КОНЦЕРН БЕЛЛЕГПРОМ РЕСПУБЛИКА БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ «БАРАНОВИЧСКИЙ ГОСУДАРСТВЕННЫЙ КОЛЛЕДЖ ЛЕГКОЙ ПРОМЫШЛЕННОСТИ ИМ. В.Е. ЧЕРНЫШЕВА»

**ОТЧЕТ**

**ПО ТЕХНОЛОГИЧЕСКОЙ ПРАКТИКЕ**

обучающегося 4 курса 516 группы

Александровича Игоря Андреевича

Специальность: 2-40 01 01 «Программное обеспечение информационных технологий»

Специализация: 2-40 01 01 35 «Программное обеспечение обработки экономической и деловой информации»

Место прохождения практики: г. Барановичи, ЗАО «БСЗ Атлант»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Руководитель практики от производства: |  |  |  | В.А. Абашин |
| Руководитель практики от колледжа: |  | подпись |  | Э.С. Борздая |
|  |  | подпись |  |  |
| Обучающийся: |  |  |  | И.А. Александрович |
|  |  | подпись |  |  |
| Защита практики: |  |  |  |  |
|  |  | отметка |  | подписьпринимающего |

Барановичи, 2025

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc169264499)

[1 КРАТКАЯ ХАРАКТЕРИСТИКА МЕСТА ПРОХОЖДЕНИЯ ПРАКТИКИ 4](#_Toc169264500)

[2 ТЕОРЕТИЧЕСКИЕ ОСНОВЫ ВЕБ-КОНСТРУИРОВАНИЯ 5](#_Toc169264501)

[2.1 Понятие веб-конструирования, веб-программирования, вёрстки 5](#_Toc169264502)

[2.2 Языки и технологии, применяемые в веб-программировании 5](#_Toc169264503)

[2.3 Инструменты для вёрстки сайта 6](#_Toc169264504)

[3 РАЗРАБОТКА ВЕБ-САЙТА 9](#_Toc169264505)

[3.1 Физическая и логическая структура сайта 9](#_Toc169264506)

[3.2 Вёрстка главной страницы сайта 10](#_Toc169264507)

[3.3 Использование css 11](#_Toc169264508)

[3.4 Интерактивность сайта 12](#_Toc169264509)

[3.5 Адаптивность веб-дизайна 12](#_Toc169264510)

[ЗАКЛЮЧЕНИЕ 13](#_Toc169264511)

[СПИСОК ЛИТЕРАТУРЫ 14](#_Toc169264512)

[ПРИЛОЖЕНИЕ А 15](#_Toc169264513)

[ПРИЛОЖЕНИЕ Б 16](#_Toc169264514)

[ПРИЛОЖЕНИЕ В 17](#_Toc169264515)

# **ВВЕДЕНИЕ**

Технологическая практика студентов является важной частью учебного процесса. Основная цель такого обучения – закрепление и углубление получен­ных в ходе теоретических занятий знаний, а также формирование профессио­нальных умений и практических навыков.

Задачи, решаемые в рамках технологической практики, включают:

– приобретение студентами профессиональных навыков и умений по вы­бранной специальности;

– закрепление, углубление и систематизация знаний, полученных по про­фильным предметам;

– развитие профессионального мышления и самостоятельности в принятии решений;

– формирование умений организаторской работы в производственной среде и взаимодействия в коллективе;

– повышение уровня квалификации по специальности либо освоение смеж­ных профессий с присвоением разрядов.

В процессе прохождения технологической практики составляются индиви­дуальные планы-графики, по которым студенты распределяются по различным отделам, службам и рабочим местам. Практика нацелена на всестороннюю под­готовку студентов к выполнению профессиональных обязанностей на рабочих местах, соответствующих квалификации специалистов со средним специальным образованием. Важной составляющей качественной подготовки специалистов является последовательное формирование необходимых навыков и умений на каждом этапе практики при соблюдении единого подхода и взаимосвязи всех этапов.

Отчетность по итогам технологической практики включает дневник и письменный отчет, описывающий выполненные задания в соответствии с про­граммой. К отчету могут прилагаться дополнительные материалы.

