

Специальность: Frontend-программист “Цифровые профессии”

**«Создание сайта-визитки с элементами интернет-магазина для  
презентации услуг арт-сервиса Штрихкод, а также ознакомления с  
творчеством современных фотографов и художников с возможностью  
приобретения их работ».**

Специальность:  
Frontend-программист  
«Цифровые профессии»  
Поскробко Екатерина Юрьевна

Москва, 2024 год.

## **Оглавление**

Введение	3
1. Создание современных веб-сайтов.	4
1.1. Определение понятия «веб-сайт».	4
1.2. Классификация сайтов.	4
1.3. Устройство сайта.	7
1.4. Способы создания веб-сайтов.	9
1.5. Обзор программных средств фронтенд разработки.	13
1.6. Фреймворки, библиотеки и препроцессоры.	16
1.7. Этапы выполнения проектов по созданию веб-сайтов.	20
2. Практическая реализации задачи по созданию веб-сайта.	21
2.1. Программные продукты и инструменты веб-разработки.	21
2.2. Верстка страниц сайта.	23
2.3. Стилизация страниц сайта.	28
2.4. Методология БЭМ для верстки и стилизации сайта.	28
2.5. Загрузка и подключение шрифтов.	29
2.6. Использование изображений.	30
2.7. Нормализация настроек.	31
2.8. Создание интерактивных элементов главной страницы сайта.	31
2.9. Разработка внутренних страниц сайта.	36
2.10. Проверка веб-сайта на ошибки.	45
2.11. Загрузка веб-сайта в сеть интернет.	49
Заключение	50
Список использованной литературы	51

## **Введение**

### Цель:

Создать фронтенд часть для адаптивного сайта-визитки для арт-сервиса «Штрихкод», деятельность которого заключается в формировании частных и корпоративных коллекций. Специалисты-искусствоведы «Штрихкода» тесно сотрудничают с авторами коллекционных фотографий, графики и офортами. Все работы сопровождаются сертификатами с атрибуцией и подписью автора. Для своих клиентов в арт-сервисе печатают и оформляют работы на музейном уровне. Некоторые фотографии и картины можно приобрести в стандартном размере непосредственно через интернет-магазин.

### Основные задачи:

1. Разобраться в базовых подходах к созданию веб-сайтов, выбрать оптимальную методологию и необходимые инструменты для создания веб-сайта в соответствии с минимальным техническим заданием.
2. Продумать дизайн.
3. Создать структуру веб-страницы и сделать ее стилизацию, то есть осуществить верстку сайта.
4. При помощи CSS3 и JavaScript придать сайту интерактивность.
5. Протестировать веб-сайт с целью выявления ошибок.
6. Разместить веб-сайт в сети интернет.

Используемые инструменты: HTML, CSS, SASS, JavaScript, GitHub Visial Studio Code и прочие.

### Суть дипломного проекта:

Разработать сайт, презентующий услуги арт-сервиса, представляющий талантливых современных художников и фотографов и их работы, а также позволяющий приобрести некоторые произведения искусства.

# **1. Создание современных веб-сайтов.**

## **1.1. Определение понятия «веб-сайт».**

**Веб-сайт** — это набор логически связанных между собой веб-страниц, расположенных в Сети под единым именем — доменом. Веб-страницы — это документы, которые написаны на языке разметки HTML и доступны в вебе. Веб-страницы содержат контент — например, статьи, фото или видео.

## **1.2. Классификация сайтов.**

Все веб-сайты можно классифицировать по разным критериям. Так, наиболее интересными являются следующие признаки:

- назначение,
- применяемая технология,
- размер.

*по назначению:*

**Сайт-визитка** — небольшой сайт из одной или нескольких страниц, на котором размещают основную информацию о человеке, компании, товаре или услуге. На сайте-визитке можно разместить описание товара или компании, фотографии и контактные данные. Сайт именно такого формата и создавался мной в рамках практической части данной работы.

**Корпоративный сайт** — сайт с подробной информацией о компании, её товарах или услугах. Также на сайте могут быть разделы с новостями, документами, вакансиями. Корпоративные ресурсы создают, чтобы продавать товары или услуги. Поэтому на корпоративных сайтах, в отличие от сайтов-визиток, можно оформить заказ или хотя бы связаться с менеджером — например, оставить заявку на обратный звонок.

**Интернет-магазин** — сайт, на котором продаёт товары один продавец. В интернет-магазине есть каталог товаров, корзина и система оплаты. Интернет-магазины заводят производители, дистрибуторы, торговые компании.

Разновидностью интернет-магазина является другой тип сайтов — маркетплейсы. Разница заключается в том, что сайтом владеет не отдельный продавец, а компания-посредник.

**Новостной сайт** — ресурс, на котором размещают новости. Новостной сайт может быть узкотематическим — например, только с новостями спорта. Новостные сайты создают, например, СМИ, которые раньше существовали в офлайне.

**Блог**, или информационный сайт, — сайт, на котором автор или группа авторов регулярно размещает информационные статьи. Статьи в блогах могут быть посвящены жизни автора, хобби, узкой области знаний, профессиональным или практическим вопросам. Блоги обычно посвящены одной теме.

*по размеру:*

Одностраничный сайт позволяет эффективно представить пользователю информацию об услугах специалиста или компании, рассказать об определенном продукте. Словом, такой сайт не предусматривает никаких действий от пользователя, а следовательно, не обладает сколько-нибудь сложным функционалом.

Примером одностраничного сайта является landing page или сайт-визитка, размещающим всю информацию на одной странице.

**Многостраничный сайт** может иметь различную структуру, разное количество страниц, тематическую организацию и схему внутренних связей. Общим для многостраничных сайтов является обязательное наличие главной страницы, имеющей связи со всеми тематическими разделами, которые, в свою очередь,

могут иметь собственные главные страницы и подразделы со страницами следующего уровня. Количество страниц многостраничного сайта определяется общим объемом и характером материалов веб-сайта.

*по технологии:*

**Статический сайт** — это сайт, который состоит из фиксированных веб-страниц с постоянным содержимым. Каждая страница представляет собой отдельный хранящийся на сервере HTML-файл и отображается в браузере пользователя без изменений.

### **Особенности статических сайтов:**

- простота создания и поддержки,
- быстрая загрузка страниц,
- отсутствие сложных серверных настроек,
- независимость от баз данных и серверных языков программирования.

### **Пример статического сайта:**

```
<!DOCTYPE html>
<html>
<head>
<title>Статический сайт</title>
</head>
<body>
<h1>Добро пожаловать на мой статический сайт</h1>
<p>Здесь вы найдете информацию о моих увлечениях и интересах.</p>
</body>
</html>
```

**Динамический сайт** — это сайт, который генерирует контент в реальном времени в зависимости от взаимодействия пользователя с сайтом или других переменных. Такие сайты используют серверные языки программирования,

такие как PHP, Python или Ruby, и базы данных для хранения и обработки информации.

### **Особенности динамических сайтов:**

- гибкость и масштабируемость,
- возможность создания сложных веб-приложений,
- использование баз данных и серверных языков программирования,
- больше возможностей для взаимодействия с пользователем.

Статические сайты подходят для простых веб-страниц с постоянным контентом, в то время как динамические сайты предназначены для создания сложных веб-приложений и предоставления пользователю больше возможностей для взаимодействия. Выбор между статическим и динамическим сайтом зависит от ваших целей и требований к проекту.

### **1.3. Устройство сайта.**

По существу, сайт состоит из файлов кода, хостинга, домена, функциональности, дизайна и контента.

**Файлы кода** имеют две основные составляющие:

- Frontend (фронтенд) — клиентская часть, которая обрабатывается на стороне клиента. Браузеры самостоятельно обрабатывают:
  - разметку сайта на языке HTML,
  - стили сайта на языке CSS,
  - различные виды анимации и обработку данных на языке JavaScript.
- Backend (бекенд) — серверная, отвечающая за «логику работы» и скрытая от взгляда пользователя часть сайта. Бекенд отвечает за данные в целом: их получение, хранение, а также предусматривает возможность регистрации или авторизации.

Backend часто представлен двумя компонентами:

- логические обработчики,
- база данных.

**Хостинг** — это размещение файлов сайта на сервере. Любой сайт — набор файлов, причём не только файлов кода, но и файлов изображений, документов и текста. Чтобы люди могли увидеть их и взаимодействовать с ними, сайт должен быть загружен на сервер, компьютер со специальным программным обеспечением и круглосуточным доступом в интернет.

**Доменное имя, или домен**, — это уникальное название сайта, адрес, по которому его можно найти.

Доменное имя состоит из нескольких частей. Разберем на примере secrets.tinkoff.ru

- tinkoff — это домен второго уровня (SLD), левая часть до точки. То есть, это комбинация букв, цифр и символов, используемых для обозначения сайта. Обычно он совпадает илиозвучен с названием бренда и логотипом, легко читается и запоминается.
- .ru — домен первого (верхнего) уровня или доменная зона (TLD), правая часть после точки. Ее нельзя придумать, а можно выбрать из существующих. Зоны разделяются географически и тематически. Например, .RU — для России, .EU — для стран ЕС, .COM — для коммерческой сферы и т. д.
- Домен нулевого уровня — точка после доменной зоны (tinkoff.ru.), которая не отображается в адресной строке.

**Поддомены** — это необязательные подмножества доменного имени. Обычно их создают на крупных порталах для удобства пользователей. Например, если интернет-магазин работает в нескольких регионах, часто создают поддомены под каждый регион отдельно. Это помогает не только клиентам, но и продвижению сайта в поисковых системах. Один из поддоменов в нашем

примере — [secrets.tinkoff.ru](http://secrets.tinkoff.ru), который создан как информационно-новостной портал.

**Функциональность** — набор возможностей, которые предоставляет сайт. У каждого ресурса он свой в зависимости от того, зачем сайт сделали. Например, на большинстве сайтов можно переходить по страницам. В интернет-магазине пользователь может зарегистрироваться, купить и оплатить товар. На сайте компании — заполнить форму обратной связи, чтобы ему перезвонили. Рассчитать стоимость проекта на онлайн-калькуляторе — тоже функциональность сайта.

