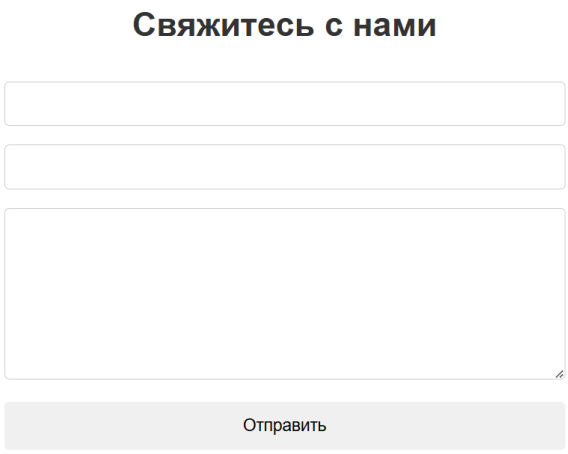

Рис. 4 – Сбор информации от пользователей

В Django файл urls.py в приложении отвечает за управление URL-адресами (маршрутами).

```
from .views import index
from django.urls import path
from .views import success_view
from .views import textile_gallery



urlpatterns = [
    path('', index,  name='user_information'),
    path('/success', success_view, name='success'),
    path('', textile_gallery, name='textile_gallery'),

]
```

В Django файл views.py в приложении отвечает за определение логики обработки запросов и формирования ответов для пользователей. Он содержит функции или классы представлений (views), которые обрабатывают HTTP-запросы и возвращают HTTP-ответы. Вот основные моменты о views.py:

В файле views.py создаются функции, которые будут обрабатывать запросы и перенаправлять на другую страницу после нажатия на «Отправить» из Рис. 4. Дале после заполнения имени, электронной почты и сообщения веб –приложение нас сбрасывает на страницу Рис.5.

```
from django.shortcuts import render, redirect
from .forms import UserInformationForm
from .models import Textile


def index(request):
    if request.method == 'POST':
        form = UserInformationForm(request.POST)
        if form.is_valid():
            form.save()  # Сохранение данных в базу
            return redirect('success')  # Перенаправление после успешного сохранения
    else:
        form = UserInformationForm()
    return render(request, 'blogapp/index.html', {'Name': 'Имя', 'Email': 'Email', 'Massage': 'Сообщение',
'form': form})


def success_view(request):
    return render(request, 'blogapp/success.html')


def textile_gallery(request):
    textiles = Textile.objects.all()  # Получаем все ткани из базы данных
    return render(request, 'blogapp/index.html', {'textiles': textiles})
```