Технологическая практика проводится в подразделениях автоматизиро­ванных систем управления (АСУ) промышленных предприятий и организаций различного профиля, а также в IT-отделах компаний. В моем случае практика проходит в ЗАО «БСЗ» (Барановичский станкостроительный завод) Атлант и продлится с 11.11.2024 по 03.01.2025.

# **1 КРАТКАЯ ХАРАКТЕРИСТИКА МЕСТА ПРОХОЖДЕНИЯ ПРАКТИКИ**

Барановичский техникум легкой промышленности был основан в июле 1960 года. Развивающаяся текстильная промышленность Беларуси остро нужда­лась в увеличении числа специалистов в этой области. Для удовлетворения этой потребности и был создан техникум, первым директором которого стал Юрий Александрович Циреня. В 1970 году техникум получил имя Василия Ефимовича Чернышева. С 1 ноября 1999 года техникуму был присвоен статус колледжа.

В 2006 году колледж открыл набор на новую специальность 2-40 01 01 «Программное обеспечение информационных технологий» с квалификацией техник-программист.

Техник-программист для автоматизированных производственных систем подготавливается для участия в разработке и эксплуатации программного обес­печения на производстве.

Главной особенностью этой специальности является углубленное изуче­ние вопросов создания, развертывания и конфигурирования специализирован­ного программного обеспечения и интернет-сайтов, учитывая вопросы безопас­ности и дальнейшего сопровождения.

Кроме того, выпускники научатся работать в коллективе, эффективно об­щаться с коллегами и руководством, а также использовать информационно-ком­муникационные технологии в профессиональной деятельности.

Содержание обучения постоянно обновляется в соответствии с изменени­ями в сфере высоких IT-технологий и перспективами развития цифровой эконо­мики в нашей стране.

Программистов можно с гордостью назвать «элитой» колледжа. Специа­листы, способные создавать программы от начала до конца, высоко ценятся се­годня.

# **2 ТЕОРЕТИЧЕСКИЕ ОСНОВЫ ВЕБ-КОНСТРУИРОВАНИЯ**

# **2.1 Понятие веб-конструирования, веб-программирования, вёрстки**

Веб-конструирование, веб-программирование и верстка являются ключе­выми аспектами создания и разработки веб-сайтов.

Веб-конструирование включает в себя проектирование и создание струк­туры веб-сайта. Это процесс планирования и разработки визуального макета, определение навигации, расположения элементов, а также создание пользова­тельского интерфейса. Веб-конструирование направлено на обеспечение удоб­ства использования сайта и его привлекательности для посетителей.

Веб-программирование – это процесс написания кода, который обеспечи­вает функциональность веб-сайта. Оно включает использование различных язы­ков программирования, таких как HTML, CSS, JavaScript и других. Веб-програм­мисты создают интерактивные элементы, такие как формы, анимации, обработку данных на стороне клиента и сервера, а также обеспечивают безопасность и ста­бильность работы сайта.

Верстка – это процесс преобразования дизайна веб-сайта в код. Верстка отвечает за точное отображение макета на экране пользователя. Она включает использование HTML для создания структуры страницы и CSS для оформления визуального стиля. Верстальщики стремятся к тому, чтобы сайт выглядел одина­ково хорошо на различных устройствах и в разных браузерах, обеспечивая крос­сбраузерную совместимость и адаптивность дизайна.

Все три процесса – веб-конструирование, веб-программирование и верстка – тесно связаны и взаимодополняют друг друга, что позволяет создавать совре­менные, функциональные и эстетически привлекательные веб-сайты.

# **2.2 Языки и технологии, применяемые в веб-программировании**

Веб-программирование включает в себя множество языков и технологий, которые используются для создания и поддержания веб-сайтов и веб-приложе­ний. Каждый из этих инструментов имеет свою уникальную роль и предназначе­ние.

HTML (HyperText Markup Language) является основным языком для созда­ния структуры веб-страниц. С его помощью определяется, какие элементы будут присутствовать на странице, такие как заголовки, абзацы, изображения, ссылки и другие блоки контента. HTML служит фундаментом, на котором строится веб-страница.

CSS (Cascading Style Sheets) используется для оформления веб-страниц. С его помощью задаются стили и визуальные эффекты, такие как цвета, шрифты, отступы, выравнивание и другие параметры дизайна. CSS позволяет создавать привлекательные и удобные интерфейсы, обеспечивая единый стиль для всех страниц сайта.