**Дизайн** — визуальное оформление сайта. Цвета, шрифты и изображения, используемые на странице, расстояние между элементами и их порядок — всё это компоненты дизайна. Дизайн разрабатывают и для элементов самого сайта, и для контента — его наполнения. Дизайн нужен, чтобы сайт выглядел привлекательно, а людям было удобно им пользоваться.

**Контент сайта** — это информационное наполнение веб-страниц: текст, изображения, видео, аудио и мультимедиа.

#### **1.4. Способы создания веб-сайтов.**

Можно выделить три подхода к созданию веб-сайтов:

1. с применением конструктора сайта,
2. на CMS-системе,
3. с использованием языка разметки HTML и языков программирования.

#### **Разработка с использованием конструктора**

Конструктор — онлайн сервис, позволяющий создать сайт на основе готовых шаблонов, которые предоставляет платформа. Конструктор сайтов логичнее использовать для простых сайтов-визиток или лэндинг страниц. То есть имея лишь минимальные знания, а то и вообще без знаний о веб-разработке таким

способом вполне получится создать простой сайт. Совершенно очевидно, что разработать сложное веб-приложение или интернет-магазин на конструкторе невозможно. Среди самых известных конструкторов сайтов стоит отметить: Tilda Publishing, Nethouse и Wix.

#### Преимущества:

- простота использования,
- невысокая стоимость,
- легкость использования готовых стандартных модулей.

#### Недостатки:

- дополнительные расходы,
- размещение на хостинге разработчика данного конструктора,
- электронная почта с именем домена,
- домен третьего или более высокого уровня,
- тяжеловесность сайта,
- ограничения по возможности SEO.

### **Разработка на CMS-системе**

CMS (Content management system) – система управления содержимым – набор базовых и вспомогательных инструментов, позволяющих создать веб-сайт и обеспечить его работу, обновление содержимого и взаимодействие с пользователями сайта. Разработка осуществляется через панель управления.

CMS позволяет реализовать гораздо более значительные проекты, по сравнению с конструкторами сайтов, например, сайт интернет-магазина или многостраничный корпоративный сайт.

#### Преимущества:

- много бесплатных шаблонов,
- легкость управления контентом,
- большое количество готовых фреймворков – моделей, плагинов и дополнений.

#### Недостатки:

- низкий уровень безопасности,

- для реализации нестандартных решений требуется знание основ верстки и программирования,
- затрудненная миграция между хостингами,
- платный дополнительный контент, более затратный по сравнению с конструкторами сайтов.

При выборе CMS стоит обратить внимание на следующие варианты:

1. WordPress. Самая универсальная и распространенная на сегодняшний день система управления, которая подойдет и в случае создания сложного ресурса, и простого для одновременного посещения большого количества пользователей. Система поддерживает широкий диапазон дополнений и тем оформления при том, что система бесплатна и обладает русскоязычным интерфейсом. Простота размещения материала (в два клика) – еще один неоспоримый плюс использования такой системы.
2. Joomla. Этую бесплатную CMS отличает значительная функциональность, огромный перечень оформлений и дополнений, а также широкий спектр для SEO-оптимизации. Единственная сложность – разобраться в системе и подстроить ее под себя. В целом это хороший инструмент для создания сайтов, который отвечает всем главным принципам CMS: понятное и удобное управление, возможность самостоятельного наполнения и редактирования сайта, наличие статистики посещений и многое-многое другое.
3. DLE (DataLife Engine). Платная, однако очень функциональная. Чаще всего такой движок используют в работе с сайтом, требующим использование значительной информационной базы (к примеру, новостного портала). Однако использование многочисленных модулей позволяет превратить сайт в каталог и галерею. Один из главных плюсов такой системы: при большом количестве запросов такая ситуация проходит для сервера почти незаметно. Надежная защита содержимого, простая оптимизация материала для поисковых

систем – это все DLE. Именно поэтому ее выбирают для крупных посещаемых проектов. При этом административная часть будет интуитивно понятна любым пользователям, даже с самыми базовыми знаниями о веб-технологиях.

4. Drupal. Система существует на рынке более 10 лет и поставляется с открытым исходным кодом, а ее главным достоинством, как правило, называют гибкую архитектуру. Также Drupal дает возможность создавать многоязычные веб-сайты.

5. Bitrix от 1С – это платный продукт, на основе которого создаются разнообразные ресурсы с динамическим контентом, система оснащена визуальными редакторами PHP и HTML. Сложность – требует некоторого времени для ознакомления и работы. Среди положительных аспектов этой CMS выделяют интеграцию с «1С:Предприятие 8», технологию композитного сайта, которая позволяет сайтам на Битриксе работать быстрее, отдельный SEO-модуль, а также в целом масштабируемость и отказоустойчивость этой системы.

## **Создание сайта на языках программирования**

Использования языков программирования позволяет создавать проекты любой сложности с учетом индивидуальных требований клиента.

Стоит отметить, что как и для изготовления любого «штучного товара» на создание такого сайта требуется больше времени и усилий.

### **Преимущества:**

- создание нешаблонного решения, максимально оптимизированного для решения конкретных бизнес задач клиента,
- лучшие возможности для продвижения по сравнению с сайтами, созданными с использованием конструкторов и CMS систем,
- уникальный дизайн способный обеспечить максимальную эффективность UI/UX.

### **Недостатки:**

- высокая стоимость для простых сайтов по сравнению с CMS и конструкторами,
- необходимы знания в области программирования, алгоритмов, архитектуры веб-приложений, структур данных и т.п.,
- длинные сроки разработки для простых сайтов.

## 1.5. Обзор программных средств фронтенд разработки.

Базовыми инструментами фронтенд разработки являются HTML, CSS и JavaScript.

- **HTML** (Hypertext Markup Language) - HyperText Markup Language – язык гипертекстовой разметки. Это набор команд, следуя которым, браузеры выводят на монитор различные документы и странички сайтов. Покажем на примере, что такое язык разметки HTML:

*К примеру:*

- *<h1>Это заголовок</h1>*
- *<p>Это текстовый абзац <a href="...">с гиперссылкой</a>.</p>*
- *<ul>*
- *<li>Это элемент ненумерованного перечисления</li>*
- *<li>И ещё один</li>*
- *</ul>*

Главной целью разработки HTML5 стало улучшение читаемости кода человеком и машиной с сохранением обратной совместимости, равно как и поддержки мультимедийных технологий.

Основными новинками HTML5 являются: переопределение и стандартизация ряда элементов, новые элементы мультимедиа `<video>`, `<audio>`, геолокация, улучшение формы и разметки документа, математические формулы и новые семантические теги, файлы `svg`, обработка ошибок, элемент `<canvas>` для непосредственного метода рисования, управление кэшем для онлайн-работы, несколько новых API и прочие.

- **CSS** (Cascading Style Sheets) - это каскадные таблицы стилей. По сути — язык, который отвечает за описание внешнего вида HTML-документа. Подавляющее большинство современных веб-сайтов работают на основе связки HTML+CSS.

- теги не дублируются;
- документ проще обслуживать;
- внешний вид всего сайта можно изменить централизованно, а не корректировать форматирование каждой странички.

CSS позволяет разработчикам создавать стильные и привлекательные веб-страницы, которые могут быть адаптированы под различные устройства и размеры экранов.

Основные принципы CSS:

- Каскадность - стили могут быть определены на разных уровнях, и значения, определенные на более высоком уровне, могут быть переопределены на более низком уровне.
- Наследование - стили, определенные для родительского элемента, могут быть унаследованы его дочерними элементами.
- Приоритетность - стили могут иметь разную приоритетность, и значение, определенное для элемента с более высокой приоритетностью, будет использоваться вместо значения элемента с более низкой приоритетностью.

С выходом стандарта CCS3 появились новые возможности по стилизации HTML документов. Теперь появилась возможность создавать анимированные элементы без использования JavaScript. Среди прочих изменений отметим следующие:

- расширение возможностей селекторов — псевдоэлементы и псевдоклассы,
- внедрение переменных и закрепление элементов на странице position: sticky
- плавность прокрутки и добавление теней к тексту,
- подключение сторонних шрифтов и расчет значений с использованием calc(),
- рамки для форматирования границ элемента,
- улучшение возможностей адаптивной верстки и т.д.

Кроме базовых средств фронтенд разработки, есть широкий спектр дополнительного инструментария - диспетчеры пакетов, CSS-препроцессоры, фреймворки, библиотеки и многое другое.

- **JavaScript** — это мощный язык программирования, который используется для создания интерактивных и динамических веб-сайтов.

Первая версия языка JavaScript была разработана Брэнданом Эйхом за 10 дней в 1995 году в компании Netscape. Первоначально язык назывался Mocha, потом был переименован в LiveScript. На JavaScript оказали влияние такие языки, как C и Java. Поскольку последний в то время был достаточно популярен, то LiveScript было решено переименовать в JavaScript.

Примечательно, что в 1996 году компания Microsoft выпустила аналог JavaScript под названием JScript. Первым браузером который поддерживал JScript был Internet Explorer 3.0.

Наиболее широкое применение JavaScript находит в браузерах (хотя может использоваться и не только в них) для придания веб-страницам интерактивности. Языком JavaScript не владеет какая-либо компания или организация, а название «JavaScript» является зарегистрированным товарным знаком корпорации Oracle в США.

Стандартизация JavaScript началась только в конце 1996 года и получила рабочее название ECMA-262. Ассоциация ECMA, созданная ещё в 1961 году с целью стандартизации информационных и коммуникационных систем, из-за проблем с торговой маркой не могла использовать "JavaScript" в качестве названия. Так появился стандарт ECMAScript, а JavaScript — это его коммерческое название.

Стандарт ежегодно обновляется при этом одной из фундаментальных версий стандарта является ECMAScript 6 (ES-2015), так как она внесла несколько существенных изменений: стрелочные функции, объявление переменных с помощью let и const, промисы, Spread/Rest синтаксис, цикл for...of, шаблонные литералы, новые методы объекта и структуры данных Map/Set, деструктуризация, параметры по умолчанию в функциях и т.д.

Некоторые из ключевых особенностей JavaScript включают в себя:

1. Программирование, управляемое событиями: JavaScript позволяет разработчикам создавать интерактивные веб-страницы, которые реагируют на действия пользователя, такие как нажатие кнопки, прокрутка страницы или ввод данных в форму.
2. Асинхронное программирование: JavaScript поддерживает асинхронное программирование, что позволяет разработчикам писать код, который не блокирует выполнение другого кода.
3. Кроссплатформенная совместимость: JavaScript поддерживается всеми основными веб-браузерами, что делает его популярным выбором для веб-разработки.
4. Расширяемость: JavaScript можно расширить с помощью библиотек и фреймворков, таких как jQuery, React и Vue.js, которые упрощают разработку сложных веб-приложений.

