МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»
РТУ МИРЭА

Институт информационных технологий (ИИТ)
Кафедра инструментального и прикладного программного обеспечения (ИиППО)

ОТЧЁТ ПО КУРСОВОЙ РАБОТЕ
Ознакомительная практика

Отчет представлен к
рассмотрению:
Студент группы ИКБО-16-22 «__» июня 2023 Схрейдер А.К.
(подпись и расшифровка подписи)

Отчет утвержден.
Допущен к защите:

Руководитель практики
от кафедры «__» июня 2023 Матчин В.Т.
(подпись и расшифровка подписи)

Москва 2023 г.

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт информационных технологий (ИТ)
Кафедра инструментального и прикладного программного обеспечения (ИиППО)

ЗАДАНИЕ
на выполнение курсовой работы

по дисциплине: Разработка клиентских частей интернет-ресурсов
по профилю: Разработка программных продуктов и проектирование информационных систем
направления профессиональной подготовки: Программная инженерия (09.03.04)

Студент: Схрейдер Александр Кунрадович
Группа: ИКБО-36-22
Срок представления к защите: 12.12.2023
Руководитель: Василий Тимофеевич Матчин, старший преподаватель

Тема: Клиентская часть интернет-ресурса «150 лет С.В. Рахманинову»

Исходные данные: используемые технологии: HTML5, CSS3, JavaScript, редактор кода Visual Studio Code, версия 1.83.1, наличие: интерактивного поведения веб-страниц, межстраничной навигации, внешнего вида страниц, соответствующего современным стандартам веб-разработки; технологий адаптивной верстки для полноценного отображения контента на различных браузерах и видах устройств. Нормативный документ: инструкция по организации и проведению курсового проектирования СМКО МИРЭА 7.5.1/04.И.05-18.

Перечень вопросов, подлежащих разработке, и обязательного графического материала: 1. Провести анализ предметной области разрабатываемой клиентской части интернет-ресурса. 2. Обосновать выбор технологий разработки клиентской части интернет-ресурса. 3. Создать пять и более веб-страниц интернет-ресурса. 4. Организовать межстраничную навигацию. 5. Реализовать слой клиентской логики веб-страниц с применением технологии JavaScript. 6. Провести оптимизацию веб-страниц и размещаемого контента для браузеров и различных видов устройств. 7. Создать презентацию по выполненной курсовой работе. 8. Оформление курсовой работы должно соответствовать ГОСТ 7.32-2017

Руководителем произведён инструктаж по технике безопасности, противопожарной технике и правилам внутреннего распорядка.

Зав. кафедрой ИиППО: _____/Р. Г. Болбаков/, «18» сентября 2023 г.
Задание на КР выдал: _____/В.Т. Матчин/, «18» сентября 2023 г.
Задание на КР получил: _____/А.К. Схрейдер /, «18» сентября 2023 г.

РЕФЕРАТ

Отчет 21 с., 15 источн., 1 прил., 1 листинг

КЛИЕНТСКАЯ ЧАСТЬ ИНТЕРНЕТ-РЕСУРСА «150 ЛЕТ С.В.РАХМАНИНОВУ»

Объектом исследования являются средства создания веб-приложений на основе HTML, CSS и JavaScript.

Цель работы – Провести анализ предметной области разрабатываемой клиентской части интернет-ресурса. Обосновать выбор технологий разработки клиентской части интернет-ресурса. Создать пять и более веб-страниц интернет-ресурса. Организовать межстраничную навигацию. Реализовать слой клиентской логики веб-страниц с применением технологии JavaScript. Провести оптимизацию веб-страниц и размещаемого контента для браузеров и различных видов устройств.

В процессе работы проводилось изучение особенностей отдельных инструментов и методов создания веб-приложений на основе HTML, CSS и JavaScript, которые могут быть применены для создания клиентской части интернет-ресурса.

В результате работы был разработан программный код, реализующий определенные функции веб-приложения на основе указанных технологий. Эти функции включают в себя отображение информации о С.В. Рахманинове, предоставление возможности просмотра его произведений, а также взаимодействие пользователей с ресурсом.

Область применения результатов: разработка клиентской части веб-приложения на основе HTML, CSS и JavaScript для интернет-ресурса, посвященного 150-летию С.В. Рахманинова.

Оглавление

1.ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ

В настоящем отчете применяются следующие термины с соответствующими определениями.

1. HTML (HyperText Markup Language) - язык разметки, используемый для создания структуры и содержания веб-страниц.
2. CSS (Cascading Style Sheets) - язык стилей, используемый для задания внешнего вида и форматирования веб-страниц.
3. JavaScript - объектно-ориентированный язык программирования, используемый для создания интерактивных элементов и функциональности на веб-страницах.
4. Веб-приложение - программа, которая работает в браузере и обеспечивает пользователю доступ к данным и функциональности через интернет.
5. SPA (Single Page Application) - веб-приложение, которое загружает только одну страницу и динамически обновляет ее содержимое без перезагрузки всей страницы.
6. MVC (Model-View-Controller) - архитектурный шаблон, используемый для разделения приложения на три компонента: модель, представление и контроллер.
7. Framework - набор инструментов, библиотек и шаблонов, который облегчает процесс разработки веб-приложений, предоставляя готовые решения для повторяющихся задач.

2.ВВЕДЕНИЕ

В эпоху цифровых технологий и виртуальной связи, интернет-ресурсы становятся неотъемлемой частью современного общества, играя ключевую роль в распространении информации, культурном обмене и сохранении наследия выдающихся личностей. Одним из ярких примеров такого виртуального пространства является интернет-платформа, посвященная выдающемуся российскому композитору и пианисту Сергею Васильевичу Рахманинову.

Данный отчет фокусируется на клиентской части интернет-ресурса, посвященного 150-летнему юбилею С.В. Рахманинова. Комплексный подход к созданию веб-сайта обеспечивает не только техническую функциональность, но и обогащение пользовательского опыта, позволяя любителям музыки и искусства поглубже погрузиться в мир творчества этого выдающегося композитора.

С учетом значимости истории, духовного наследия и творческого вклада С.В. Рахманинова, интернет-ресурс стремится не только предоставить информацию о его жизни и творчестве, но и создать интерактивное взаимодействие с пользователями. Клиентская часть сайта представляет собой виртуальный портал, предлагающий посетителям уникальные возможности в области обучения, анализа и восприятия музыки.

Основные разделы веб-ресурса включают в себя биографическую информацию о композиторе, аудио материалы, видео материалы и интерактивное музыкальное пианино. Структура сайта обеспечивает доступность контента для широкого круга пользователей, независимо от уровня музыкальной подготовки, предоставляя возможность как профессиональным музыкантам, так и любителям музыки, насладиться творчеством Рахманинова.

3.ОНОВНЫЕ ПОНЯТИЯ И ТЕХНОЛОГИИ

3.1HTML: структура

HTML (Hypertext Markup Language) - это язык, который используется для создания веб-страниц. Его цель - структурировать содержимое страницы, расположить на ней элементы и определить ее внешний вид. В этом ответе мы рассмотрим синтаксис и структуру HTML, чтобы понять, как создаются веб-страницы.

Документ HTML состоит из двух основных разделов: тегов HEAD и BODY. Тег HEAD содержит метаданные, такие как заголовок страницы, автор, ключевые слова и описания. Эта информация используется поисковыми системами для индексации страницы. Тег BODY содержит фактическое содержимое страницы, включая текст, изображения, видео, ссылки и другие элементы.

HTML использует теги для обозначения элементов на странице. Каждый тег заключен между символами < и >, и некоторые теги могут содержать текст или атрибуты между открывающим и закрывающим тегами.

Некоторые из наиболее часто используемых тегов HTML:

• <html>: обозначает начало и конец документа HTML

• <head>: содержит метаданные для страницы

• <title>: определяет заголовок страницы, который отображается в верхней части окна браузера

• <body>: определяет содержимое страницы

• <h1>-<h6>: определяют заголовки разных уровней

• <p>: определяет параграфы текста

• <a>: определяет ссылки

• <img>: определяет изображения

• : определяет ненумерованный список

• : определяет упорядоченный список

Атрибуты используются в тегах для определения дополнительных характеристик элементов на странице. Некоторые общие атрибуты включают:

• id: определяет уникальный идентификатор элемента на странице

• class: определяет класс элемента, который может быть использован для стилизации элементов с помощью CSS

• href: определяет ссылку на другую страницу или документ.