JavaScript – это язык программирования, который добавляет интерактив­ность и динамичность на веб-страницы. С его помощью можно создавать анима­ции, обрабатывать пользовательские события, управлять содержимым и струк­турой страницы без перезагрузки. JavaScript широко используется для создания сложных веб-приложений, таких как онлайн-игры, социальные сети и другие ин­терактивные платформы.

Помимо основных языков, в веб-программировании применяются различ­ные фреймворки и библиотеки. jQuery – это популярная библиотека JavaScript, которая упрощает работу с DOM, обработку событий и анимацию. Angular, React и Vue.js – это фреймворки для разработки одностраничных приложений (SPA), которые помогают управлять состоянием приложения и создавать компоненты.

На стороне сервера используются такие языки, как PHP, Python, Ruby и Node.js. Эти языки позволяют обрабатывать запросы от клиентов, взаимодей­ствовать с базами данных и выполнять серверные задачи. PHP часто использу­ется для создания динамических веб-страниц и управления контентом, а Node.js позволяет использовать JavaScript для серверной разработки.

Базы данных, такие как MySQL, PostgreSQL и MongoDB, используются для хранения и управления данными веб-приложений. Они обеспечивают надежное хранение данных и быстрый доступ к ним при обработке запросов.

Современные технологии, такие как AJAX (Asynchronous JavaScript and XML) и WebSockets, позволяют создавать веб-приложения с реального времени, обеспечивая мгновенный обмен данными между клиентом и сервером без пере­загрузки страницы.

Веб-программирование продолжает развиваться, и новые инструменты и технологии постоянно появляются на рынке, расширяя возможности разработ­чиков и улучшая качество создаваемых веб-приложений.

# **2.3 Инструменты для вёрстки сайта**

Для верстки сайтов существует множество инструментов, которые помо­гают разработчикам создавать качественные, адаптивные и кроссбраузерные веб-страницы. Эти инструменты могут быть разделены на несколько категорий, каждая из которых выполняет определенные функции.

Редакторы кода являются основными инструментами для верстальщиков. Среди них выделяются такие популярные редакторы, как Visual Studio Code, Sublime Text и Atom. Эти редакторы предлагают множество функций, таких как подсветка синтаксиса, автодополнение кода, интеграция с системами контроля версий и множество плагинов, которые облегчают процесс разработки.

Фреймворки CSS значительно упрощают и ускоряют процесс верстки. Наиболее известными из них являются Bootstrap и Foundation. Эти фреймворки предоставляют готовые компоненты и стили, которые можно использовать для создания адаптивных и стильных интерфейсов. Они включают в себя сеточные системы, типографику, формы, кнопки и другие элементы, которые легко настра­иваются под нужды конкретного проекта.

Препроцессоры CSS помогают упростить и улучшить процесс написания CSS-кода. С их помощью можно использовать переменные, вложенные правила, функции и другие возможности, которые недоступны в чистом CSS. Самыми по­пулярными препроцессорами являются Sass и LESS. Они позволяют писать бо­лее структурированный и удобочитаемый код, который затем компилируется в обычный CSS.

Инструменты для автоматизации задач помогают ускорить процесс разра­ботки и повысить эффективность. Среди них наиболее популярны такие инстру­менты, как Gulp и Webpack. Они позволяют автоматизировать такие задачи, как компиляция препроцессоров, минификация CSS и JavaScript, оптимизация изоб­ражений, запуск локального сервера и перезагрузка страницы при изменениях в коде.

Инспекторы кода и инструменты разработчика встроены в современные браузеры и предоставляют мощные средства для отладки и тестирования веб-страниц. Такие инструменты, как Chrome DevTools и Firefox Developer Tools, позволяют анализировать структуру HTML, просматривать и изменять стили CSS, отлаживать JavaScript, а также тестировать адаптивность и производитель­ность веб-страниц.

Инструменты для тестирования и отладки кроссбраузерной совместимости помогают убедиться, что сайт правильно отображается и функционирует во всех популярных браузерах и на различных устройствах. Сервисы, такие как BrowserStack и CrossBrowserTesting, предоставляют возможность тестировать сайт в различных браузерах и операционных системах, что позволяет выявить и устранить проблемы с отображением и функциональностью.