### **1.6. Фреймворки, библиотеки и препроцессоры.**

**CSS-фреймворки** — это набор технологий, включающий в себя усовершенствованный синтаксис CSS, дополнительные функции, а также ряд шаблонов для быстрой развертки сайта без необходимости все элементы стиля прописывать самостоятельно.

По объему функциональности CSS-фреймворки разделяют на полнофункциональные и легкие, имеющие только специализированные инструменты.

CSS-фреймворки также используют разные подходы к обеспечению кроссбраузерности применяя либо сброс стилей - reset.css или нормализацию Normalize.css.

Преимущества:

- ускорение процесса разработки, за счет уничтожения привычных для CSS сложностей с визуальным оформлением страниц,

- добавление в сайт или приложение кроссбраузерность,
- вынуждение команды разработчиков следовать определенным правилам дизайна,
- корректное выравнивание всех элементов на странице,
- создание подключаемых стилей, более удобными с точки зрения поддержки.

#### Недостатки:

- увеличение «веса» проекта не используемым кодом,
- ограничение вариантов дизайна стандартами применяемого фреймворка,
- создание «многоклассие» стилей на одном элементе.

Наиболее популярными css-фреймворками являются Bootstrap, Materialize CSS, Foundation.

**CSS препроцессоры** — это программы, позволяющие генерировать CSS-код, используя уникальный синтаксис, присущий только конкретному препроцессору.

По сути, препроцессоры — это отдельные скриптовые языки, у каждого из которых свое представление о том, как должен работать CSS, и свой стек возможностей (хоть и довольно часто пересекающихся между собой).

Они нужны, чтобы сделать CSS чище и в заметной степени сократить количество кода, которое приходится писать для оформления сайта. А в некоторых случаях, чтобы добавить в CSS функции, по умолчанию отсутствующие в языке.

Код, написанный на препроцессорах схож с кодом CSS, но на для использования в браузере его необходимо преобразовать в чистый CSS с помощью компилятора.

Наиболее широко используемые CSS препроцессоры: SASS, Stylus, LESS, PostCSS.

Выбор препроцессора зависит от требований по интеграции с другими программными средствами, задействованными в проекте, определяются особенностями конкретного проекта и личными предпочтениями.

**Фреймворки и библиотеки JavaScript** делают процесс разработки веб-приложений эффективнее и быстрее. Обладая широкой функциональностью, они снабжают фронтенд разработчиков универсальными инструментами для решения задач по:

- оптимизации работы с DOM,
- тестированию, анализу правильности и лаконичности написания кода за счет экосистемы пользователей, создавших широкий спектр готовых решений,
- обеспечению предсказуемости и удобства поддержки приложений,
- улучшению взаимодействие частей приложения друг с другом и масштабируемости кода благодаря компонентному подходу,
- упрощение маршрутизации.

Именно эти возможности фреймворков JavaScript и делают их востребованными. Неудивительно, что многие современных веб-приложения, выполнены с использованием фреймворков. Но не стоит преувеличивать универсальность фреймфорков. В первую очередь, нужно оценивать целесообразность их использования в каждой конкретной ситуации учитывая следующие факторы:

- индивидуальные требования проекта,
- способность и готовность всей команды разработчиков работать с конкретным фреймворком,
- наличие знаний у специалиста и понимание особенностей работы фреймворка,
- рациональность использования.

Фреймворки начали активно развиваться начиная с 2010 года. В настоящий момент на рынке существует множество JS-фреймворков, каждый из которых имеет свои особенности и преимущества. Рассмотрим список самых популярных фреймворков.

## *React*

Самая популярная JavaScript-библиотека, создала ее Facebook. React использует компонентную модель и виртуальный DOM для создания масштабируемых и

быстрых веб-приложений. Фреймворк обладает активной экосистемой плагинов и библиотек, таких как Redux и React Router, что расширяет его возможности в управлении состоянием и навигацией. Это делает React идеальным выбором для создания современных интерактивных веб-приложений.

### *Angular*

Фреймворк от Google, который предлагает полный набор инструментов для разработки сложных веб-приложений. Он обладает мощными возможностями тестирования и развертывания, а также использует двустороннее связывание данных для автоматического обновления пользовательского интерфейса при изменении модели данных и наоборот. Благодаря своей комплексной функциональности и подходу к разработке, Angular стал популярным выбором для создания крупных и сложных веб-приложений, особенно в коммерческой сфере.

### *Vue.js*

Легко интегрируемый фреймворк с простой синтаксической моделью. Vue.js пользуется популярностью благодаря своей простоте и понятности, подходит для разработчиков разного уровня. Он использует реактивный подход, отслеживая изменения данных и обновляя соответствующие части пользовательского интерфейса. Удобные инструменты позволяют создавать компоненты для множества задач. Vue.js обладает чистым и интуитивным синтаксисом, ускоряя разработку и поддержку приложений.

### *Ember.js*

Мощный JavaScript-фреймворк, предназначенный для разработки пользовательских интерфейсов. Ember.js имеет встроенные инструменты для работы с базой данных и серверным взаимодействием. Он активно поддерживается сообществом разработчиков и предлагает разнообразие расширений и плагинов. Ember.js является отличным выбором для разработки

крупных и масштабируемых веб-приложений, где важна структура, организация кода и эффективное управление состоянием.

### *Node.js*

Среда выполнения JavaScript, основанная на движке Chrome V8. Она предоставляет широкий набор инструментов и библиотек для создания серверных приложений. Главное преимущество Node.js — его асинхронная и событийно-ориентированная модель выполнения, обеспечивающая высокую эффективность обработки запросов. Он также обладает простым доступом к файловой системе и возможностью работы с сетевыми протоколами. Благодаря активному сообществу и широкому выбору модулей, Node.js популярен в разработке серверных приложений.

Конечно, это только некоторые из самых популярных фреймворков JavaScript. Сложно выделить лучший среди них, поэтому разработчики могут выбрать фреймворк в зависимости от своих потребностей, опыта и предпочтений.

Хотя библиотеки и фреймворки во многом схожи между ними есть отличия. Библиотека — хранилище фрагментов кода, используемого разработчиками в приложении для решения ограниченного набора задач или конкретной задачи. Фреймворк же создает каркас приложения, на основе которого и выстраивается вся функциональность приложения.

Приложение может быть создано, с использованием только фреймворка или библиотеки, но на практике чаще всего придерживаются комбинированного подхода и используют возможности обоих инструментов.

## **1.7. Этапы выполнения проектов по созданию веб-сайтов.**

Обычно выделяют четыре основных этапа разработки веб-сайта:

1. подготовка,
2. проектирование,
3. разработка,
4. поддержка.

На первом этапе устанавливаются цели и задачи создания веб-сайта, исследуется целевой рынок, изучаются потребности потенциальных клиентов, анализируется деятельность конкурентов и определяется общая концепция сайта с учетом основного функционала и ассортимента продукции, на создаваемом сайте. Результатом этого этапа является техническое задание для проектирования веб-сайта.

На этапе проектирования разрабатывается архитектура и дизайн сайта. В этом процессе участвует дизайнер, создающий внешний вид сайта. Также могут привлекаться специалисты по SEO и контент-менеджеры для оптимизации структуры сайта и наполнения его информацией.

Программисты приступают к работе на следующем этапе, реализуя проект с помощью программного кода. Фронтэнд-разработчик занимается интерфейсной частью проекта, а бэкэнд-специалист отвечает за функциональную часть сайта с использованием необходимых технологий. Важным этапом является тестирование веб-сайта, проводимое специалистом по тестированию программного обеспечения. Разработка завершается размещением сайта на сервере, обучением заказчика пользоваться сайтом и сдачей проекта.

Этап поддержки включает техническую поддержку, исправление ошибок, добавление нового функционала, а также рекламу и продвижение сайта в интернете. Каждый этап выполняется специалистами с учетом сложности проекта и других факторов. Существует несколько моделей и методологий управления созданием IT-продуктов, которые могут применяться при разработке веб-сайтов. Выбор оптимального подхода важен для сроков и бюджета проекта. Различные модели разработки включают в себя waterfall, V-образную, спиральную, интерактивную и гибкие модели.

## **2. Практическая реализации задачи по созданию веб-сайта.**

### **2.1. Программные продукты и инструменты веб-разработки.**

Любая разработка начинается с выбора среды разработки. Интегрированные среды (IDE) предоставляют комплексную среду для разработки, тестирования и

развертывания программного обеспечения. IDE включают в себя редактор кода, компилятор или интерпретатор, инструменты отладки, а также другие функции, такие как интеграция контроля версий и управление проектами.

Разработчики часто предпочитают IDE, потому что они обеспечивают универсальное решение для всех аспектов процесса разработки. IDE позволяют писать и редактировать код, компилировать и запускать код, а также устранять ошибки в одной среде. Они также предоставляют дополнительные функции, такие как завершение кода, подсветка синтаксиса и инструменты отладки, ускоряющие и повышающие эффективность разработки.

Однако некоторые разработчики предпочитают использовать текстовые редакторы и инструменты командной строки из-за их гибкости и возможности настройки в соответствии с потребностями разработки. Выбор между IDE и текстовым редактором зависит от личных предпочтений и требований проекта.

Наиболее популярные примеры IDE включают в себя:

- Visual Studio: интегрированная среда разработки от Microsoft, поддерживающая множество языков программирования.
- Eclipse: IDE с открытым исходным кодом, поддерживающая различные языки программирования, включая Java, C++ и Python.
- Xcode: разработана Apple для создания приложений для MacOS, iOS, watchOS и tvOS.
- IntelliJ IDEA: IDE от JetBrains с поддержкой Java, Kotlin и других языков.
- PyCharm: также от JetBrains, ориентирована на разработку Python.

Для разработки дипломного проекта выбран **Visual Studio Code (VS Code)** — бесплатный редактор кода с открытым исходным кодом, предоставляющий множество возможностей для веб-разработки, машинного обучения и других задач программирования. VS Code поддерживает расширения и плагины, позволяющие настроить его под конкретные нужды разработчика. Он поддерживает множество языков программирования и предлагает широкие функциональные возможности.