Пример HTML кода:

Листинг 3.1 – Программный код

```
<!DOCTYPE html>
<html>
<head>
  <title>Моя веб-страница</title>
</head>
<body>
  <h1>Добро пожаловать на мою страницу!</h1>
  <p>Это пример текста на странице.</p>
  <img src="haha.jpg" alt="Пример изображения">
  <a href="http://www.something.com">Пример ссылки
на другую страницу</a>
</body>
</html>
```

3.2CSS: стилизация веб-страниц

CSS (Cascading Style Sheets) - это язык стилей для оформления веб-страниц. Он позволяет задавать различные параметры визуального представления веб-страниц, такие как цвет, размер и расположение элементов, шрифты, отступы и многое другое.

Стили CSS могут быть определены как внутри HTML-документа, так и в отдельном файле, который загружается браузером. Разделяя HTML-структуру и ее визуальное представление, CSS позволяет создавать универсальные и гибкие стили, которые могут быть использованы на нескольких страницах сайта.

Стилизация веб-страниц с помощью CSS является важной частью веб-разработки. Научиться использовать CSS позволяет веб-разработчикам создавать красивые, функциональные и доступные веб-сайты.

Существует множество способов использования CSS, включая использование классов и идентификаторов, псевдоэлементов и псевдоклассов,

анимации и переходов, и многого другого. При этом важно помнить о правильной организации кода и о том, что каскадность стилей может иметь важное значение при определении приоритета применения стилей.

Использование CSS также позволяет создавать адаптивные дизайны, которые могут быть оптимизированы для разных устройств и разных размеров экранов. Все это делает CSS незаменимым инструментом для создания современных веб-сайтов и приложений.

В целом, стилизация веб-страниц с помощью CSS - это широкая тема, которая может быть исследована и улучшена на протяжении всей карьеры веб-разработчика.

3.3JavaScript

JavaScript (JS) - это легкий, интерпретируемый или компилируемый язык программирования с функциями первого класса. Хотя он наиболее известен как язык сценариев для веб-страниц, многие не-браузерные среды также используют его, такие как Node.js, Apache CouchDB и Adobe Acrobat. JavaScript является прототипно-ориентированным, мультипарадигмальным, однопоточным, динамическим языком, поддерживающим объектно-ориентированное, императивное и декларативное (например, функциональное программирование) стили.

Динамические возможности JavaScript включают создание объектов во время выполнения, переменные списков параметров, переменные функций, создание динамических скриптов (с помощью eval), интроспекцию объектов (с помощью for...in и утилит объектов) и восстановление исходного кода (функции JavaScript сохраняют свой исходный текст и могут быть получены через toString()).

JavaScript - это язык программирования, который широко используется в веб-разработке и имеет множество возможностей для создания интерактивных пользовательских интерфейсов и динамических веб-страниц. В настоящее время JavaScript является одним из самых популярных языков программирования в мире, и он продолжает развиваться и улучшаться с каждым годом.

4.СОЗДАНИЕ ВЕБ-ПРИЛОЖЕНИЯ НА ОСНОВЕ HTML, CSS И JavaScript

При разработке веб-приложения есть несколько ключевых шагов, которые разработчики должны выполнить. Прежде всего, они должны определить цель приложения и целевую аудиторию. Затем они могут начать работу над планом внешнего вида и функциональности, используя HTML и CSS.

HTML является основой приложения и используется для создания базовой структуры, включая заголовки, абзацы и ссылки. CSS помогает создать более привлекательный внешний вид, добавляя стили, такие как шрифты, цвета и макет. Когда дизайн готов, разработчики могут усовершенствовать приложение, добавляя JavaScript для создания интерактивных и динамических функций.

С помощью JavaScript можно создавать формы, кнопки и другие интерактивные элементы, выполнять расчеты, проверять данные, введенные пользователем, и добавлять анимацию для улучшения пользовательского опыта. Однако необходимо уделить внимание не только эстетическому оформлению, но и доступности приложения для всех пользователей. Это включает тестирование на разных браузерах, обеспечение дружественности к мобильным устройствам и удобства использования для людей с ограниченными возможностями. Не учитывать эти аспекты может негативно сказаться на успехе приложения.

5.ИСПОЛЬЗОВАНИЕ БИБЛИОТЕК И ФРЕЙМВОРКОВ ДЛЯ УПРОЩЕНИЕ РАЗРАБОТКИ

Когда речь идет о разработке программного обеспечения, существует множество инструментов, которые могут сделать процесс более простым и эффективным. Один из таких инструментов - это использование библиотек и фреймворков. Это заранее написанные куски кода, которые могут использоваться для выполнения общих задач, таких как обработка аутентификации пользователя или подключение к базе данных. Используя эти готовые решения, разработчики могут сэкономить время и сосредоточиться на уникальных аспектах своего проекта.

Библиотеки обычно имеют меньший объем и фокусируются на конкретной функциональности, например, генерации PDF или разборе JSON. Фреймворки, с

другой стороны, более крупные и всесторонние. Они предоставляют структуру и набор соглашений для создания целого приложения, включая обработку маршрутизации, взаимодействие с базой данных и пользовательский интерфейс.

Использование как библиотек, так и фреймворков может значительно ускорить время разработки и уменьшить объем кода, который необходимо написать. Однако важно использовать их с умом. Использование слишком большого количества библиотек или фреймворков может увеличить размер приложения и усложнить его обслуживание. Кроме того, слишком сильная зависимость от готовых решений может ограничить творческие способности разработчика и его способность к инновациям.

В заключение, использование библиотек и фреймворков может значительно упростить процесс разработки программного обеспечения, но важно найти баланс между использованием существующих решений и созданием собственного кода.

6.КОД САЙТА

6.1Первая страница

Начальная страница веб-сайта играет фундаментальную роль в привлечении внимания посетителей и установлении первого визуального контакта с предлагаемым контентом. Это виртуальное приветствие предоставляет посетителям первое впечатление о сайте, его тематике и стиле, поэтому она занимает особое место в пользовательском опыте.

На первой странице сайта "150 лет С.В. Рахманинову" гостей встречает уникальное сочетание эстетики, информационной наполненности и удобства взаимодействия. Цель первой страницы - погрузить посетителя в атмосферу творчества композитора, предоставив краткую, но привлекательную информацию о его жизни и значимости.

Ключевые элементы интерфейса на первой странице спроектированы с учетом легкости навигации, позволяя посетителям быстро получить представление о том, что предлагает сайт. Это включает в себя ярко выделенные кнопки перехода

к основным разделам и небольшие анимации элементов, создающие динамичный и запоминающийся пользовательский опыт.

Значительное внимание уделяется также текстовой информации на первой странице. Краткие и содержательные аннотации о цели сайта, его уникальных возможностях и предлагаемых ресурсах помогают посетителям быстро понять, почему стоит провести время именно здесь.

Для более детального визуального представления о первой странице сайта, предлагаю ознакомиться с рисунком 1.1 и 1.2.

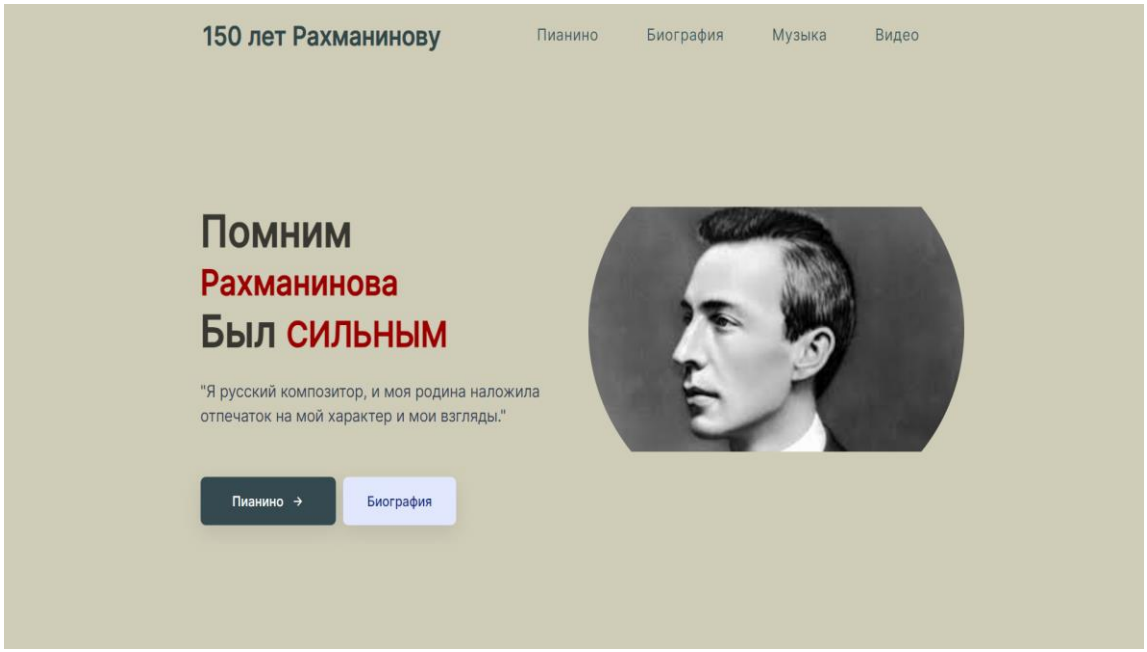

Рисунок 1.1 – Первая часть первой страницы сайта [разработано автором]