Системы контроля версий играют важную роль в процессе разработки, поз­воляя отслеживать изменения в коде и управлять версиями проекта. Git является наиболее популярной системой контроля версий, а платформы, такие как GitHub и GitLab, предоставляют дополнительные инструменты для совместной работы над проектами, управления задачами и автоматизации рабочих процессов.

Использование этих инструментов в совокупности позволяет верстальщи­кам создавать качественные, функциональные и визуально привлекательные веб-страницы, обеспечивая при этом эффективный и упорядоченный процесс разработки.

Браузеры – это программы, позволяющие пользователям просматривать веб-страницы, используя для этого протокол HTTP или HTTPS. Они интерпре­тируют коды HTML, CSS и JavaScript, чтобы отобразить содержимое сайта в по­нятной для человека форме.

Существует множество веб-браузеров, но самыми популярными из них яв­ляются Google Chrome, Mozilla Firefox, Safari, Microsoft Edge и Opera. Каждый из них имеет свои уникальные особенности и инструменты для разработчиков, та­кие как консоль разработчика, просмотр элементов страницы и тестирование адаптивного дизайна.

Для верстки сайта в данном случае использовался Google Chrome, по­скольку он обладает широкими возможностями для отладки и оптимизации веб-страниц, а также поддерживает последние стандарты веб-разработки.

Для работы с макетом и выделение необходимых элементов макета для вёрстки обычно используются графические редакторы, например, Photoshop.

# **3 РАЗРАБОТКА ВЕБ-САЙТА**

# **3.1 Физическая и логическая структура сайта**

Физическая структура сайта представляет собой организацию папок и фай­лов на сервере. Это включает в себя:

– Главную директорию, которая содержит все файлы сайта;

– Поддиректории для различных типов контента, таких как изображения, стили (CSS), скрипты (JavaScript) и другие ресурсы;

– Основные файлы, такие как «index.html» или «index.php», которые обычно служат точками входа на сайт.

Логическая структура сайта описывает организацию и связи между стра­ницами и разделами сайта. Это включает:

– Навигационные ссылки, которые помогают пользователям перемещаться по сайту;

– Структуру меню, отражающую иерархию страниц и разделов;

– Взаимосвязи между страницами, такие как ссылки на другие страницы, внутренние и внешние ссылки, и перекрестные ссылки для улучшения юзаби­лити и SEO.