VS Code является гибким и мощным редактором кода с активным сообществом пользователей. Он обладает функциями, такими как Intellisense,строенная поддержка Git, отладка, настройка тем и поддержка нескольких языков программирования. Этот редактор обеспечивает эффективную работу разработчиков и может быть настроен под конкретные потребности проекта.

**Сервис GitHub**, разработанный компанией Microsoft, был выбран для работы с удаленными репозиториями.

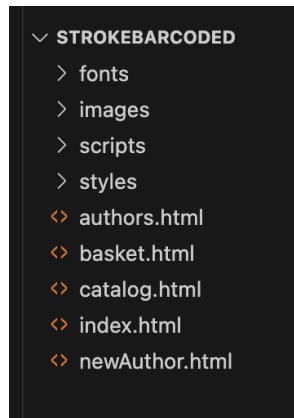
GitHub — популярная платформа, на которой к концу 2023 года было зарегистрировано более 100 миллионов разработчиков и более 4 миллионов организаций, создавших свыше 330 миллионов репозиториев.

Для дипломного проекта на GitHub был создан новый публичный репозиторий, используемый в ходе работы над проектом.

## 2.2. Верстка страниц сайта.

Файлы практической части работы хранятся в специально созданной для этого папке StrokebarCoded (англоязычный вариант названия Штрихкод).

Структура файлов в папке выглядит следующим образом:



- папка fonts – нестандартные шрифты, используемые в проекте,
- папка images – изображения,
- папка scripts – файлы с JavaScript кодом для реализации интерактивных функций,
- папка styles – CSS и SCSS файлы для стилизации контента,

- файлы внутренних страниц сайта: authors.html, basket.html, index.html, newAuthor.html, catalog.html – файлы с HTML кодом.

Для создания структуры страницы в VS Code создается файл \*.html – файл с HTML кодом, в котором задается базовая структура страницы сайта.

Определяется тип документа, язык содержимого, кодировка и заголовок страницы в браузере.

В качестве примера рассмотрим фрагмент кода главной страницы:

```

1  <!DOCTYPE html>
2  <html lang="ru">
3
4  <head>
5      <meta charset="UTF-8">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <title>Homepage</title>
8      <link rel="stylesheet" href="styles/normalize.css">
9      <link rel="stylesheet" href="styles/style.css">
10     <script src="scripts/tabs.js" defer></script>
11     <script src="scripts/basketCounter.js" defer></script>
12 </head>
13

```

В теге <head> подключены основный файлы со стилями style.css, normalize.css и файлы с JavaScript кодом, прописаны относительные пути к этим файлам.

Разбиение на блоки:

```

14  <body>
15  >     <header class="header center">...
58
59     </header>
60     <main>
61         <section class="top center">...
72         </section>
73         <section class="catalog center">...
270        </section>
271        <section class="new-author center">...
283        </section>
284        <section class="authors center">...
318        </section>
319    </main>
320    <footer class="footer center">...
353    </footer>
354 </body>

```

В структуре макета выделяем основные блоки:

- шапка сайта <header>
- основное содержимое сайта <main>
- подвал сайта <footer>

Стоит отметить, что HTML использует теги (tags) для определения элементов страницы. Теги создаются с использованием угловых скобок <>, и обычно имеют начальный и конечный тег, где конечный тег отличается от начального символом "/".

Внутри тегов могут находиться атрибуты (attributes), которые определяют дополнительные характеристики элементов, такие как их цвет, размер, ссылки на другие страницы и т.д.

Задача тегов сделать код читаемым, логичным и структурированным. Вот основные теги проекта:

1. <html>: начальный тег для HTML-документа.
2. <head>: содержит метаданные документа, такие как заголовки страницы, ссылки на стили CSS, и другие данные, которые не отображаются на странице.
3. <body>: определяет содержимое страницы, которое отображается в браузере.
4. <section> : Определяет раздел в документе
5. <title>: определяет заголовок документа.
6. <div>: определяет блок элементов на странице.
7. <p>: определяет абзац текста.
8. <img>: определяет изображение на странице.
9. <a>: определяет ссылку на другую страницу или ресурс.
- 10.<form>: определяет форму на странице, которую пользователь может заполнить и отправить.
- 11.<button>: определяет нажатую кнопку

В современной верстке для распределения элементов на странице чаще всего используются теги `<div>` и `<section>`.

`<div>` — это тег в HTML, который используется для определения контейнеров на веб-страницах. Он создает блоки, в которые можно помещать другие элементы, такие как текст, изображения, видео, таблицы и т.д. `<div>` не предназначен для определения семантики содержимого, а скорее служит для оформления и стилизации. Он не влияет на структуру документа, но может использоваться для группировки и организации элементов страницы.

Пример использования тега `<div>`:

```
<div class="catalog_heading">
    <h2 id="catalog" class="catalog_title">Работы авторов</h2>
    <div class="tab">
        <button class="tab_button tab--active">Глеб Баранов</button>
        <button class="tab_button ">Антон Лялин</button>
        <button class="tab_button">Марк Марков-Гринберг</button>
    </div>
</div>
```

Здесь создается блок, внутри которого находится заголовок второго уровня и три кнопки. Контейнер `<div>` позволяет объединить эти элементы, чтобы упростить их оформление и стилизацию. `<div>` можно также использовать для разделения элементов страницы на группы, например для создания секций или блоков с футером и хедером. В сочетании с CSS стилями, `<div>` может быть мощным инструментом для создания разнообразных макетов и оформления веб-страниц.

`<section>` — это используемый для определения разделов на веб-страницах. Как правило, он содержит группы связанных элементов: заголовки, параграфы, изображения и другие элементы. `<section>` обычно используется для структурирования документа и улучшения доступности и семантической структуры.

Пример использования тега `<section>`:

```
<section class="catalog center">
    <div class="catalog_heading">...</div>
    <div class="catalog_box">...</div>
</section>
```

Здесь создается раздел страницы, содержащий два блока div. Контейнер `<section>` позволяет группировать эти элементы, чтобы указать, что они относятся к одному разделу страницы.

`<section>` обычно используется в сочетании с другими семантическими тегами, такими как `<article>`, `<header>`, `<footer>` и `<nav>`, чтобы создать структурированную и понятную семантическую разметку веб-страницы. `<section>` также может использоваться для стилизации страницы с помощью CSS. Он позволяет определять общие стили для групп связанных элементов и упрощает оформление веб-страницы.

Понимание тегов является необходимым для создания и правильного размещения элементов на странице. Например, правильное использование тегов позволяет браузеру корректно интерпретировать содержимое страницы и отображать его правильно. Также это может повлиять на поисковую оптимизацию (SEO), что важно для любого веб-сайта.

Написание сайта с опорой на смысловое предназначение компонентов и логическую структуру документа есть семантический подход. Целью такого подхода является оптимизация разметки страницы с точки зрения семантики, что представляется важным фактором при создании сайта так как позволяет:

- написать сайт более доступным для пользователей с ограниченными возможностями
- обеспечить более высокое ранжирование в поисковой выдаче
- сделать код более понятным и структурированным, что облегчает

командную разработку и последующую поддержку сайта

Кроме этого, стратегия построения разметки страницы направлена на обеспечения максимальной отзывчивости и адаптивности сайта под различные типы мобильных устройств.

## **2.3. Стилизация страниц сайта.**

Стилизация HTML страницы выполнена с использованием препроцессора SASS.

Для работы с SASS создан файл style.scss и установлены необходимые расширения в редакторе VS Code (1. Live Sass Compiler 2. Sass 3. Live Server). В проекте используется функционал SASS: вложенность, фрагментирование, переменные, импорт и тп.

Стили в SASS написана в синтаксисе SCSS.

Переменные и миксины, используемые в проекте располагаются в файле \_vars.scss, который подключается к основному файлу со стилями с помощью директивы @import ‘vars’;

Пример миксина, используемого при стилизации:

```
@mixin font_styles {  
    color: $text_color_1;  
    font-style: normal;  
    font-weight: 500;  
    line-height: normal;  
}
```

Стили, относящиеся к HTML коду, который повторно используется на внутренних страницах сайта, в частности шапка и подвал вынесены в отдельные файлы, подключаемые к основному файлу со стилями с помощью директивы @import.

## **2.4. Методология БЭМ для верстки и стилизации сайта.**

Определение стилей для элементов HTML выполнено с применением селекторов класса при этом название классов задаются на основе методологии БЭМ (Блок-Элемент-Модификатор).

Методологию БЭМ придумали и начали развивать в компании Яндекс с 2005 года для улучшения исполнения собственных проектов. Методология показала

свою эффективность во множестве проектов как Яндекс, так и других компаний, и сейчас широко используется разработчиками по всему миру.

Блок - Логически и функционально независимый компонент страницы.

Элемент - Составная часть блока, которая не может использоваться в отрыве от него.

Модификатор - сущность определяющая внешний вид, состояние и поведение блока или элемента.

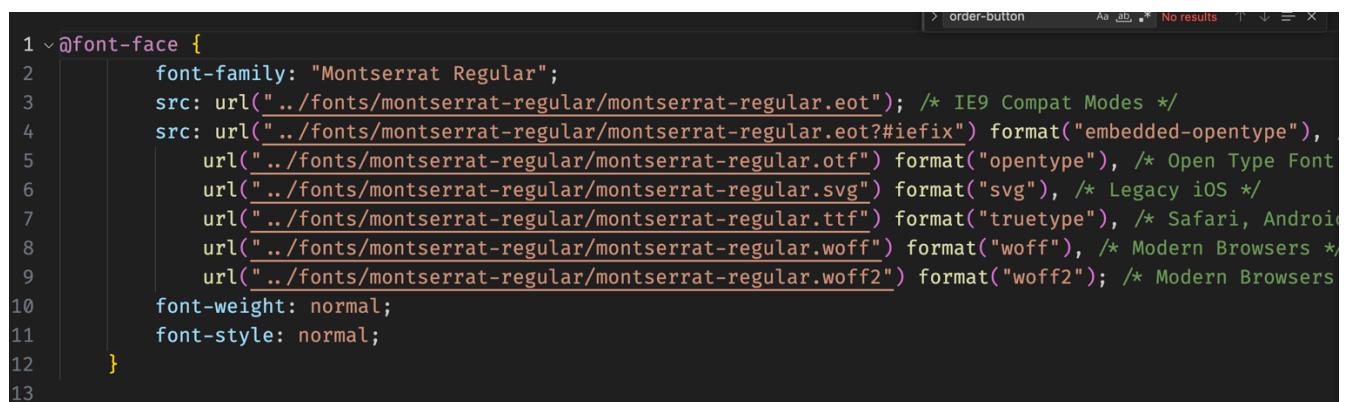
Положительные эффекты этой методологии проявляются в дипломном проекте:

- наглядная структура проекта,
- единый способ именования классов,
- легкая читаемость кода,
- повторное использование частей кода.