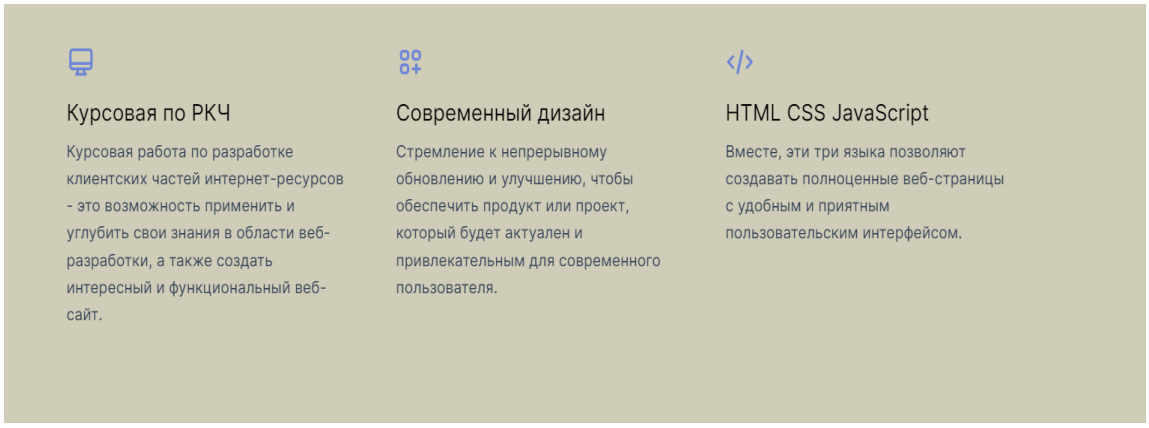

Рисунок 1.2 – Вторая часть первой страницы сайта [разработано автором]

Давайте погрузимся в мир кода и рассмотрим веб-сайт изнутри, где каждая строка кода — это мозаика, создающая визуальное волшебство первой страницы. Этот аспект предоставляет уникальную перспективу, позволяя увидеть, как техническая гармония воплощает идеи и эстетику.

Приведенный код листинга 5.1 представляет собой часть HTML-структуры страницы, ответственной за верхнюю часть сайта – его заголовок или, как в данном случае, шапку. Давайте рассмотрим основные элементы этого листинга.

Листинг 5.1 – Программный код заголовка первой страницы

```
    <header class="container header">
        <nav class="nav">
            <div class="logo">
                <h2>150 лет Рахманинову</h2>
            </div>

            <div class="nav_menu" id="nav_menu">
                <button              class="close_btn"
id="close_btn">
                    <i class="ri-close-fill"></i>
                </button>

                <ul class="nav_menu_list">
                    <li class="nav_menu_item">
                        <a    href="homepage/home.html"
class="nav_menu_link">пианино</a>
                    </li>
                    <li class="nav_menu_item">
                        <a
href="timeline/timeline.html"
class="nav_menu_link">биография</a>
                    </li>
                    <li class="nav_menu_item">
                        <a
href="musicpage/musicpage.html"
class="nav_menu_link">музыка</a>
                    </li>
                    <li class="nav_menu_item">
                        <a  href="videopage/video.html"
class="nav_menu_link">видео</a>
                    </li>
                </ul>
            </div>
```

```
            <button class="toggle_btn" id="toggle_btn">
                <i class="ri-menu-line"></i>
            </button>
        </nav>
    </header>
```

Разбор кода:

1. <header class="container header">: Этот тег <header> обозначает начало блока шапки страницы. Класс "container" вероятно указывает на структурный контейнер, а "header" означает, что это часть шапки.
2. <nav class="nav">: Тег <nav> указывает на начало блока навигации. Класс "nav" вероятно связан с стилями этого блока.
3. : Этот блок содержит логотип сайта, представленный текстом "150 лет Рахманинову".
4. : Здесь находится блок, содержащий кнопку закрытия (close_btn) и список навигационных элементов (nav_menu_list). ID "nav_menu" может быть использован в JavaScript для управления отображением этого меню.
5. <button class="toggle_btn" id="toggle_btn">: Это кнопка-переключатель, предназначенная для открытия и закрытия меню. Она содержит иконку в виде трех горизонтальных линий, что обычно ассоциируется с иконой бургера.

Все вместе этот код формирует структуру и визуальный интерфейс верхней части сайта, предоставляя навигационные элементы и информацию о сайте.

Приведенный код листинга 5.2 представляет собой часть CSS страницы, ответственной за верхнюю часть сайта – его заголовок или, как в данном случае, шапку. Давайте рассмотрим основные элементы этого листинга.

Листинг 5.2 – Программный код

```
    @import
url("https://fonts.googleapis.com/css2?family=Inter:wght@
400;500&display=swap");

    * {
      margin: 0;
      padding: 0;
```

```
    box-sizing: border-box;
    font-family: "Inter", sans-serif;
  }
  *,
  *::before,
  *::after {
    box-sizing: border-box;
  }


  :root {
    --fixed-header-height: 4.5rem;
  }


  body {
    width: 100%;
    height: 100vh;
    overflow-x: hidden;
    background-color: #CFCCB7;
  }
  ul li {
    list-style-type: none;
  }
  a {
    text-decoration: none;
  }
  button {
    background-color: transparent;
    border: none;
    outline: none;
    cursor: pointer;
  }

  .container {
    width: 100%;
  }
  @media screen and (min-width: 1040px) {
    .container {
      width: 1040px;
      margin: 0 auto;
    }
  }
  /* ==== HEADER ==== */
  .header {
```

```
    height: var(--fixed-header-height);
    padding: 0 1.7rem;
  }


  .nav {
    width: 100%;
    height: 100%;
    display: flex;
    align-items: center;
    justify-content: space-between;
  }

  .logo h2 {
    font-size: 30px;
    color: #31484b;
  }

  .nav_menu_list {
    display: flex;
    align-items: center;
  }
  .nav_menu_list .nav_menu_item {
    margin: 0 2rem;
  }
  .nav_menu_item .nav_menu_link {
    font-size: 18.5px;
    line-height: 27px;
    color: rgb(56, 88, 94);
    text-transform: capitalize;
    letter-spacing: 0.5px;
  }
  .nav_menu_link:hover {
    color: #9b0000;
  }
  .toggle_btn {
    font-size: 20px;
    font-weight: 600;
    color: var(--lg-heading);
    z-index: 4;
  }
  .nav_menu,
  .close_btn {
    display: none;
  }
```

```
    .show {
      right: 3% !important;
    }
    @media screen and (min-width: 768px) {
      .toggle_btn {
        display: none;
      }
      .nav_menu {
        display: block;
      }
    }

    @media screen and (max-width: 768px) {
      .logo h2 {
        font-size: 23px;
      }
      .nav_menu {
        position: fixed;
        width: 93%;
        height: 100%;
        display: block;
        top: 2.5%;
        right: -100%;
        background-color: #fff;
        padding: 3rem;
        border-radius: 10px;
        box-shadow: 0 0.5rem 1.5rem rgba(22, 28, 45, 0.1);
        z-index: 50;
        transition: 0.4s;
      }
      .nav_menu_list {
        flex-direction: column;
        align-items: flex-start;
        margin-top: 4rem;
      }
      .nav_menu_list .nav_menu_item {
        margin: 1rem 0;
      }
      .nav_menu_item .nav_menu_link {
        font-size: 18px;
      }
      .close_btn {
        display: block;
        position: absolute;
        right: 10%;
```

```
        font-size: 25px;
        color: #50689e;
      }
      .close_btn:hover {
        color: #000;
      }
      .wrapper {
        padding: 0 0.7rem;
      }
      .grid-item-1 {
        padding-left: 0rem;
      }
      .main-heading {
        font-size: 35px;
      }
      .view_more_btn {
        width: 140px;
        height: 55px;
        font-size: 13.5px;
        margin-right: 1rem;
      }
      .grid-cols-3 {
        grid-template-columns:            repeat(auto-fit,
minmax(100%, 1fr));
      }
      .featured_info p {
        line-height: 23px;
        font-size: 14px;
      }
    }

    @media screen and (max-width: 991px) {
      .wrapper {
        padding-top: 3rem;
      }
      .grid-cols-2 {
        grid-template-columns:            repeat(auto-fit,
minmax(100%, 1fr));
      }
      .grid-item-1 {
        order: 2;
        display: flex;
        flex-direction: column;
        align-items: center;
        justify-content: center;
```

```
      padding-top: 0;
    }
    .main-heading {
      font-size: 32px;
      text-align: center;
      line-height: 40px;
    }
    .info-text {
      font-size: 16px;
      text-align: center;
      padding: 0.7rem;
    }
    .btn_wrapper {
      width: 100%;
      display: flex;
      align-items: center;
      justify-content: center;
    }
    .grid-item-2 {
      order: 1;
      display: flex;
      flex-direction: column;
      align-items: center;
      justify-content: center;
    }
    .team_img_wrapper {
      width: 350px;
      height: 350px;
    }
    .featured_info span {
      font-size: 19px;
    }
  }
```