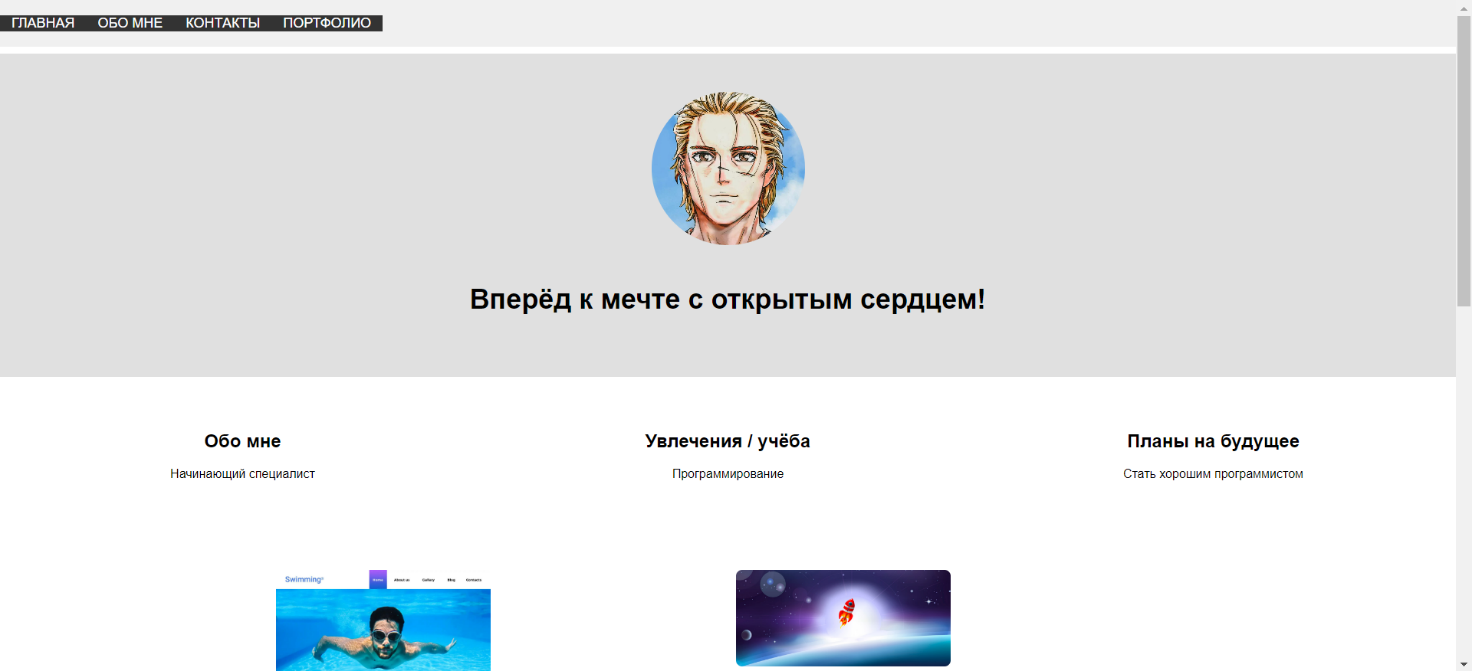
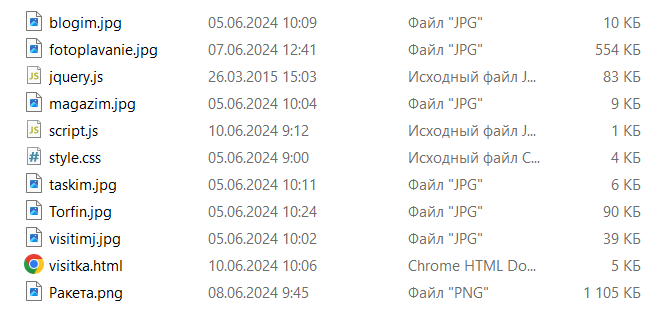
Эти две структуры тесно связаны и вместе обеспечивают функциональ­ность, удобство использования и эффективность работы сайта. На рисунке 3.1.1 представлена логическая структура, на рисунке 3.1.2 представлена физическая структура.

Рисунок 3.1 – Логическая структура сайта

Рисунок 3.2 – Физическая структура сайта

# **3.2 Вёрстка главной страницы сайта**

Вёрстка веб-страниц представляет собой процесс создания HTML-раз­метки и стилей для отображения контента на веб-сайтах. Существуют несколько видов вёрстки:

– Табличная вёрстка: Использует HTML-таблицы для расположения эле­ментов на странице. Это устаревший метод, который сейчас практически не ис­пользуется из-за ограниченной гибкости и сложности в поддержке.

– Блочная вёрстка: Основана на использовании блочных элементов HTML, таких как «<div>», с применением CSS для их позиционирования и стилизации. Этот метод является более гибким и широко используется.

– Вёрстка с использованием Flexbox: Flexbox (Flexible Box Layout) – это современный метод, который обеспечивает простое и эффективное выравнива­ние элементов в контейнере. Flexbox позволяет легко создавать адаптивные ма­кеты, особенно в однорядных или одно-колоночных структурах.

– Вёрстка с использованием CSS Grid: CSS Grid Layout – это мощная си­стема, которая позволяет создавать сложные сетки для дизайна страниц. Grid предоставляет гибкость в расположении элементов в двумерной сетке (по стро­кам и столбцам).

– Адаптивная вёрстка: Предполагает использование медиа-запросов в CSS для создания макетов, которые адаптируются к различным размерам экранов и устройствам. Адаптивная вёрстка обеспечивает оптимальное отображение сайта на любых устройствах, от мобильных телефонов до настольных компьютеров.

– Респонсивная вёрстка: Подход, при котором дизайн страницы динамиче­ски изменяется в зависимости от размера окна браузера. Этот метод включает использование гибких сеток, гибких изображений и медиа-запросов для обеспе­чения удобного просмотра на любых устройствах.