Использование других типов селекторов при стилизации сведено к минимуму.

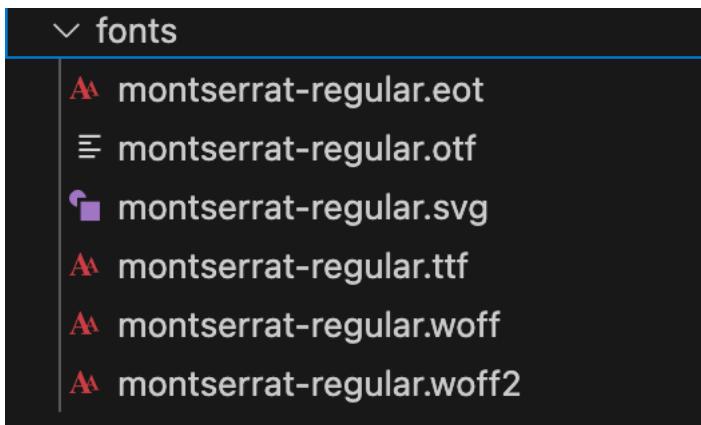
## 2.5.Загрузка и подключение шрифтов.

В качестве шрифта для проекта был выбран: Montserrat, находящийся в свободном доступе на ресурсе <https://fonts.google.com/>. Для подключения данного шрифта к проекту используется правило @font-face и генератор шрифтов приложения transfonter.org, который на основе загруженного шаблона шрифта создает шрифты в нужных форматах и формирует @font-face и правило для подключения к проекту.



```
1 @font-face {
2     font-family: "Montserrat Regular";
3     src: url("../fonts/montserrat-regular/montserrat-regular.eot"); /* IE9 Compat Modes */
4     src: url("../fonts/montserrat-regular/montserrat-regular.eot?#iefix") format("embedded-opentype"),
5          url("../fonts/montserrat-regular/montserrat-regular.otf") format("opentype"), /* Open Type Font */
6          url("../fonts/montserrat-regular/montserrat-regular.svg") format("svg"), /* Legacy iOS */
7          url("../fonts/montserrat-regular/montserrat-regular.ttf") format("truetype"), /* Safari, Android */
8          url("../fonts/montserrat-regular/montserrat-regular.woff") format("woff"), /* Modern Browsers */
9          url("../fonts/montserrat-regular/montserrat-regular.woff2") format("woff2"); /* Modern Browsers */
10    font-weight: normal;
11    font-style: normal;
12 }
```

Сгенерированные в [transfonter.org](http://transfonter.org) шрифты в форматах TTF, OTF, WOFF, WOFF2, SVG, для корректного отображения шрифта Montserrat в разных браузерах, сохранены в папке fonts.



Правило @font-face прописано для всех указанных типов шрифтов в файле \_fonts.scss, который подключается к основному Sass-файлу - homepage.scss через директиву @import ‘fonts’;

## 2.6. Использование изображений.

В проекте применяются два типа изображений — растровые и векторные. Все они хранятся в папке images и подключаются через относительные ссылки. Векторные изображения в формате svg использовались для логотипа и иконок соцсетей и мессенджеров. Остальные изображения, преимущественно фотографии, — в формате jpeg. Растровые изображения были оптимизированы в онлайн-сервисе TinyPNG, что позволило увеличить скорость работы сайта, снизив «вес» каждого изображения в среднем на 50% без потери качества изображения.

Для добавления изображений на страницу используются следующие способы:

- тег <img> в HTML коде как для jpeg изображений
- тег <svg> в HTML коде
- через свойство background-image в CSS при использовании фонового изображения.

## 2.7. Нормализация настроек.

Для улучшения кроссбраузерности сайта подключен файл normalize.css в теге <head> файла index.html. Файл с css-кодом можно скачать с сайта [github.com](https://github.com). Normalize.css — альтернатива CSS Reset, предполагающему сброс исходных настроек.

Проект normalize.css решает следующие задачи:

- сохранение полезных настроек браузера, а не удаление их,
- нормализация стилей для многих HTML-элементов,
- корректировка ошибок и основные несоответствия браузеров,
- повышение удобства использования.

## 2.8. Создание интерактивных элементов главной страницы сайта.

Интерактивность элементов сайта организована как за счет применения функционала HTML5, CSS3 так и JavaScript.

Так, для кнопок были реализованы эффекты наведения с использованием CSS3: изменение стилей элементов при наведении мышью.

Переходы по странице сайта при нажатии кнопок меню навигации реализованы через элемент html <a>, создающим гиперссылки на нужные места главной страницы через якорные *id*, присвоенные целевым элементам.

```
23  <nav class="header__nav">
24    <ul class="header__ul">
25      <li class="header__li"><a href="homepage.html#catalog" class="header__link">Каталог</a>
26      </li>
27      <li class="header__li"><a href="homepage.html#new" class="header__link">Новинки</a></li>
28      <li class="header__li"><a href="authors.html" class="header__link">Наши авторы</a></li>
29      <li class="header__li"><a href="homepage.html#about" class="header__link">О нас</a></li>
30    </ul>
31  </nav>
```

Стандартные псевдоклассы *hover*, *focus*, *active* позволяют в полной мере выполнить задачу придания нужного стиля элементу при изменении его состояния, а возможность вложенности классов SASS придают хорошую читаемость коду.

```

66  &_button {
67      @include button;
68      padding: 20px 60px;
69      background-color: $background_common;
70      color: $text_color_1;
71      border: 1px solid $text_color_1;
72
73      &:hover {
74          background-color: $text_color_1;
75          color: $background_common;
76          border: solid 1px transparent;
77      }
78      &:focus {
79          background-color: $header_footer_color;
80          color: $text_color_1;
81          border: solid 1px $text_color_1;
82      }
83      &:active {
84          background-color: $button_bgc_nav_active;
85          color: $background_common;
86          border: solid 1px transparent;
87      }
88  }

```

## Переключение вкладок

### Работы авторов

Глеб Баранов

Антон Лялин

Марк Марков-Гринберг



серия «Kingdom of Angels»

#### Ангелы и шкатулка

Холст, темпера

64 500 руб

[В корзину](#)



серия «Kingdom of Angels»

#### Ангел сновидений

Холст, темпера

36 500 руб

[В корзину](#)



серия «Kingdom of Angels»

#### Черный ангел

Холст, темпера

30 000 руб

[В корзину](#)

Структура блока с кнопками и карточками Раздела «Работы авторов» в html:

```
76 <div class="tab">
77   <button class="tab__button tab--active">Глеб Баранов</button>
78   <button class="tab__button">Антон Лялин</button>
79   <button class="tab__button">Марк Марков-Гринберг</button>
80 </div>

82 <div class="catalog__box">
83   <div class="catalog__items content--active">...
144   </div>
145   <div class="catalog__items ">...
206   </div>
207   <div class="catalog__items">...
268   </div>
269 </div>
```

После верстки и стилизации этой части сайта, задача переключения вкладок реализуется на JavaScript в файле tabs.js.

По умолчанию активной является первая кнопка, имеющая класс *tab--active* и блок с вкладками с классом *content--active*.

Переключение вкладок осуществляется за счет добавление класса *tab--active* и *content--active* на нажатую кнопку и соответствующую ей вкладку по событию “*click*” на любой из кнопок с классом *class="tab\_\_button"*.

С помощью метода *querySelectorAll()* по названию соответствующего класса получаем список всех кнопок и вкладок.

```
4 const tabElements = document.querySelectorAll(".tab__button");
5 const contentElements = document.querySelectorAll(".catalog__items");
```

Так как переключение вкладок осуществляется также из навигационного меню подвала сайта, то относящийся к переключению вкладок код обрамляется в функцию *function tabsSwitch(buttonNames)* и может быть повторно использован, например, при создании функционала переключения вкладок в нижнем или верхнем меню. В качестве аргументов в эту функцию передаются название класса кнопок, с которых будут переключаться вкладки.

На все элементы списка *tabElements* при помощи цикла добавляется обработчик событий *click*.

При наступлении события *click* производится удаление классов *tab--active* и *content--active* методом *forEach* со всех элементов списка кнопок и вкладок и затем добавление этих классов кнопке, на которой сработало событие и соответствующей ей вкладке.

Поскольку действия по добавлению и удалению могут использоваться в других частях кода, для них написаны отдельные функции:

```
function addClassToListEl(elementName, className) {  
    elementName.classList.add(className);  
}  
и
```

```
function removeClassFromList(elementName, ListName) {  
    elementName.forEach((element) => element.classList.remove(ListName));  
},
```

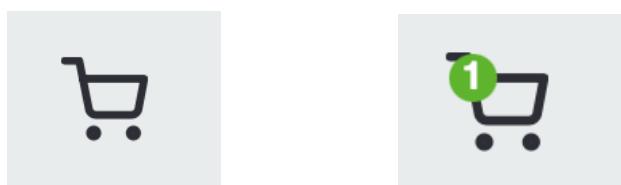
которые вызываются внутри функции *tabsSwitch*.

Счетчик товаров, добавленных в корзину и сохранения данных о товарах в localStorage.

Программа для отображения добавленного в корзину количества товаров и сохранения данных о них в хранилище браузера *localStorage* на главной странице сайта написана на JavaScript и находится в файле *basketCounter.js*.

Иконка корзины, расположенная в правом верхнем углу страницы сайта отображается без значка с количеством товаров, пока в корзину не добавлен хотя бы один товар.

При нажатии на кнопку “В корзину” карточки товара, на корзине появляется цифра с количеством товаров в корзине.