Основные аспекты CSS кода:

1. Импорт шрифта Inter с использованием Google Fonts.
2. Сброс стилей и установка базовых свойств для всех элементов.
3. Определение переменной --fixed-header-height для высоты фиксированной шапки.
4. Стилизация основного контейнера, адаптивная ширина при разрешении 1040px и выше.
5. Стилизация шапки (.header) с высотой, отступами и фоновым цветом.

6. Стилизация навигационного блока (.nav) и логотипа.
7. Установка стилей для пунктов меню и ссылок, включая эффект при наведении.
8. Стилизация кнопки-переключателя (.toggle_btn) с учетом размера, веса и цвета.
9. Отображение и скрытие элементов (.nav_menu, .close_btn) с использованием класса .show.

Медиа-запросы для адаптивности:

Код включает медиа-запросы, которые оптимизируют стили для различных устройств и разрешений экрана.

Для широких экранов (минимум 1040px) предусмотрена фиксированная ширина контейнера.

Для средних экранов (от 768px до 1040px) сохраняется адаптивная ширина контейнера.

Навигационное меню (Hamburger menu):

При ширине экрана менее 768px применяются стили для мобильного отображения.

Кнопка переключения меню (toggle_btn) видна только на экранах с шириной более 768px.

На мобильных устройствах меню (nav_menu) отображается при нажатии на кнопку и скрывается при повторном нажатии.

Для стилизации мобильного меню используются анимации (transition), задающие плавное открытие и закрытие.

Определение стилей для кнопок:

Стили для кнопок включают нулевые отступы и рамки (button), а также определены стили для кнопок с классами .btn, .view_more_btn, и .documentation_btn. При наведении на кнопки реализованы эффекты изменения цвета и тени.

Адаптация контента:

На мобильных устройствах (ширина экрана до 768px) текст и изображения адаптированы для лучшей читаемости и визуального восприятия.

Для этого использованы медиа-запросы, определяющие стили для соответствующих разрешений экрана.

Этот код демонстрирует гибкость и адаптивность дизайна, обеспечивая удобное взаимодействие с сайтом на различных устройствах, можно убедиться в этом посмотрев рисунок 1.3.

Рисунок 1.3 – Первая страница сайта представленная на смартфоне [разработано автором]

Приведенный код листинга 5.3 представляет собой часть JavaScript страницы, ответственной за верхнюю часть сайта – его заголовок или, как в данном случае, шапку. Давайте рассмотрим основные элементы этого листинга.

Листинг 5.3 – Программный код

```
    document.addEventListener("DOMContentLoaded",
function () {
      const options = {
        strings:  ['ВЕЛИКИМ',  'КОМПОЗИТОР',  'МУДРЫМ',
'СИЛЬНЫМ' ],
        typeSpeed: 150,
        backSpeed: 50,
        backDelay: 1000,
        loop: true
      };

      const              multiTextElement              =
document.querySelector('.multi-text');
      let currentTextIndex = 0;
      let currentText = '';
      let isDeleting = false;

      function type() {
        const                fullText                =
options.strings[currentTextIndex];
        if (isDeleting) {
          currentText      =      fullText.substring(0,
currentText.length - 1);
        } else {
          currentText      =      fullText.substring(0,
currentText.length + 1);
        }

        multiTextElement.textContent = currentText;

        let typeSpeed = options.typeSpeed;
        if (isDeleting) {
          typeSpeed /= 2;
        }

        if (!isDeleting && currentText === fullText) {
          typeSpeed = options.backDelay;
          isDeleting = true;
        } else if (isDeleting && currentText === '') {
          isDeleting = false;
          currentTextIndex = (currentTextIndex + 1) %
options.strings.length;
        }
```

```
        setTimeout(type, typeSpeed);
      }

      type();
    });
```

Этот JavaScript-код отвечает за анимированный текстовый эффект, который позволяет создать динамичное изменение текста на веб-странице. Вот пошаговое объяснение:

Событие загрузки контента:

document.addEventListener("DOMContentLoaded", function () {...}: Этот код запускается, когда DOM (Document Object Model) полностью загружен.

Настройка параметров анимации:

const options = {...}: Объект options содержит параметры для настройки анимации.

strings: Массив строк текста, который будет анимироваться.

typeSpeed: Скорость появления символов (набора текста).

backSpeed: Скорость удаления символов (стирания текста).

backDelay: Задержка перед стиранием текста после завершения печати.

loop: Флаг, указывающий, следует ли бесконечно повторять анимацию.

Выбор элемента для анимации:

const multiTextElement = document.querySelector('.multi-text');: Получение элемента с классом .multi-text, куда будет вставляться анимированный текст.

Инициализация переменных:

let currentTextIndex = 0;: Индекс текущей строки текста.

let currentText = '';: Текущий текст, который будет отображаться.

let isDeleting = false;: Флаг, указывающий, удаляется ли текст.

Функция для анимации текста:

function type() {...}: Эта функция выполняет основную логику анимации.

Получает текущий текст из массива options.strings и обновляет multiTextElement с текущим текстом.

Управляет скоростью появления и удаления символов.

Определяет, когда начинать стирание текста и когда закончить его.

Вызов функции для начала анимации:

type();: Запуск функции type(), чтобы начать анимацию текста.

Этот код создает эффект многоточечного текста, который появляется, стирается и затем заменяется следующей строкой текста в бесконечном цикле.

6.2Вторая страница

Вторая страница: "Музыкальная Галерея"

На второй странице интернет-ресурса, посвященного великому композитору С.В. Рахманинову, открывается удивительная "Музыкальная Галерея". Этот раздел не только предоставляет уникальную возможность виртуально испытать звучание пианино, но и обогащен цитатами самого Рахманинова, призванными вдохновить посетителей этой виртуальной палитры музыки.

Интерактивное Пианино:

Посетители могут взять виртуальное пианино под свое управление, играя любимые мелодии композитора. Эта функциональность предоставляет уникальный опыт, позволяя каждому почувствовать себя частью музыкального творчества Рахманинова. Откройте для себя виртуозность его произведений, воспроизводя их собственными руками.

Цитаты С.В. Рахманинова:

Каждый аккорд и нота на странице сопровождаются цитатами великого композитора. Слова Рахманинова приглашают посетителей окунуться в его внутренний мир, вдохновляясь его мыслями о музыке, творчестве и жизни. Эти цитаты становятся мостом между произведениями и душой композитора.

Футер:

На странице присутствует стильный футер, который является не только элементом дизайна, но и практичным навигационным инструментом

Вторая страница "Музыкальной Галереи" не только раскрывает творческий мир композитора, но и создает интерактивную среду, погружая посетителей в увлекательный опыт виртуального исследования музыки.

Для более детального визуального представления о второй странице сайта, предлагаю ознакомиться с рисунком 2.1, 2.2, 2.3 и 2.4.