Каждый вид вёрстки имеет свои преимущества и области применения, и выбор подходящего метода зависит от требований конкретного проекта.

В моём проекте при создании использовалась адаптивная блочная вёрстка для просмотра проекта на различных устройствах с различными типами экранов с использованием блоков <div>.

Пример представлен ниже:

<div class="hero\_\_photo">

<img src="Torfin.jpg" alt="Hero Image" class="hero\_\_image">

</div>

<div class="hero\_\_text">

<h2>Вперёд к мечте с открытым сердцем!</h2>

</div>

# **3.3 Использование CSS**

CSS используется создателями веб-страниц для задания цветов, шрифтов, стилей, расположения отдельных блоков и других аспектов представления внеш­него вида веб-страниц.

В моём проекте в файле CSS описаны стили компонентов, в котором описан стиль основного тела страницы.

Пример представлен ниже:

body {

font-family: Arial, sans-serif;

margin: 0;

padding: 0;

box-sizing: border-box;

padding-top: 70px;

}

# **3.4 Интерактивность сайта**

Интерактивность сайта – это способность веб-страницы реагировать на действия пользователя. Это позволяет создавать более динамичные и увлека­тельные взаимодействия, улучшая пользовательский опыт.

В моём проекте представлен код, который при нажатии на текст меняется его шрифт.

Пример представлен ниже:

$(document).ready(function() {

var mainColor = $('#mainColor').val();

var highlightColor = $('#highlightColor').val();

var mainSize = $('#mainSize').val() + 'px';

var highlightSize = $('#highlightSize').val() + 'px';

$('#mainColor, #highlightColor, #mainSize, #highlightSize').on('input', function() {

mainColor = $('#mainColor').val();

highlightColor = $('#highlightColor').val();

mainSize = $('#mainSize').val() + 'px';

highlightSize = $('#highlightSize').val() + 'px';

});

# **3.5 Адаптивность веб-дизайна**

Адаптивность – одно из ключевых требований, предъявляемых к совре­менным сайтам. Ресурс должен одинаково хорошо демонстрироваться на экране компьютера, планшете, смартфоне. Для этого создаётся адаптивный дизайн и ис­пользуется адаптивная вёрстка.

В данном проекте использована система адаптивной вёрстки для одина­ково хорошего просмотра сайта на любых устройствах с любым разрешением экрана, основанная на медиа-запросах.

@media(max-width:767px){

#privacy li a{

font-size: 10px;}

#social-link li a{

font-size: 10px;}

}

# **ЗАКЛЮЧЕНИЕ**

Заключение учебной практики по веб-разработке подводит итоги проде­ланной работы и приобретенных навыков. За время практики я по­лу­чил ценный опыт в области веб-программирования, верстки и веб-конструи­ро­вания, освоили современные технологии и инструменты, необходимые для про­фессиональной деятельности.

Основные результаты учебной практики:

– Сформированы устойчивые навыки работы с основными языками веб-разработки, такими как HTML, CSS и JavaScript;

– Освоены принципы адаптивного дизайна и кроссбраузерной совмести­мости;

– Приобретены практические навыки использования фреймворков и биб­лиотек, таких как Bootstrap и jQuery;

– Развиты умения тестирования, отладки и оптимизации веб-сайтов;

– Совершенствованы навыки работы в команде и эффективного взаимо­действия с коллегами и руководством.

Проведенная практика позволила мне не только закрепить теорети­ческие знания, полученные в ходе обучения, но и применить их на практике, что явля­ется важным шагом на пути к профессиональному становлению. Получен­ные знания и навыки станут прочной основой для дальнейшего развития в обла­сти веб-разработки и помогут успешно справляться с профессиональными зада­чами в будущем.

Для улучшения практики по веб-разработке можно предложить следую­щие идеи:

– Регулярные код-ревью. Проведение регулярных код-ревью с одногрупп­никами позволит не только улучшить качество кода, но и обеспечить обмен зна­ниями и лучшие практики в команде. Это также способствует более глубокому понима­нию проекта среди всех участников команды.

– Применение тестирования. Внедрение автоматизированных тестов для фронтенда и бэкенда, включая юнит-тесты, интеграционные тесты и тесты поль­зовательского интерфейса, поможет снизить количество ошибок и улучшить ста­бильность веб-приложений.

