МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ  
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ  
ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий  
Кафедра Программной инженерии  
Специальность 1-40 80 05 «Программная инженерия»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА КУРСОВОГО ПРОЕКТА**

по дисциплине «Компьютерные языки разметки»

Тема «Веб-сайт для театра»

**Исполнитель**

студент 1 курса 9 группы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ч. Н. Юрьевич

подпись, дата

**Руководитель**

преподаватель-стажер \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. В. Комкова

должность, учен. степень, ученое звание подпись, дата

Допущен(а)к защите \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

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

дата, подпись

Курсовой проект защищен с оценкой

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_ А. В. Комкова

подпись дата инициалы и фамилия

Содержание

[Введение 3](#_Toc197962838)

[1 Постановка задачи 5](#_Toc197962839)

[1.1 Обзор аналогичных решений 5](#_Toc197962840)

[1.1.1 Сайт rtbd.by 5](#_Toc197962841)

[1.1.2 Сайт slonimteatr.by 6](#_Toc197962842)

[1.1.3 Сайт bolshoibelarus.by 6](#_Toc197962843)

[1.1.4 Сайт beltuz.by 7](#_Toc197962844)

[1.2 Техническое задание 8](#_Toc197962845)

[1.3 Выбор средств реализации программного продукта 8](#_Toc197962846)

[1.4 Вывод 9](#_Toc197962847)

[2 Проектирование страниц веб-сайта 10](#_Toc197962848)

[2.1 Выбор способа вёрстки 10](#_Toc197962849)

[2.2 Выбор стилевого оформления 10](#_Toc197962850)

[2.3 Выбор шрифтового оформления 11](#_Toc197962851)

[2.4 Разработка логотипа 11](#_Toc197962852)

[2.5 Разработка пользовательских элементов 12](#_Toc197962853)

[2.6 Разработка спецэффектов 13](#_Toc197962854)

[2.7 Выводы 15](#_Toc197962855)

[3 Реализация структуры веб-сайта 16](#_Toc197962856)

[3.1 Структура HTML-документа 16](#_Toc197962857)

[3.2 Добавление таблиц стилей SCSS и CSS 17](#_Toc197962858)

[3.3 Использование стандартов XML (SVG) 18](#_Toc197962859)

[3.4. Управление элементами DOM 19](#_Toc197962860)

Введение

В условиях активного развития информационных технологий создание веб-сайтов для учреждений культуры становится неотъемлемой частью их присутствия в цифровом пространстве. Современный веб-сайт позволяет театру не только распространять информацию о постановках, но и взаимодействовать с посетителями, формировать афишу, публиковать новости и предоставлять доступ к актуальным событиям в удобной форме. Курсовой проект посвящён разработке многостраничного веб-сайта театра с динамической афишей и новостным разделом.

Разработка функционального и адаптивного веб-сайта для театра, обеспечивающего пользователям удобный доступ к информации о спектаклях и новостях, реализованного с использованием современных веб-технологий (HTML, SCSS, JavaScript, JSON).

1. Анализ аналогичных решений.
2. Выбор способа верстки.
3. Выбор стилевого оформления.
4. Разработка логотипа.
5. Разработка пользовательских элементов.
6. Разработка спецэффектов.
7. Добавление веб-сайту адаптивности.
8. Добавление веб-сайту кроссбраузерности.
9. Тестирование веб-сайта.
10. Создание руководства пользователя.

Потенциальные зрители театра, желающие ознакомиться с расписанием постановок, прочитать новости, а также получить общую информацию о театре. Основная задача сайта — сделать взаимодействие с театром простым и удобным.

Актуальность обусловлена необходимостью представления театра в цифровой среде, расширения его аудитории, а также повышения оперативности распространения информации. Веб-сайт позволяет автоматизировать процесс публикации афиши, обновления новостей и представления информации о театре, тем самым улучшая взаимодействие между учреждением культуры и его посетителями.