Рисунок 2.1 – Первая часть второй страницы сайта (интерактивное пианино) [разработано автором]

Рисунок 2.2 – Вторая часть второй страницы сайта [разработано автором]

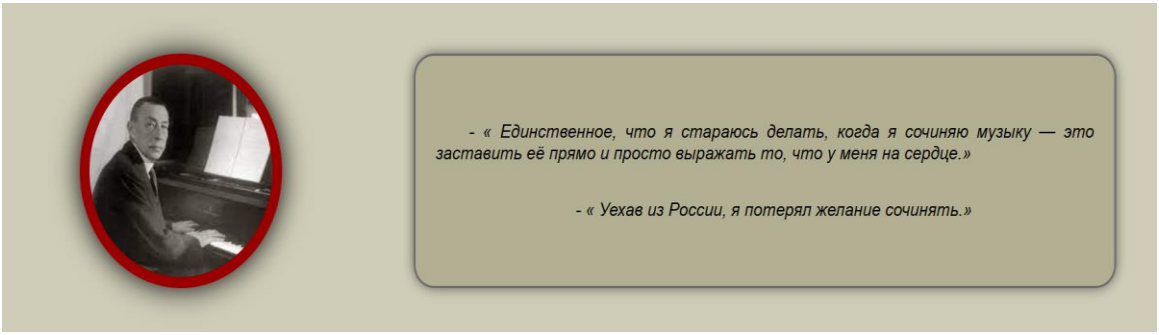

Рисунок 2.3 – Третья часть второй страницы сайта [разработано автором]

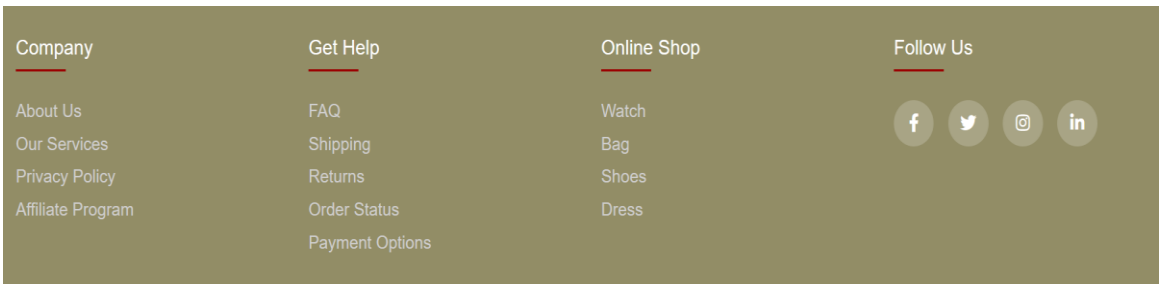

Рисунок 2.4 – Первая часть второй страницы сайта
[разработано автором]

Дополнительно, была создана адаптивная версия для мобильных устройств. Для более детального визуального представления ознакомиться с рисунком 2.5 и 2.6.

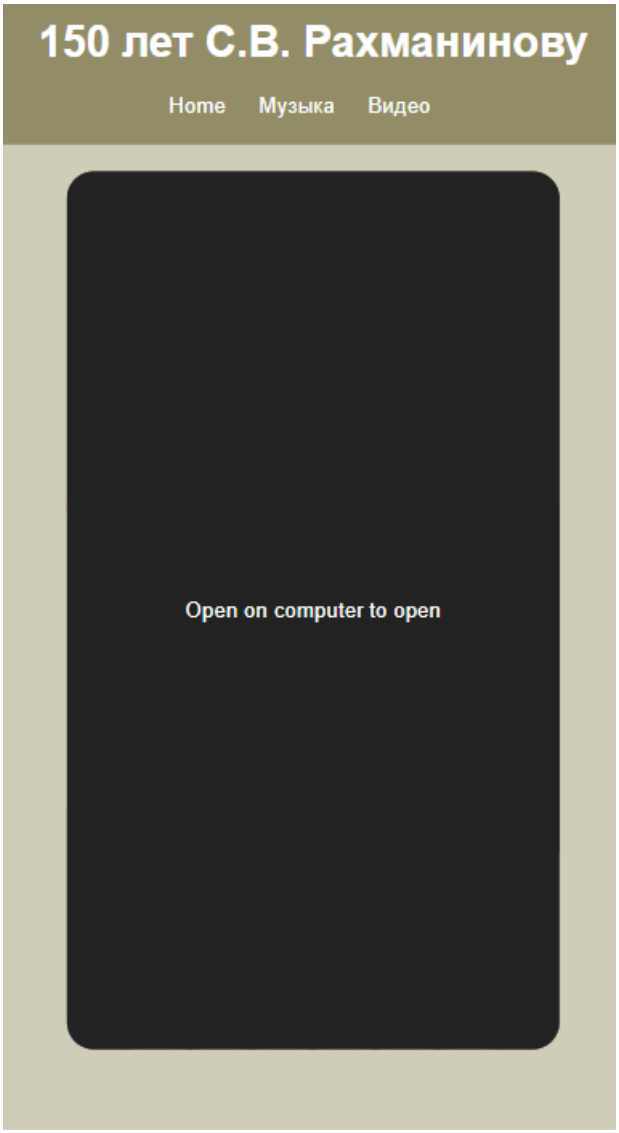

Рисунок 2.4 – адаптивная версия для мобильных устройств
[разработано автором]

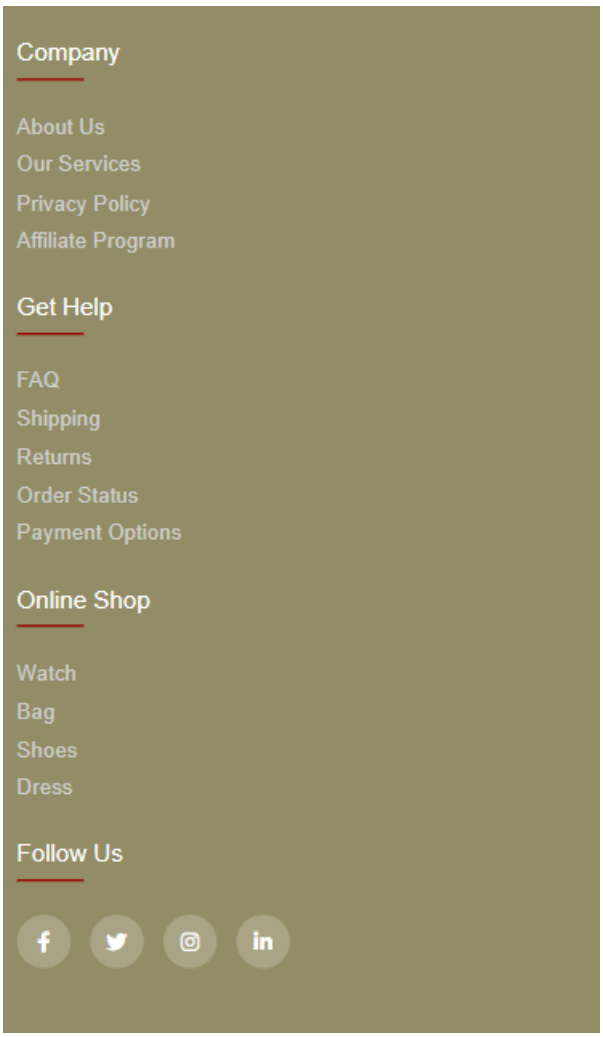

Рисунок 2.4 – адаптивная версия для мобильных устройств
[разработано автором]

6.3Третья страница

Третья страница: "Музыкальный Плеер"

На третьей странице виртуального пространства, посвященного творчеству С.В. Рахманинова, предстает перед посетителями "Музыкальный Плеер". Этот раздел призван углубить музыкальное восприятие и предоставить возможность прослушивания непревзойденных произведений великого композитора.

Музыкальный Плеер:

Посетители могут наслаждаться прослушиванием музыки С.В. Рахманинова через встроенный музыкальный плеер. Этот функционал позволяет выбирать из богатого каталога произведений, создавая персональный плейлист и настраивая воспроизведение по собственному вкусу.

Эксклюзивный Каталог:

"Музыкальный Плеер" включает в себя эксклюзивный каталог музыкальных композиций Рахманинова, предоставляя уникальную возможность погружения в атмосферу его музыкального наследия. Каждое произведение снабжено информацией о его истории, создавая более глубокое понимание контекста.

Интерактивное Управление:

Уникальная возможность управления воспроизведением - от паузы до перемотки - делает взаимодействие с музыкой Рахманинова еще более вдохновляющим. Играйте любимые композиции, создавайте собственные музыкальные путеводители и наслаждайтесь музыкальным наследием великого композитора в любое время и в любом месте.