Рис. 5 – Сохранение данных

Если нажать на «Обратно», то мы подойдём на главную страницу

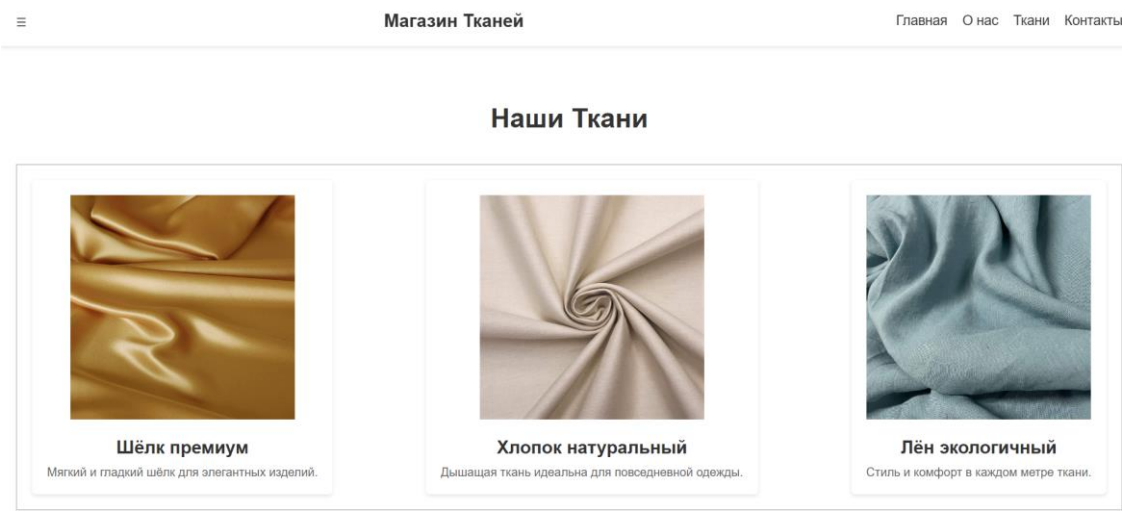

Рис.6 - Главная страница сайта

А также для сайта были использованы стили файлов scc и js

Stytes.css

```
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}

body {
  font-family: 'Arial', sans-serif;
  color: #333;
}

a {
  text-decoration: none;
  color: inherit;
}


header {
  background-color: #fff;
  position: fixed;
  width: 100%;
  top: 0;
  z-index: 100;
  box-shadow: 0 2px 5px rgba(0,0,0,0.1);
}
```

```
nav {
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: 20px;
}

.logo {
    font-size: 24px;
    font-weight: bold;
}

nav ul {
    list-style: none;
    display: flex;
}

nav ul li {
    margin-left: 20px;
}

nav ul li a {
    font-size: 18px;
    transition: color 0.3s;
}

nav ul li a:hover {
    color: #007BFF;
}

.hero-section {

    background-size: cover;
    background-position: center;
    height: 100vh;
    padding-top: 100px;
    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: center;
    text-align: center;
    color: #fff;
}

.hero-section h1 {
    font-size: 48px;
    margin-bottom: 20px;
}
```

```
.hero-section p {
    font-size: 24px;
    margin-bottom: 30px;
}

.btn {
    background-color: #007BFF;
    color: #fff;
    padding: 15px 30px;
    font-size: 18px;
    border: none;
    border-radius: 5px;
    cursor: pointer;
    transition: background-color 0.3s;
}

.btn:hover {
    background-color: #0056b3;
}

section {
    padding: 80px 20px;
    text-align: center;
}

section h2 {
    font-size: 36px;
    margin-bottom: 40px;
}

.product-grid {
    display: flex;
    flex-wrap: wrap;
    justify-content: space-between;
    border: 2px solid #ccc;
    padding: 20px;
}

.product-card {
    background-color: #fff;
    padding: 20px;
    box-shadow: 0 2px 5px rgba(0,0,0,0.1);
    border-radius: 5px;
}

.product-card img {
    width: 300px;
    height: 300px;
    margin-bottom: 20px;
```

```
}

.product-card h3 {
    font-size: 24px;
    margin-bottom: 10px;
}

.product-card p {
    font-size: 16px;
    color: #666;
}

#contact-form {
    max-width: 600px;
    margin: 0 auto;
}

#contact-form input,
#contact-form textarea {
    width: 100%;
    padding: 15px;
    margin-bottom: 20px;
    border: 1px solid #ccc;
    border-radius: 5px;
}

#contact-form button {
    width: 100%;
    padding: 15px;
"""background-color: #007BFF;"""
    color: #fff;
    font-size: 18px;
    border: none;
    border-radius: 5px;
    cursor: pointer;
}

#contact-form button:hover {
    background-color: #0056b3;
}

footer {
    background-color: #f8f8f8;
    padding: 20px;
    text-align: center;
}
```

Script.js который обеспечивает навигацию по сайту:

```
document.addEventListener('DOMContentLoaded', function() {
    const nav = document.querySelector('nav');
    const menuToggle = document.createElement('div');
    menuToggle.className = 'menu-toggle';
    menuToggle.innerHTML = '&#9776;';
    nav.insertBefore(menuToggle, nav.firstChild);

    menuToggle.addEventListener('click', function() {
        nav.classList.toggle('active');
    });
});

// Скрипт для плавной прокрутки к секциям
const links = document.querySelectorAll('nav ul li a');

for (const link of links) {
    link.addEventListener('click', clickHandler);
}

function clickHandler(e) {
    e.preventDefault();
    const href = this.getAttribute('href');
    const offsetTop = document.querySelector(href).offsetTop - 70;

    scroll({
        top: offsetTop,
        behavior: 'smooth'
    });
}
```

Перейдем в проект blogject, в работе использовались setting.py и urls.py

В setting.py

Импортировали библиотеки:

```
from pathlib import Path
import os
```

Настроили:

1. ALLOWED_HOSTS = ['127.0.0.1'] – локальный адрес, где выводится WEB-приложение.

2. INSTALLED_APPS = […,

'blogapp',

] - было добавлено приложение.

3. DATABASES = {

'default': {

'ENGINE': 'django.db.backends.sqlite3',

'NAME': BASE_DIR / 'db.sqlite3',

}

}– установлена база данных SQLite3

4. LANGUAGE_CODE = 'ru-ru' – Настроили русский язык в панели администратора

5. TIME_ZONE = 'Asia/Yekaterinburg' – установили местное время

6. STATIC_URL = 'static/'

STATICFILES_DIRS = [

os.path.join(BASE_DIR, "static"),] – установили статические файлы для изображений, CSS, JS.

7. MEDIA_URL = '/media/'

MEDIA_ROOT = os.path.join(BASE_DIR, 'media') – настройка места, где хранятся изображения добавленные в базу данных.