Добавление информации о выбранных товарах в localStorage выполняется следующим образом:

- Id товара, указанный в data-number карточки товара добавляется в массив и далее данные из массива сохраняются в виде строки в localStorage используя метод `setItem(key, value)` и `JSON.stringify()`
- Начальное количество товаров, равное одной штуке для всех выбранных товаров добавляется в массив `productItemQtyArray` и далее данные из массива сохраняются в виде строки в localStorage, используя метод `setItem(key, value)` и `JSON.stringify()`
- Цена одного товара, указанная в data-price карточки товара добавляется в массив `productItemPriceArray` и далее данные из массива сохраняются в виде строки в localStorage используя метод `setItem(key, value)` и `JSON.stringify()`
- Значение счетчика числа товаров используя метод `setItem(key, value)`

Сохраненная в localStorage информация в последствии используется на странице Корзины для реализации функционала этой веб-страницы.

Кроме этого, клик кнопки “В корзину”, вызывает изменение текста на кнопке карточки товара с “В корзину” на “Товар в корзине”. В случае повторного нажатия на кнопку “Товар в корзине” увеличения числа товаров в корзине и записи данных в localStorage не происходит.



серия «Kingdom of Angels»

**Ангелы и шатер**

Холст, темпера

120 000 руб

[Товар в корзине](#)



серия «Kingdom of Angels»

**Ангелы и шкатулка**

Холст, темпера

40 000 руб

[В корзину](#)



серия «Kingdom of Angels»

**Синий ангел**

Холст, темпера, акрил

45 000 руб

[Товар в корзине](#)

Данный функционал реализован следующим образом.

Метод querySelector() находит и добавляет элементы по названию класса.

```
5 const headerBasketElement = document.querySelector(".header__basket");
6 const basketCounterElement = document.querySelector(".basket-counter");
7 const catalogBoxElement = document.querySelector(".catalog_box");
8 const cardButtonElement = document.querySelector(".card_button");
9
```

Добавляется событие “click” на элемент с каталогом карточек с помощью метода addEventListener.

В случае одновременного выполнения двух условий – нажатие происходит на элементе с классом «card\_button» и текст внутри «В корзину» совершаются следующие действия:

- счетчик товаров, изначально равный нулю, увеличивается на единицу,
- скрытый символ со счетчиком меняет свойство visibility с hidden на visible, что делает его видимым на странице сайта,
- меняется текст кнопки на «Товар в корзине»,
- меняется стилизация нажатой кнопки за счет замены класса.

```
21 catalogBoxElement.addEventListener("click", (e) => {
22   if (
23     e.target.className === "card_button" &&
24     e.target.innerText === "В корзину"
25   ) {
26     counter++;
27     headerBasketElement.style.visibility = "visible";
28     e.target.innerText = "Товар в корзине";
29     e.target.classList.remove("card_button");
30     e.target.classList.add("catalog_card_button_basket");
31   }
32 }
```

## 2.9. Разработка внутренних страниц сайта.

### 1. Страница «Каталог».

На странице предусмотрен функционал для поиска, фильтрации и добавления товаров в корзину.

Переход с главной страницы сайта на данную страницу сайта осуществляется через кнопку <button> вложенную внутрь элемента <form> с атрибутом action содержащим адрес страницы.

Переход со страницы Каталог на главную страницу выполняется через навигационное меню шапки и подвала сайта кликом по иконке логотип.

## Работы

Глеб Баранов  Антон Лялин  Марк Марков-Гринберг

Сортировка  
По популярности ▾



серия "Портреты Африки"  
**Портрет №1**  
Печать архивными чернилами на хлопковой бумаге  
35 000 руб  
[В корзину](#)



серия "Портреты Африки"  
**Портрет №2**  
Печать архивными чернилами на хлопковой бумаге  
60 000 руб  
[В корзину](#)



серия "Портреты Африки"  
**Портрет №3**  
Печать архивными чернилами на хлопковой бумаге  
120 000 руб  
[В корзину](#)

HTML-код страницы находится в файле catalog.html. Стилизация выполнена с использованием возможностей CSS препроцессора SASS. Файлы со стилями хранятся в папке styles:

```
# catalog.css
# catalog.css.map
♀ catalog.scss
```

Интерактивный функционал реализован с помощью CSS3 и JavaScript. Файлы с кодом JavaScript хранятся в папке scripts: catalog.js, basketCounter.js.

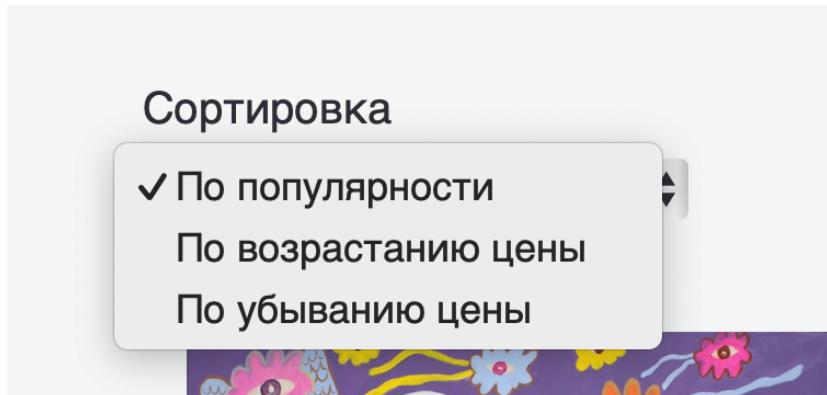
Интерактивный функционал с кодом на CSS3: переходы на другие страницы сайта,

эффекты наведения, меню сортировки, чек-бокс хранятся в файлах .css.

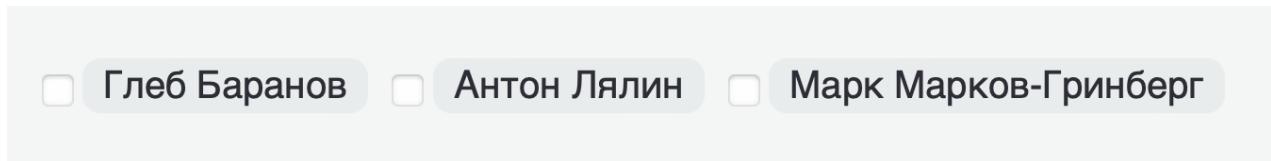
### Интерактивный функционал на JavaScript:

- сортировка товаров по нескольким критериям:

По умолчанию применяется сортировка по популярности.



- фильтрация работ по авторам:



Изначально на странице отображаются все работы всех авторов. При выборе одного или нескольких авторов выводятся принадлежащие ему/им работы. Если все фильтры сняты на экране снова отображается все товары.

Счетчик товаров, добавленных в корзину, и сохранения данных о товарах в localStorage функционирует так же, как и на главной странице сайта.

При нажатии кнопки «В корзину» карточки товара, товар добавляется в корзину и счетчик товаров корзины увеличивается на единицу, а в хранилище браузера localStorage сохраняются данные с номерами карточек выбранных товаров и числом товаров.

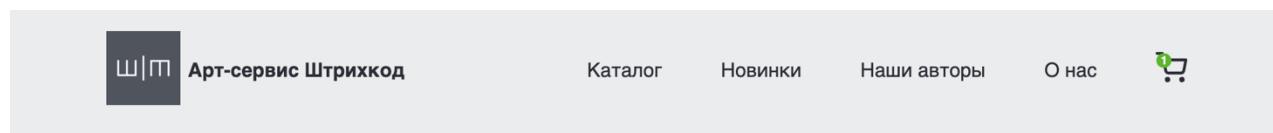
## 2. Страница «Новый автор».

На странице размещен слайдер с избранными фотографиями Сигеру Ёсио.

Переход с главной страницы сайта на данную страницу сайта и обратно выполняется аналогично переходам для страницы «Каталог».

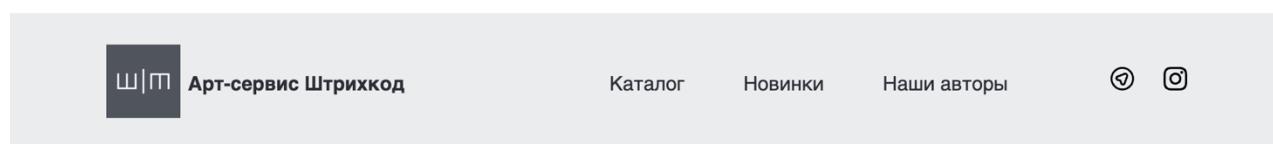
HTML-код страницы находится в файле newAuthor.html. Код для стилей хранятся в папке styles:

```
# newAuthor.css  
# newAuthor.css.map  
♀ newAuthor.scss
```



## Коллекция фотографий Сигеру Ёсида

Серия "Границы"



Файлы с кодом JavaScript хранятся в папке scripts: файлах slider.js и basketCounterElement.js.

Интерактивный функционал на CSS3 — переходы на другие страницы сайта, эффекты наведения.

Интерактивный функционал на JavaScript:

- Слайдер, отображающий фотографии знаменитого японского фотографа. В слайдере реализована возможность переключения изображений как через стрелки «Вправо/Влево», так и через навигационные индикаторы для быстрого переключения между изображениями.

При нажатии на правую стрелку отображается следующее изображение, а при нажатии на левую стрелку отображается предыдущее изображение.

Переключение изображений осуществляется циклически.

При нажатии на навигационные точки, на экран выводится изображение, соответствующее нажатой точке, при этом меняется стиль этой точки.

- Счетчик товаров, добавленных в корзину

При загрузке страницы с помощью программы, из файла basketCounterElement.js методом getItem(key) из localStorage извлекается количество товаров, добавленных в корзину и если полученное значение больше нуля, то на экран выводится это число над значком корзины.

### 3. Страница «Авторы».

На странице предусмотрена возможность просмотра подробной информации о художнике/фотографе в диалоговом окне и различными эффектами анимации. Переходы между страницами осуществляются таким же образом, как и на предыдущих внутренних страницах сайта через шапку и подвальную часть сайта для перехода на главную страницу и при нажатии на соответствующие кнопки/ссылки на остальные страницы сайта.