Третья страница "Музыкального Плеера" призвана погрузить посетителей в уникальный мир музыки Рахманинова, предоставляя удобные и многофункциональные инструменты для наслаждения и исследования его творчества.

Для более детального визуального представления о третьей странице сайта, предлагаю ознакомиться с рисунком 3.1, 3.2.

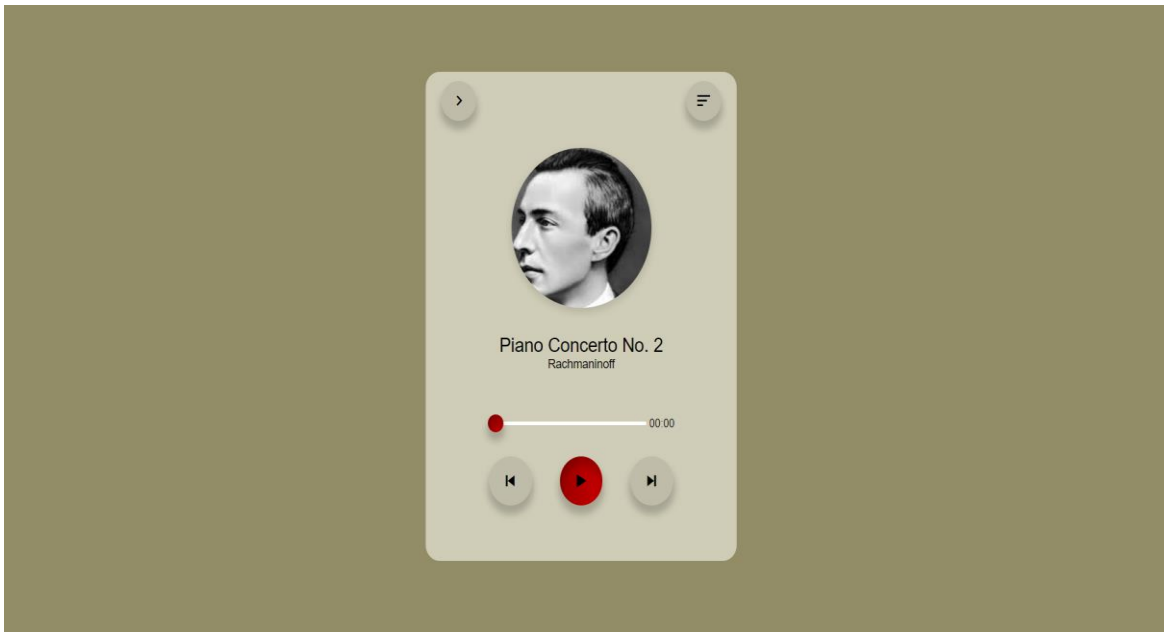

Рисунок 3.1 – Третья страница сайта (плеер)
[разработано автором]

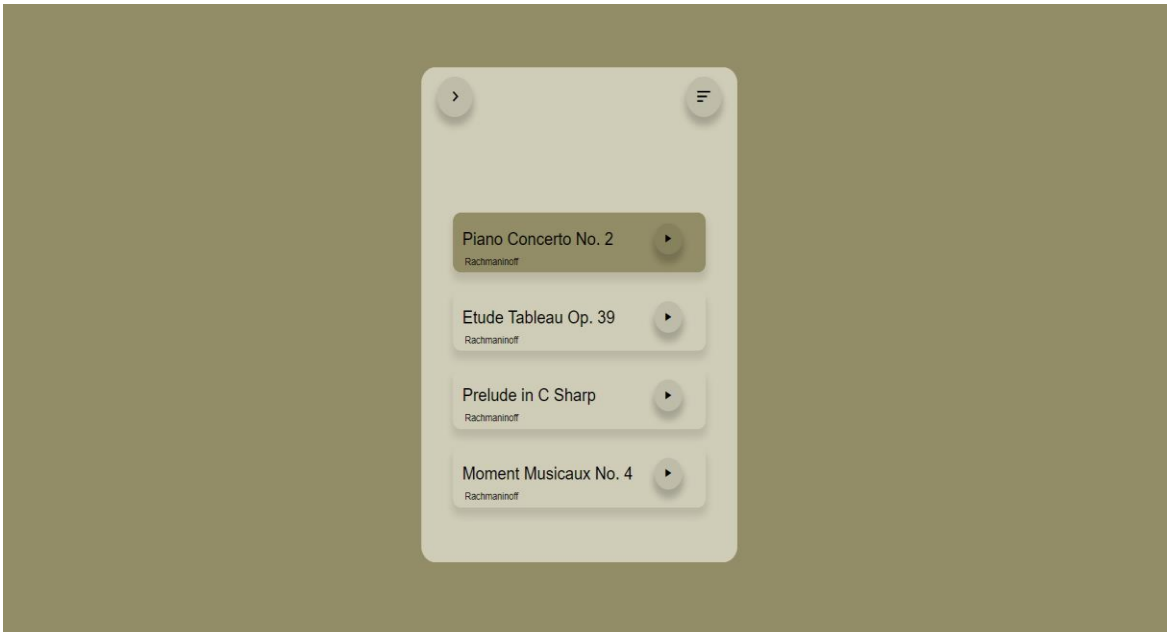

Рисунок 3.2 – Третья страница сайта (выбор пъес)
[разработано автором]

Приведенный код листинга 6.4 представляет собой часть JavaScript страницы, ответственной за верхнюю часть сайта – его заголовок или, как в данном случае, шапку. Давайте рассмотрим основные элементы этого листинга.

Листинг 6.4 – Программный код

```
    function loadTrack(track_index){
        clearInterval(updateTimer);
        reset();

        curr_track.src = music_list[track_index].music;
        curr_track.load();

        document.querySelector(".img_artist").src     =
music_list[track_index].img;
        console.log("done");
        //track_art.style.backgroundImage  =  "url("  +
music_list[track_index].img + ")";
        song_title.innerHTML                          =
music_list[track_index].name;
        //track_name.textContent                      =
music_list[track_index].name;
        artist_name.innerHTML                         =
music_list[track_index].artist;
        //track_artist.textContent                    =
music_list[track_index].artist;
```

```
        //now_playing.textContent = "Playing music " +
(track_index + 1) + " of " + music_list.length;

        var children = playlist.children;
        Array.from(children).forEach(child           =>
child.classList.remove('playing'));

        if(track_index === 0){
            first.classList.add('playing');
        } else if(track_index === 1){
            second.classList.add('playing');
        } else if(track_index === 2){
            third.classList.add('playing');
        } else if(track_index === 3){
            fourth.classList.add('playing');
        }

        updateTimer = setInterval(setUpdate, 500);

        curr_track.addEventListener('ended',
nextTrack);
    }
```

Код в листинге 6.4 представляет функцию loadTrack(track_index), ответственную за загрузку и воспроизведение нового музыкального трека при смене индекса трека. Вот пошаговое объяснение каждой части кода:

Остановка предыдущего трека и сброс таймера обновления:
clearInterval(updateTimer);: Этот код останавливает таймер обновления, если он был предварительно запущен, чтобы избежать конфликтов при загрузке нового трека.
reset();: Функция reset() может выполнять дополнительные действия для сброса состояния при смене трека.

Загрузка нового трека:
curr_track.src = music_list[track_index].music;: Устанавливает источник (src) текущего трека из массива music_list по указанному индексу.
curr_track.load();: Загружает трек для проигрывания.

Обновление информации о треке и изображении исполнителя:
document.querySelector(".img_artist").src = music_list[track_index].img;: Устанавливает источник изображения исполнителя для отображения на веб-странице.

song_title.innerHTML = music_list[track_index].name;: Обновляет заголовок трека.
artist_name.innerHTML = music_list[track_index].artist;: Обновляет имя исполнителя.

Выделение текущего трека в плейлисте:
var children = playlist.children;: Получает все дочерние элементы плейлиста.
Array.from(children).forEach(child => child.classList.remove('playing'));: Удаляет класс 'playing' у всех элементов плейлиста.
Блок условий добавляет класс 'playing' для текущего трека в плейлисте, чтобы выделить его визуально.

Запуск таймера обновления и добавление обработчика события 'ended':
updateTimer = setInterval(setUpdate, 500);: Устанавливает таймер обновления, вызывая функцию setUpdate каждые 500 миллисекунд.
curr_track.addEventListener('ended', nextTrack);: Устанавливает обработчик события 'ended', который вызывает функцию nextTrack при завершении воспроизведения текущего трека.