1 Постановка задачи

1.1 Обзор аналогичных решений

Для анализа были рассмотрены официальные сайты белорусских театров. Данные ресурсы обеспечивают посетителям доступ, главным образом, к афише спектаклей, информации о постановках, новостях, а также общей театре. Далее приведён анализ конкретных примеров веб-сайтов с визуальными материалами.

1.1.1 Сайт rtbd.by

Изображение выглядит как текст, снимок экрана, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.На рисунке 1.1 внешний вид верхней части главной страницы сайта [1]. В верхней части страницы расположено навигационное меню и логотип, в мобильной версии меню реализовано в виде «гамбургера», раскрывающегося на весь экран. Ниже можно увидеть информация об одной из актуальных постановок, ниже на сайте расположены краткие сведения о самом театре, раздел новостей и афиша.

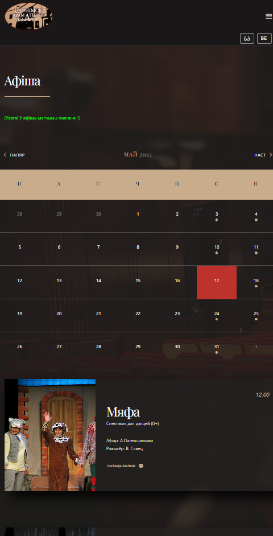
а б

а – десктопная версия, б – мобильная

Рисунок 1.1 – Главная страница веб-сайта rtbd.by

1.1.2 Сайт slonimteatr.by

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

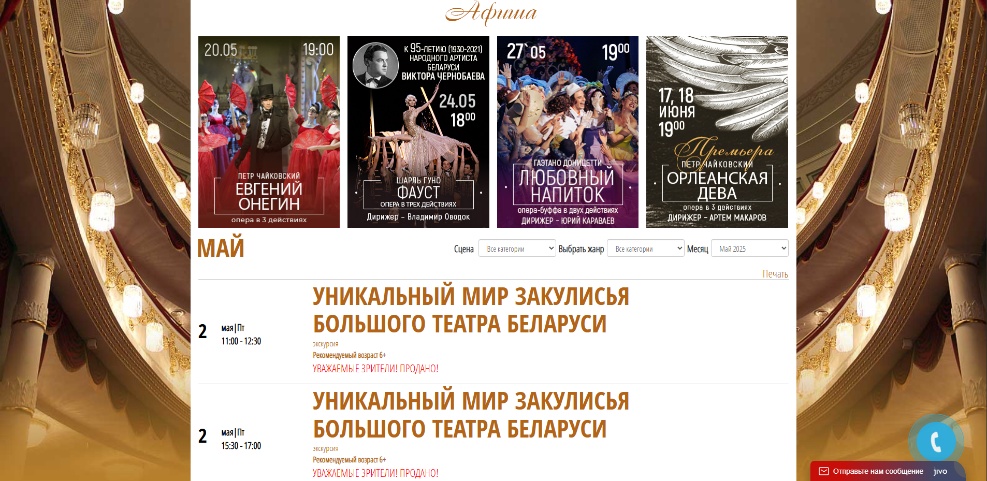
Контент, сгенерированный ИИ, может содержать ошибки.На рисунке 1.2 внешний вид афиши сайта [2]. Раздел «Афиша» содержит удобный интерактивный календарь для поиска спектаклей по дате. Оформление визуально привлекательное, с линейным представлением списка событий. Структура проста, но эффективна.

а б

а – десктопная версия, б – мобильная

Рисунок 1.2 – Афиша веб-сайта slonimteatr.by

1.1.3 Сайт bolshoibelarus.by

Изображение выглядит как текст, снимок экрана, человек, Веб-сайт

Контент, сгенерированный ИИ, может содержать ошибки.На рисунке 1.3 внешний вид афиши сайта [3]. Страница веб-сайта предлагает список доступных билетов с функциями сортировки по жанрам, месту проведения и месяцу.