HTML-код страницы находится в файле authors.html. Стилизация выполнена с использованием возможностей CSS препроцессора SASS. Файлы со стилями хранятся в папке styles:

```
# newAuthor.css  
# newAuthor.css.map  
§ newAuthor.scss
```

## Наши художники и фотографы

Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae facilis atque deserunt ullam, et corporis ex. Voluptas, assumenda impedit? Porro labore architecto nihil hic sit fugiat molestiae. Placeat, quia tenetur!



Иван Бойко  
Фотограф



Анна Аристова  
Фотограф



Улугбек Досчанов  
Художник



Петр Стариков  
График



Альберт Гогуадзе  
Художник



Глеб Баранов  
Художник



Николай Орлов  
Фотограф



Марк Марков-Гринберг  
Фотограф



Сигэру Ёсида  
Фотограф



Антон Лялин  
Фотограф

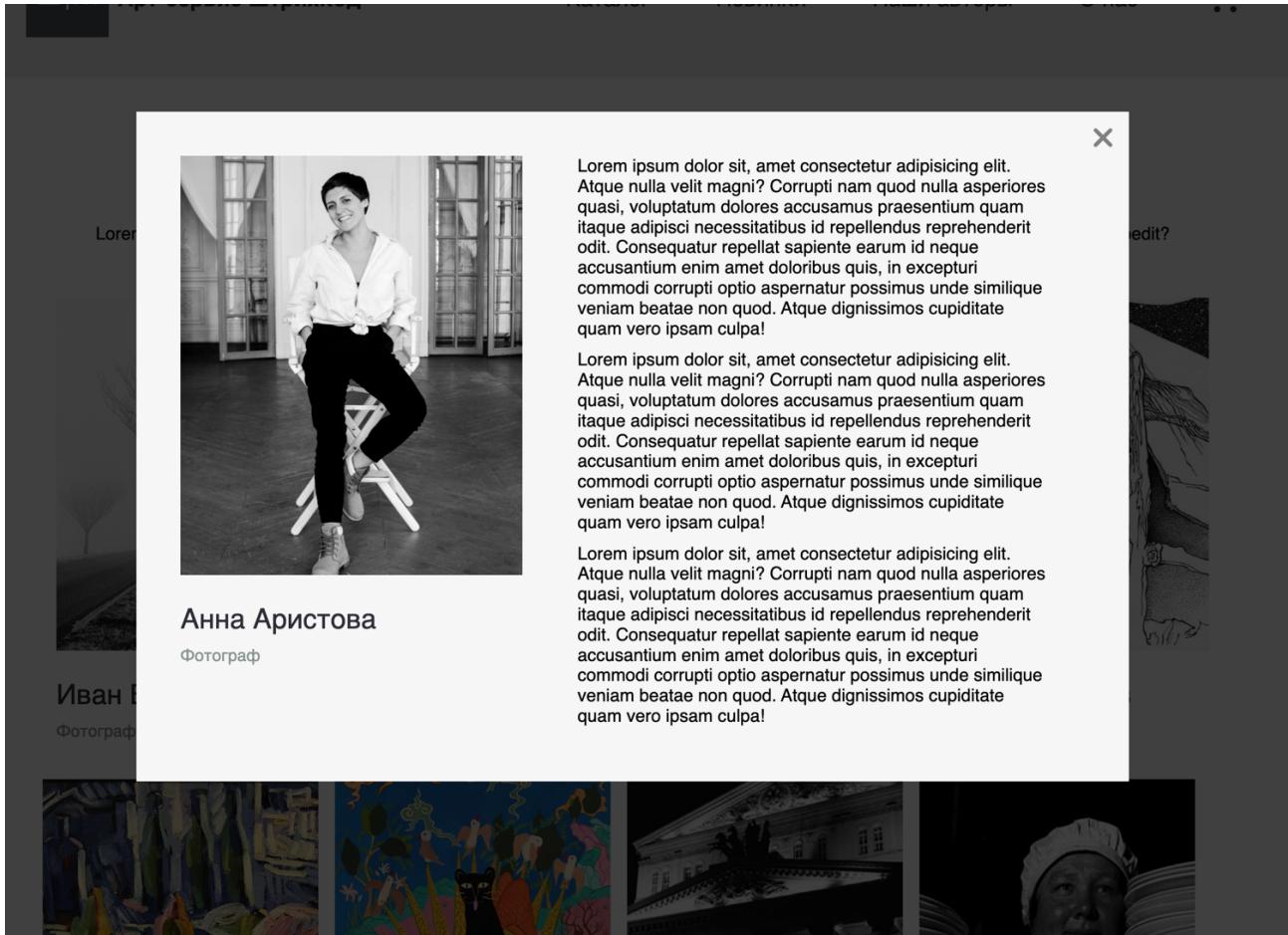
Интерактивный функционал реализован с помощью CSS3 и JavaScript. Файлы с кодом JavaScript хранятся в папке scripts: authors.js и basketCounterElement.js.

Интерактивный функционал на CSS3 — специальные эффекты появления элементов на экране при открытии страницы (фотографии авторов) и открытии диалогового окна, а также разнообразные эффекты наведения — например, увеличение изображения фотографии автора при наведении кнопки мыши.

## Интерактивный функционал на JavaScript:

*Диалоговой окно с подробной информацией о выбранном авторе.*

При клике правой кнопкой мыши по фотографии сотрудника происходит затемнение фона экрана и открывается диалоговое окно с карточкой выбранного автора.



*Счетчик добавленных в корзину товаров тождественен счетчику на странице «Новый автор».*

## **4. Страница «Корзина».**

На этой странице отображаются карточки выбранных товаров с возможностью изменения количества товаров, удаление товара, подсчет общей цены товара и итоговой цены всех товаров в корзине.

Ш|п Арт-сервис Штрихкод Каталог Новинки Наши авторы О нас

## Корзина

	серия «Kingdom of Angels» <b>Ангелы и шатер</b> Холст, темпера	120000 ₽	-	1	+	<a href="#">Удалить</a>
120 000 руб						
	серия «Kingdom of Angels» <b>Синий ангел</b> Холст, темпера, акрил	45000 ₽	-	1	+	<a href="#">Удалить</a>
45 000 руб						

Очистить корзину

[Перейти к оформлению](#)

**Ваш заказ**

Товары	2
Доставка	Бесплатно
<b>Общая сумма заказа</b>	<b>165000 ₽</b>

Ш|п Арт-сервис Штрихкод Работы авторов Новинки Наши авторы

HTML-код страницы находится в файле basket.html. Стилизация выполнена с использованием возможностей CSS препроцессора SASS. Файлы со стилями хранятся в папке styles:

```
# basket.css
# basket.css.map
♀ basket.scss
```

Интерактивный функционал реализован с помощью CSS3 и JavaScript. Файлы с кодом JavaScript хранятся в папке scripts: basket.js и basketCounterElement.js.

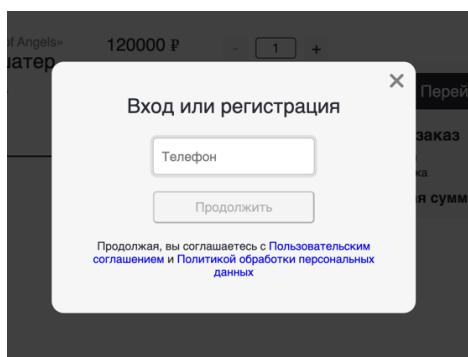
Интерактивный функционал на CSS3 – специальные эффекты появления элементов на экране при открытии модального окна, переходы на другие страницы сайта, а также разнообразные эффекты наведения.

## Интерактивный функционал на JavaScript:

- *Отображение на экране карточек товаров, добавленных в корзину*, на основе данных сохранённых в localStorage — общее количество товаров, порядковые номерах товаров, начальное количество товаров, цена товара
- *Удаление товара с экрана и удаление из localStorage* для данного товара его номера, цены, количества штук, пересчет и обновление общего количества товаров и общей суммы заказа в таблице Ваш Заказ, обновление общего количества товаров в localStorage при нажатии кнопки «Удалить».
- *Увеличение или уменьшение единиц товара* при нажатии кнопок + и – с последующим пересчетом и обновлением стоимости товара, общей стоимости заказа, обновляется количество единиц товара, а также новое число единиц товара и стоимости товара сохраняется в localStorage. Если количество единиц товара равно единице, то кнопка — неактивна.



- *Удаление всех товаров с экрана и всех данных о товаре из localStorage при нажатии на иконку корзины.*
- *Диалоговое окно с формой регистрации.*



При клике правой кнопкой мыши по кнопке происходит затемнение фона экрана и открывается диалоговое окно с формой Авторизации/Регистрации.

При нажатии на значок «X» в правом верхнем углу или любое место экрана кроме диалогового окна диалоговое окно закрывается.

- Вывод на экран *информационного сообщения, что корзина пуста*, при клике на иконку корзины, когда в ней нет товаров.

## Корзина



В корзине пока пусто

- *Счетчик товаров, добавленных в корзину – аналогично странице «Новый автор».*

### 2.10. Адаптивность сайта.

Для обеспечения адаптивности сайта при просмотре на десктопе, двух вариантах планшета и смартфона были разработаны четыре варианта верстки. Каждая из версий оптимизирована под конкретное устройство следующими средствами:

- сокращение количества столбцов на мобильных устройствах,
- изменение вертикальных и горизонтальных отступов,

- изменение размеров и пропорций элементов относительно друг друга в рамках блока,
- размеров изображений,
- перестроение расположения элементов внутри блока,
- сокращение размеров текста и межстрочных интервалов от десктопной к мобильной версии,
- скрытие второстепенных изображений на мобильной версии.

При стилизации HTML-страницы используются приемы, направленные на обеспечение максимальной отзывчивость страницы сайта на любых типах устройств. Так,

- для позиционирования элементов и блоков используются гибкие элементы Grid layout и Flexbox,
- размеры блочных элементов по необходимости изменяются с использованием адаптивных свойств max/min-width и max/min-height,
- размеров элементов, там где необходимо, используются относительные единицы: %, vh, vw,
- центрирование блоков с помощью padding,

Для управления стилями элементов, на заданных макетом значениях ширины устройств настраивается метатег viewport и используются медиазапросы, например: @media (max-width: 460px).

Для подключения тега meta viewport в разделе head файла index.html добавляется строка: <meta name="viewport" content="width=device-width, initial-scale=1.0">.