Функция loadTrack предоставляет механизм для динамического обновления веб-страницы с новой информацией о треке и его визуальным представлением при смене индекса трека.

6.4Четвёртая страница

Четвёртая страница: "Видео Галерея"

На четвёртой странице интернет-ресурса, посвященного С.В. Рахманинову, предстаёт перед посетителями увлекательная "Видео Галерея". Этот раздел предлагает уникальную возможность окунуться в визуальное восприятие музыкального наследия композитора через видео материалы.
Видео Плеер:

Посетители могут наслаждаться увлекательными видео материалами, посвященными творчеству С.В. Рахманинова, с помощью встроенного видео плеера. Этот функционал предоставляет возможность просмотра интересных моментов из жизни и творчества композитора.

Богатство Визуального Материала:

"Видео Галерея" включает в себя разнообразные видео ролики, посвящённые пьесам Рахманинова. Посетители получают уникальную возможность погрузиться в визуальный мир великого композитора.

Интерактивное Управление:

Плеер обеспечивает удобное управление видео воспроизведением, позволяя посетителям наслаждаться каждым моментом материалов. Возможность управления громкостью, перемоткой и переключением видео создает комфортное просмотренное впечатление.

Четвёртая страница "Видео Галереи" является дополнительным источником визуального погружения в творчество композитора. Она приглашает посетителей на увлекательное путешествие через зрелищные моменты, раскрывающие многогранный облик и вклад С.В. Рахманинова в музыкальное искусство.

Для более детального визуального представления о четвёртой странице сайта, предлагаю ознакомиться с рисунком 4.1.

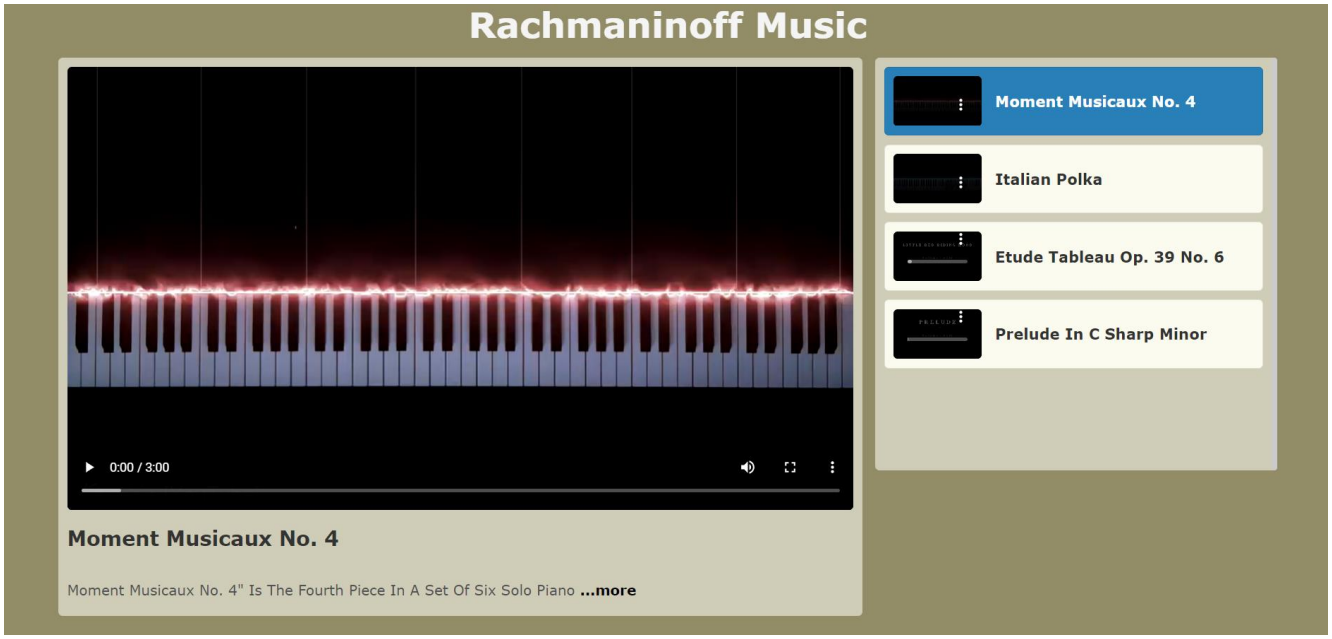

Рисунок 4.1 – Четвёртая страница сайта (видеоплеер)
[разработано автором]

Дополнительно, была создана адаптивная версия для мобильных устройств. Для более детального визуального представления ознакомиться с рисунком 4.2.

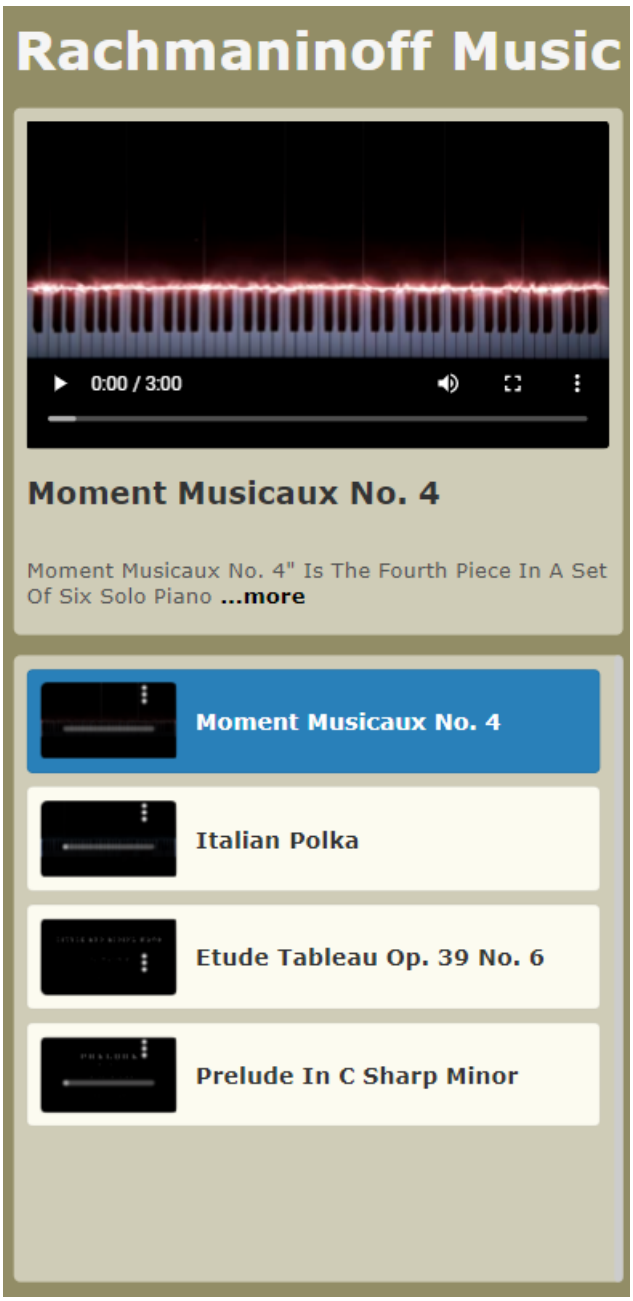

Рисунок 4.2 – адаптивная версия для мобильных устройств [разработано автором]

Приведенный код листинга 6.5 представляет собой часть JavaScript страницы, ответственной за верхнюю часть сайта – его заголовок или, как в данном случае, шапку. Давайте рассмотрим основные элементы этого листинга.

Листинг 6.5 – Программный код

```
    function truncateText(element, maxLength) {
        let originalText = element.textContent;
        let truncatedText = originalText.slice(0, maxLength) + " <a href='#' class='expansion' data-action='more'>...more</a>";
        element.innerHTML = truncatedText;
```

```
        element.addEventListener('click', function (e)
{
            e.preventDefault();

            if
(e.target.classList.contains('expansion')) {
                if (e.target.dataset.action === 'more')
{
                    element.innerHTML = originalText +
" <a href='#' class='expansion' data-action='less'>show
less</a>";
                } else {
                    element.innerHTML = truncatedText;
                }
            }
        });
    }

    truncateText(description, 70);
```

Код в листинге 6.5 представляет собой функцию truncateText, которая используется для обрезания текста внутри HTML-элемента и добавления функционала "показать больше" / "показать меньше". Вот пошаговое объяснение каждой части кода:

Сохранение оригинального текста:

let originalText = element.textContent;: Сохраняет оригинальный текст элемента в переменной originalText.