В urls.py маршруты панели администратора и маршрут к приложению blogapp.urls

```
from django.contrib import admin
from django.urls import path, include
from django.conf import settings
from django.conf.urls.static import static
```

```
urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('blogapp.urls')),
] + static(settings.STATIC_URL, document_root=settings.STATIC_ROOT)



if settings.DEBUG:
    urlpatterns += static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

ЗАКЛЮЧЕНИЕ

Таким образом, по данной работе является создание WEB-приложения тематики Интернет-магазина, который будет иметь следующий функционал:

1. Навигация на странице

2. Собирать данные из HTML-страницы.

А также закрепление изученного материала по курсу «Веб-разработка на Python» мощного стека для разработки современных веб-приложений, где Django управляет серверной логикой и базами данных.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

https://ru.wikipedia.org/wiki/HTML HTML - стандартизированный язык гипертекстовой разметки документов для просмотра веб-страниц в браузере.

https://ru.wikipedia.org/wiki/CSS CSS - формальный язык декодирования и описания внешнего вида веб-страницы.

https://ru.wikipedia.org/wiki/JavaScript JavaScript - язык сценариев для придания интерактивности веб-страницам.

https://proproprogs.ru/ Django – это фреймворк, значительно упрощающий написание скриптов на языке Python.

ПРИЛОЖЕНИЕ 1

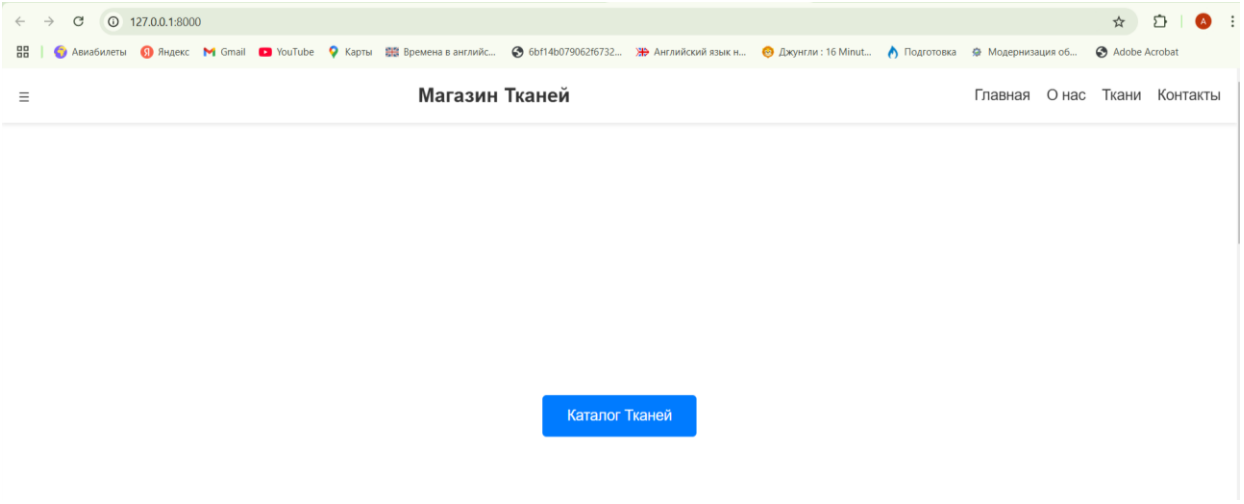

Рис.1 - Главная страница сайта (Главная)

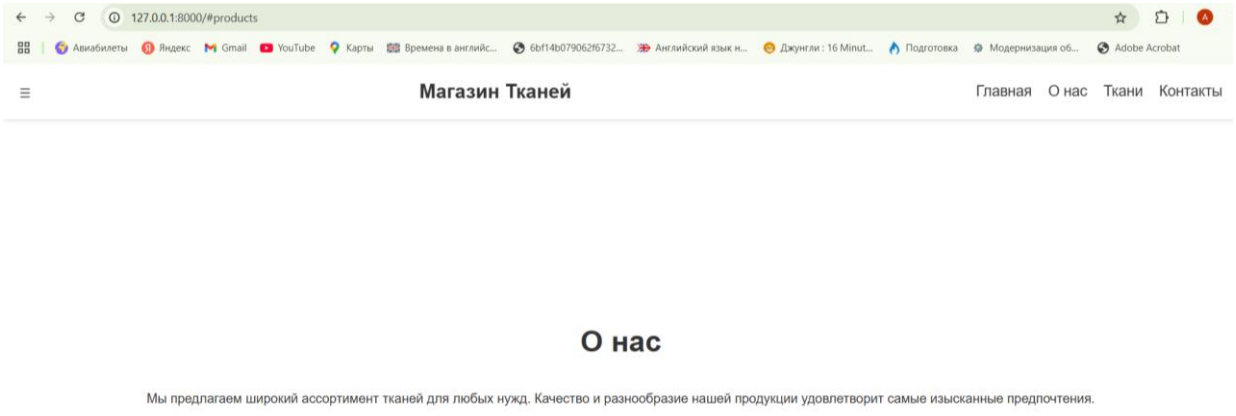

Рис.2 - Главная страница сайта (О нас)

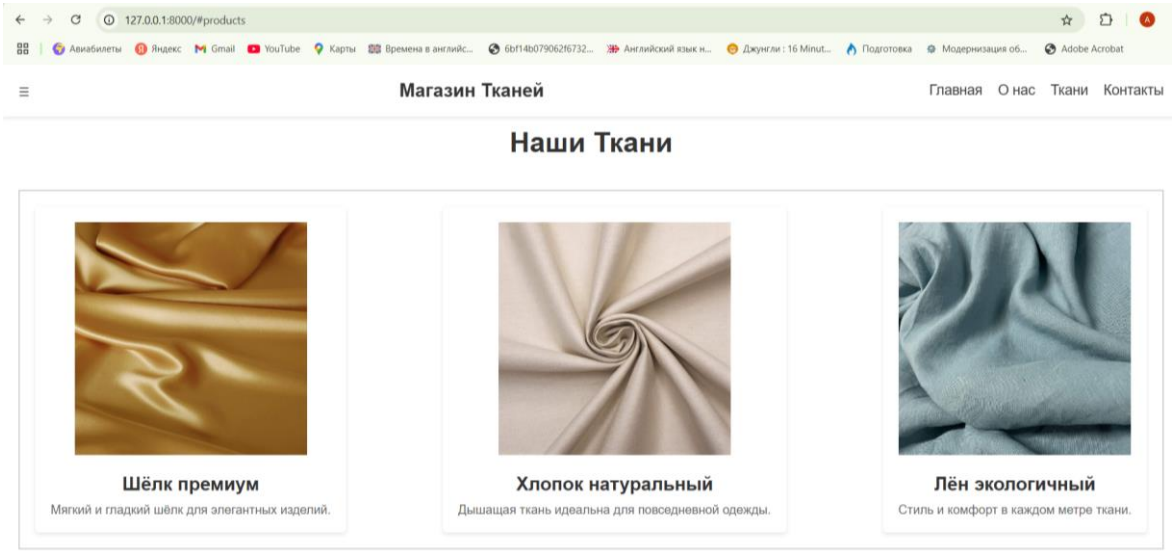

Рис.3 - Главная страница сайта (Карточки товаров)

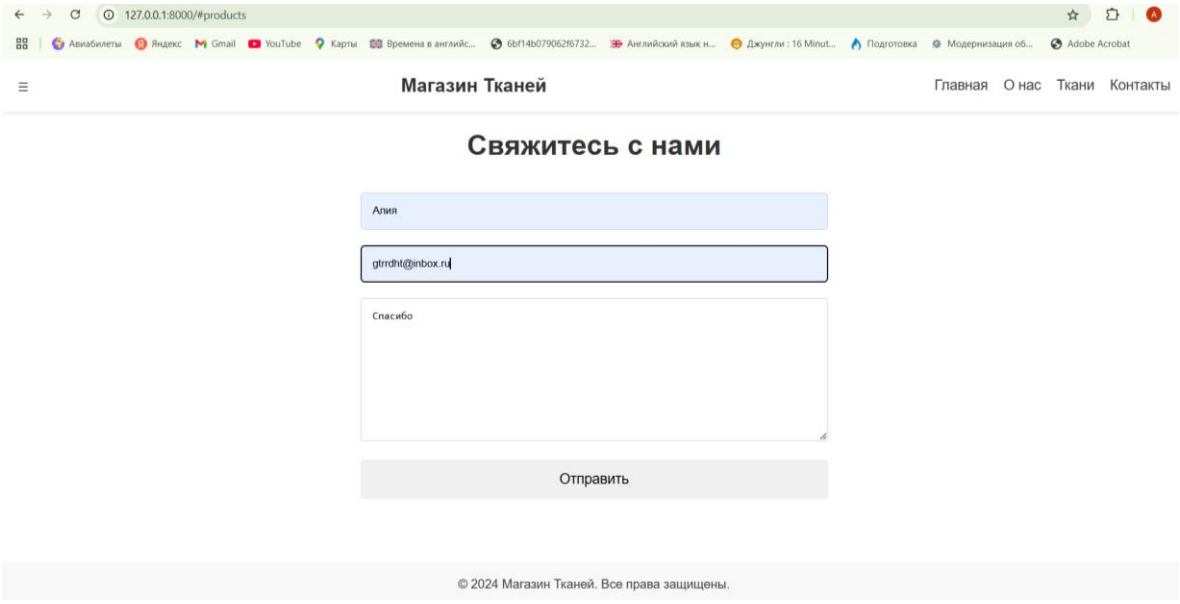

Рис.4 - Главная страница сайта (Форма связи с пользователями)

Рис.5 – Сохранение данных пользователей в базе данных