Для адаптивного дизайна значения атрибута content viewport определяться двумя параметрами:

- width=device-width - ширина видимой области веб-страницы равняется CSS ширине устройства,
- initial-scale=1 - первоначальный масштаб веб-страницы равный 1 означает то, что масштаб равен 100.

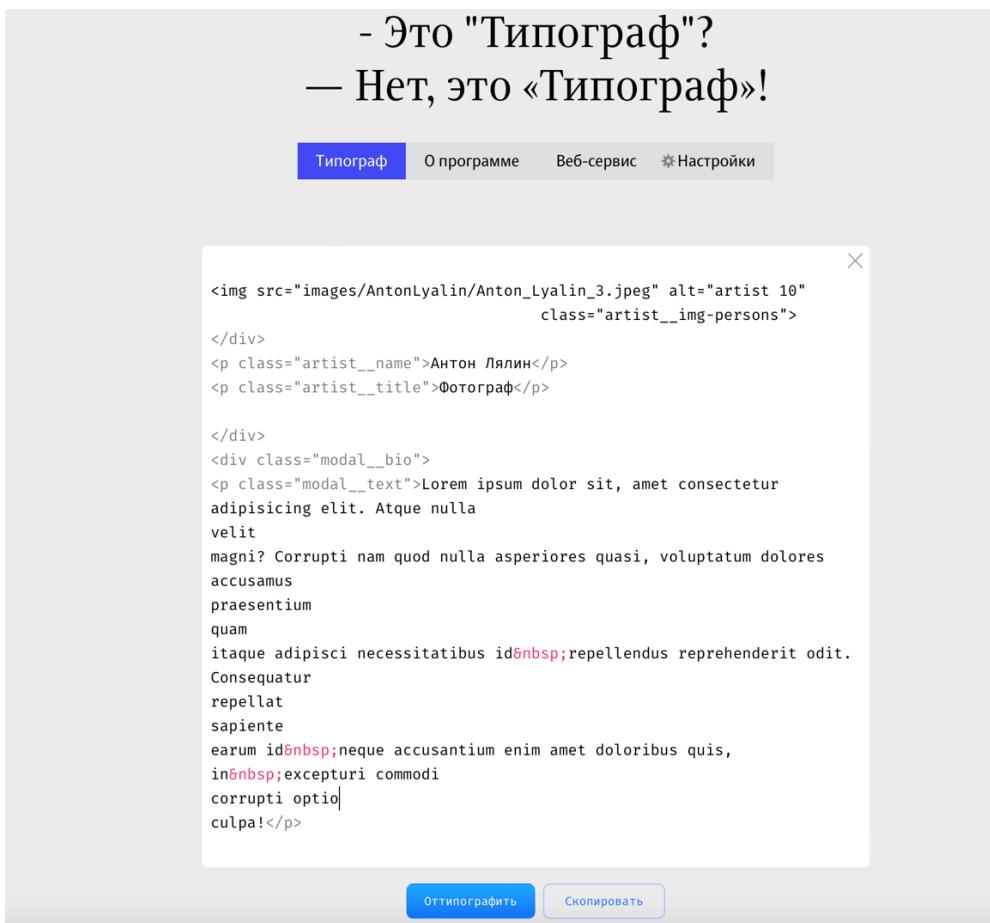
## 2.11. Проверка веб-сайта на ошибки.

### Проверка синтаксиса и типографики текста

Проведена проверка текста на наличие синтаксических ошибок. Ошибок не обнаружено.

Для подготовки типографики текста сайта перед его загрузкой в интернет использована программа «Типограф» студии Лебедева - <https://www.artlebedev.ru/typograf/>

В соответствие с рекомендациями «Типограф» произведено редактирование текста.



### Тестирование кода HTML.

HTML код успешно проверен на валидность, то есть на соответствие стандартам W3C (The World Wide Web Consortium) с использованием онлайн валидатора <https://validator.w3.org/>.

Проверка на валидаторе заключается в следующем:

- проверка синтаксических ошибок
- проверка вложенности тегов
- валидация DTD — анализ кода на соответствие Document Type Definition
- включающая проверку названий тегов, атрибутов, “встраивания” тегов
- проверка на посторонние элементы, отсутствующие в DTD

Чистота HTML кода является важным фактором влияющим на скорость загрузки веб-страницы, корректность отображения содержимого сайта на разных устройствах и браузерах, качество поисковой индексации в поисковых системах.

## Nu Html Checker

This tool is an ongoing experiment in better HTML checking, and its behavior remains subject to change

### Showing results for uploaded file authors.html

Checker Input

Show  source  outline  image report Options...

Check by  file upload  Выбрать файл файл не выбран

Uploaded files with .xhtml or .xht extensions are parsed using the XML parser.

**Document checking completed. No errors or warnings to show.**

Used the HTML parser.

Total execution time 22 milliseconds.

[About this checker](#) • [Report an issue](#) • Version: 24.5.11

## Функциональное тестирование.

До и после размещения сайта в сети интернет проведено тестирование функционала сайта. Тщательно проверялась корректность работы меню навигации, кнопок, анимационные эффекты, переключение вкладок, счетчик товаров в корзине, правильность сохранения и удаления данных хранилища браузера localStorage, правильность расчетов и реагирования на события на

странице «Корзина» и других страницах сайта, открытие и закрытие диалоговых окон, переходы внутри и между страницами.

Так, в ходе ручного тестирования была выявлена следующая ошибка: слайдер на странице «Новый автор» поднимался наверх или опускался при уменьшении и увеличении масштаба страницы в браузере. Проблема была решена удалением параметра: height: 90vh и добавлением полей.

После завершения тестирования все недочеты, обнаруженные в работе сайта, были устранены.

## 2.12. Загрузка веб-сайта в сеть интернет.

Для размещения сайта в интернет использован бесплатный хостинг <https://beget.com/>. Адрес интернет сайта в сети интернет: <http://t91613yz.beget.tech>

Разработка    Окно    Справка

Небезопасно — t91613yz.beget.tech

Homepage

Ш|П Арт-сервис Штрихкод

Каталог    Новинки    Наши авторы    О нас

Каталог

Наша миссия

познакомить коллекционеров и просто ценителей искусства с произведениями талантливых художников и фотографов.

Гарантируем высокое качество отрисовки на плотной бумаге или льняном холсте, малые тиражи и эксклюзивность работ.

Каталог

Работы авторов

Глеб Баранов    Антон Лялин    Марк Марков-Гринберг

## **Заключение**

В результате успешно выполненного дипломного проекта удалось претворить в жизнь все поставленные задачи: создать фронтенд-часть сайта для потребностей конкретного бизнеса. Ввиду отсутствия макета дизайнера и каких-либо определенных пожеланий со стороны будущих владельцев создаваемого сайта, пришлось самостоятельно продумывать макет, учитывая стиль и особенности арт-сервиса Штрихкод.

В первой части, охватывающей теоретические вопросы создания веб-сайтов, осуществлен подробный анализ современных подходов к созданию веб-сайтов, в результате чего определен способ и инструменты для реализации проекта.

Вторая часть описывает практическую сторону реализации проекта: был придуман макет сайта, выполнена верстка, стилизация, добавление интерактивных элементов с использованием HTML5, CSS3, JavaScript и иных инструментов веб-разработки.

Веб-сайт успешно прошел все основные виды тестов уместных для подобных проектов включая тестирование работы на различных типах устройств и кросбраузерное тестирование.

Веб-сайт размещен в сети интернет: <http://t91613yz.beget.tech>.

Одним из главных достижений проекта является умение правильно использовать различные инструменты и технологии для достижения поставленных целей.

Кроме того, были применены современные подходы и методологии в разработке, что позволило получить высокое качество и удобство использования веб-сайта.

Данный проект дал прекрасную возможность получить практический опыт по созданию веб-сайта, закрепить и расширить знания, полученные во время обучения в GeekBrains.

## **Список использованной литературы**

1. Codecademy - <https://www.codecademy.com/learn/learn-html>
2. Codecademy - <https://www.codecademy.com/learn/introduction-to-javascript>
3. Statista - <https://www.statista.com/chart/19058/number-of-websites-online/>
4. Skillbox Media - <https://skillbox.ru/media/marketing/chto-takoe-vebsayt-kak-on-rabotaet-i-kak-sozdayut-sayty/>
5. МакХост - <https://mchost.ru/articles/raznica-mezhdu-domenom-sajtom-i-hostingom/>
6. SkyPro - <https://sky.pro/media/chem-otlichaetsya-staticheskij-i-dinamicheskij-sajt/>
7. TimeWeb - <https://timeweb.com/ru/community/articles/top-5-samyh-populyarnyh-cms-dlya-sayta-1>, <https://timeweb.com/ru/community/articles/o-css-preprocessorah-i-freymvorkah>
8. GeekBrains - <https://gb.ru/blog/chto-takoe-html/>
9. Html academy - <https://htmlacademy.ru/blog/js/introduction-to-javascript>
10. Курс лекций Алексея Кадочникова на Geekbrains – Веб-вёрстка HTML/CSS, Продвинутый HTML + CSS, Основы Javascript, JavaScript про ECMAScript, JavaScript про API браузеров - <https://gb.ru>
11. Курс Лекций Алевтины Шаталовой на Geekbrains Знакомство с веб-технологиями - [https://gb.ru/](https://gb.ru)
12. Learn to Code - <https://www.w3schools.com/>
13. FreeCodeCamp - <https://www.freecodecamp.org/>
14. Документация на русском языке JS - <https://learn.javascript.ru/>
15. Kata academy - <https://kata.academy/article/chto-takoye-js-freymvorki-spisok-luchshikh>
16. Bootstrap - <https://getbootstrap.com/>
17. Популярные названия классов HTML/CSS - [https://dan-it.gitlab.io/fe-book/programming\\_essentials/html\\_css/ext\\_popular\\_css\\_classes\\_name/ext\\_popular\\_css\\_classes\\_name.html](https://dan-it.gitlab.io/fe-book/programming_essentials/html_css/ext_popular_css_classes_name/ext_popular_css_classes_name.html)
18. Шрифты: <https://fonts.google.com/>

19. «Типограф» студии Лебедева -<https://www.artlebedev.ru/typograf/>

20. <https://beget.com/>

21. Как работает Веб - Изучение веб-разработки | MDN -  
<https://developer.mozilla.org/>