Обрезка текста и добавление ссылки "показать больше":

let truncatedText = originalText.slice(0, maxLength) + " ...more";: Создает новую строку, в которой оригинальный текст обрезается до указанной длины (maxLength), а после добавляется ссылка "показать больше" с классом expansion и атрибутом data-action='more'.

Замена текста в элементе:

element.innerHTML = truncatedText;: Заменяет содержимое HTML-элемента (element) на обрезанный текст с добавленной ссылкой "показать больше".

Добавление обработчика событий для ссылки:

element.addEventListener('click', function (e) {...});: Устанавливает обработчик события клика для элемента.

e.preventDefault();: Предотвращает стандартное действие ссылки (переход по ссылке).

if (e.target.classList.contains('expansion')) {...}: Проверяет, был ли клик на элементе с классом expansion.

e.target.dataset.action: Получает значение атрибута data-action кликнутого элемента.

Изменение содержимого элемента при клике:

Если кликнутый элемент имеет класс expansion и значение атрибута data-action равно 'more', то текст элемента заменяется на полный текст с ссылкой "показать меньше". Если значение атрибута data-action равно 'less', то текст элемента возвращается к обрезанной версии с ссылкой "показать больше".

Функция truncateText обеспечивает удобный способ управления отображением текста, позволяя пользователю динамически раскрывать или скрывать контент в зависимости от его предпочтений.

6.5Пятая страница

Пятая страница: "Биография С.В. Рахманинова"

На пятой странице интернет-ресурса, посвященного великому композитору С.В. Рахманинову, посетителям предоставляется уникальная возможность погрузиться в богатую и увлекательную биографию этого выдающегося музыкального гения.

Жизненный Путь:

Страница "Биография" представляет полный и обстоятельный обзор жизни и творчества С.В. Рахманинова, начиная с ранних лет его детства и обучения в Московской консерватории до влиятельных моментов в его международной карьере. Посетители могут открывать для себя ключевые события, формирующие личность композитора, а также влияние его семьи и времени, в которое он жил.

Страница "Биография С.В. Рахманинова" представляет собой ценный ресурс для всех, кто стремится лучше понять личность и вклад в историю мировой музыки этого великого композитора.

Для более детального визуального представления о пятой странице сайта, предлагаю ознакомиться с рисунком 5.1.

Рисунок 5.1 – Пятая страница сайта (видеоплеер)
[разработано автором]

Дополнительно, была создана адаптивная версия для мобильных устройств. Для более детального визуального представления ознакомиться с рисунком 5.2.

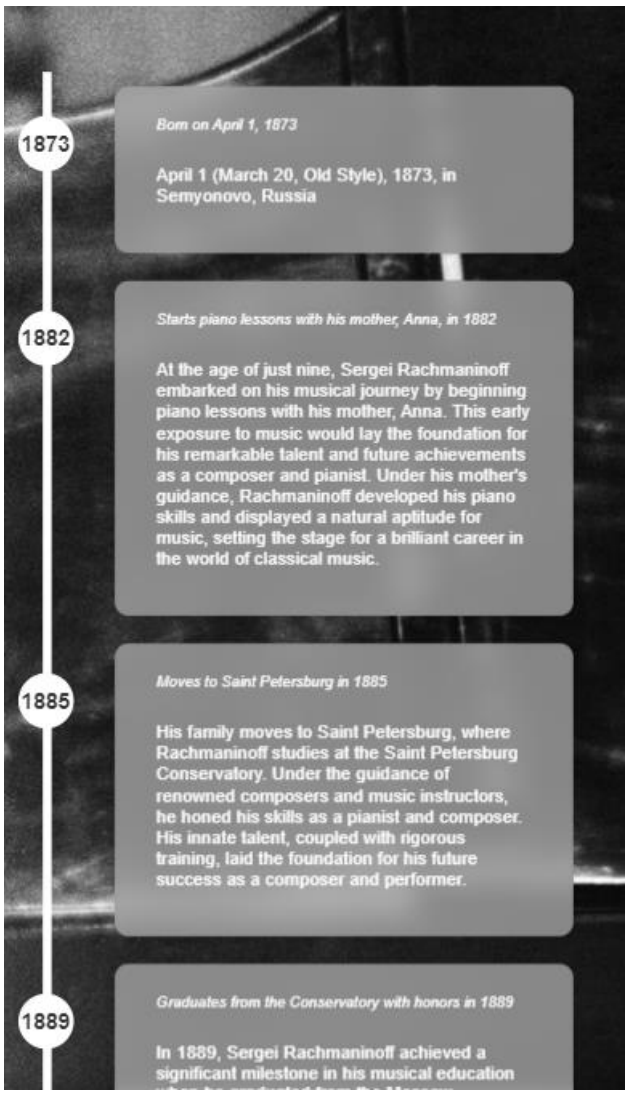

Рисунок 5.2 – адаптивная версия для мобильных устройств
[разработано автором]

6.ВЫВОД

Работа над интернет-ресурсом "150 лет С.В. Рахманинову" представляет собой значительный этап в исследовании и представлении творчества великого композитора. В ходе разработки сайта были достигнуты важные моменты и достижения, представляющие собой значимый вклад в изучение и популяризацию творчества С.В. Рахманинова.

Основные Достижения:

Создание Интерактивной Платформы: Разработана и внедрена интерактивная платформа, предоставляющая посетителям уникальную возможность погружения в мир музыки и биографии С.В. Рахманинова.

Клиентская Часть Сайта: Реализована клиентская часть ресурса, включая главную страницу с обзором, страницы с пианино, биографией, музыкальным и видео контентом.

Биография и Творчество: Подробно исследована и представлена биография композитора, а также его великолепное творчество, включая аудио и видео материалы.

Интерактивные Элементы: Добавлены интерактивные элементы, такие как пианино для виртуального исполнения, аудио- и видео-плееры, обеспечивающие более глубокое взаимодействие посетителей с контентом.

Учебный Опыт и Выводы:

Расширение Знаний: В ходе работы были углублены знания о технологиях веб-разработки, создании интерфейсов, адаптации сайтов для мобильных устройств.

Исследование Музыкального Наследия: Исследование творчества С.В. Рахманинова расширило понимание его вклада в мировую музыку и стимулировало интерес к классическому музыкальному наследию.

Применение Интерактивных Элементов: Опыт создания интерактивных элементов, таких как музыкальные инструменты на сайте, обогатил навыки в области веб-разработки.

Перспективы на будущее:

Расширение Функционала: В будущем возможно расширение функционала сайта, добавление новых разделов и материалов, углубляющих понимание творчества композитора.

Развитие Образовательного Контента: Сайт может стать образовательным ресурсом для любителей музыки, обеспечивая уникальный опыт в изучении жизни и творчества С.В. Рахманинова.

Сообщество Любителей Музыки: Создание сообщества, где любители музыки могут обмениваться впечатлениями, обсуждать произведения и делиться своим исполнением на виртуальном пианино.

Создание интернет-ресурса "150 лет С.В. Рахманинову" стало не только техническим и творческим достижением, но и возможностью погрузиться в мир великого композитора, делиться этим опытом с другими и внести свой вклад в сохранение и популяризацию классической музыки.

7.СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Berners-Lee T., Connolly D. RFC1866: Hypertext Markup Language-2.0. – 1995.
2. Lie H. W., Bos B. Cascading style sheets, level 1. – 1996.
3. Artzi S. et al. A framework for automated testing of JavaScript web applications //Proceedings of the 33rd International Conference on Software Engineering. – 2011. – С. 571-580.
4. Raggett D. et al. HTML 4.01 Specification //W3C recommendation. – 1999. – Т. 24.
5. Сейдаметов Г. С., Биктимиров Р. Р. Использование css grid layout при построении сеток сайта //Информационно-компьютерные технологии в экономике, образовании и социальной сфере. – 2018. – №. 3. – С. 51-58.
6. Bray T. (ed.). RFC 8259: The JavaScript object notation (JSON) data interchange format. – 2017.

Все источники были использованы в процессе написания данного текста и предоставляют информацию о HTML, CSS, JavaScript, а также о популярных библиотеках и фреймворках для разработки веб-приложений.

8.ПРИЛОЖЕНИЕ А

Разработанный код программы представлен в виде ZIP-файла “site.zip”. Также сайт был опубликован с помощью Github.