# **СПИСОК ЛИТЕРАТУРЫ**

1. Браун, Э. Изучаем JavaScript: руководство по созданию современных веб-сайтов / Э. Браун. – СПб.: ООО «Альфа-книга», 2019.
2. Дакетт, Дж. JavaScript и jQuery. Интерактивная веб-разработка / Дж. Дакетт. – М.: Изд-во «Э», 2019.
3. Джонсон, M. JavaScript: The New Generation / M. Johnson. – Нью-Йорк: TechBooks, 2019.
4. Макфарланд, Д. JavaScript и jQuery: исчерпывающее руководство / Д. Макфарланд. – Москва: Эксмо, 2020.
5. Макфарланд, Д. Большая книга CSS3 / Д. Макфарланд. – СПб.: Питер, 2019.
6. Морето, С. Bootstrap в примерах / С. Морето. – М.: ДМК Пресс, 2019.
7. Никсон, Р. Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и HTML5 / Р. Никсон. – СПб.: Питер, 2020.
8. Перепелица, Ф.А. Эффективная разработка веб-сайтов. Bootstrap / Ф.А. Перепелица. – СПб: Университет ИТМО, 2019.
9. Смит, S. Modern Web Development with HTML5 and CSS3 / S. Smith. – Лондон: DevPublishers, 2020.
10. Фрэйн, Б. HTML5 и CSS3. Разработка сайтов для любых браузеров и устройств / Б. Фрэйн. – СПб.: Питер, 2019.
11. Хоган, Б.; Уоррен, К.; Уэбер, М.; Годин, А. Книга веб-программиста: секреты профессиональной разработки веб-сайтов / Б. Хоган, К. Уоррен, М. Уэбер, А. Годин. – СПб.: Питер, 2020.

# **ПРИЛОЖЕНИЕ А**

**Фрагмент кода главной страницы**

<header>

<nav>

<ul class="nav-links"></ul>

</nav>

</header>

<main>

<section class="hero" id="hero">

<div class="hero\_\_photo">

<img src="Torfin.jpg" alt="Hero Image" class="hero\_\_image">

</div>

<div class="hero\_\_text">

<h2>Вперёд к мечте с открытым сердцем!</h2>

</div>

</section>

<section class="horizontal-sections about" id="about">

<div class="section">

<h2>Обо мне</h2>

<p>Начинающий специалист</p>

</div>

<div class="section">

<h2>Увлечения / учёба</h2>

<p>Программирование</p>

</div>

<div class="section">

<h2>Планы на будущее</h2>

<p>Стать хорошим программистом</p>

</div>

<div class="contact-info">

<h2>Мои контакты</h2>

</div>

</section>

</main>

# **ПРИЛОЖЕНИЕ Б**

**Фрагмент кода стилевого описания**

body {

font-family: Arial, sans-serif;

margin: 0;

padding: 0;

box-sizing: border-box;

padding-top: 70px;

}

header {

background-color: #f0f0f0;

padding: 20px 0;

display: flex;

justify-content: space-between;

align-items: center;

position: fixed;

width: 100%;

top: 0;

z-index: 1000;

}

.logo {

font-size: 24px;

font-weight: bold;

color: #333;

}

.nav-links {

list-style: none;

padding: 0;

margin: 0;

display: flex;

justify-content: space-around;

background-color: #333;

}

# **ПРИЛОЖЕНИЕ В**

**Фрагмент кода задания на jQuery**

$(document).ready(function() {

const menuItems = [

{ href: '#hero', text: 'ГЛАВНАЯ' },

{ href: '#about', text: 'ОБО МНЕ' },

{ href: '#contacts', text: 'КОНТАКТЫ' }

];

menuItems.forEach(item => {

$('.nav-links').append(`<li><a href="${item.href}">${item.text}</a></li>`);

});

$('.nav-links').append(`

<li class="dropdown">

<a href="#portfolio">ПОРТФОЛИО</a>

<div class="dropdown-content">

<a href="#work1">Работа 1</a>

<a href="#work2">Работа 2</a>

<a href="#work3">Работа 3</a>

<a href="#work4">Работа 4</a>

</div>

</li>

`);

$('.nav-links').css('justify-content', 'flex-end');

});