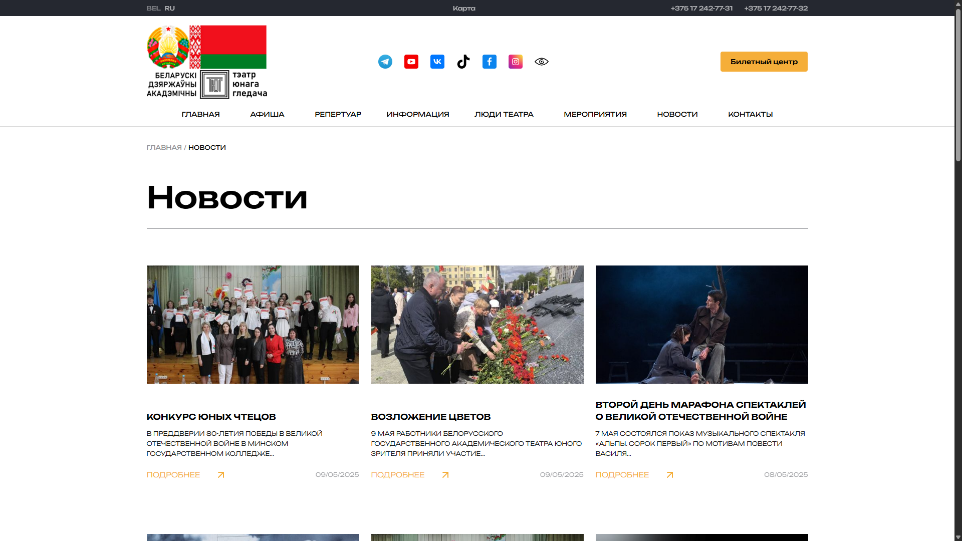
а б

а – десктопная версия, б – мобильная

Рисунок 1.3 – Афиша веб-сайта bolshoibelarus.by

1.1.4 Сайт beltuz.by

Изображение выглядит как текст, одежда, снимок экрана, человек

Контент, сгенерированный ИИ, может содержать ошибки.На рисунке 1.4 внешний вид странички с новостями сайта [4]. На странице, плиточной раскладкой размещены блоки, с кратким описанием новостей (картинка, заголовок, краткое описание, дата и ссылка на саму новость) без возможности фильтрации.

а б

а – десктопная версия, б – мобильная

Рисунок 1.4 – Новости веб-сайта beltuz.by

Общими чертами современных театральных сайтов является:

* наличие афиши с календарной навигацией;
* раздел с новостями и анонсами;
* адаптивность под различные устройства;
* визуально насыщенный и тематически оформленный дизайн;
* использование динамически подгружаемых данных (например, через JSON-файлы).

К недостаткам некоторых сайтов можно отнести:

* перегруженность визуального интерфейса;
* отсутствие упрощённой мобильной версии;

Данный анализ позволил выделить ключевые элементы, которые необходимо реализовать в рамках данного курсового проекта: простота восприятия информации, интуитивная навигация, актуальная афиша, современный дизайн и интерактивность.

1.2 Техническое задание

Разрабатываемый веб-сайт включает 6 основных HTML-страниц:

* index.html — главная страница с краткой информацией о театре, афишей и последними новостями;
* about.html — подробная информация о театре, его истории и коллективе;
* events.html — список доступных постановок;
* events-page.html — детальное описание выбранного спектакля;
* news.html — список актуальных новостей;
* news-page.html — полное содержание выбранной новости.

Все страницы оформлены в едином стиле, с общей навигационной панелью и футером.

Функциональные требования к продукту:

* динамическая загрузка данных о спектаклях и новостях из файлов JSON (events.json, news.json);
* реализация интерактивного календаря для выбора даты и отображения слайдера с соответствующими этой дате постановками;
* адаптивная верстка, корректная работа на экранах различной ширины;
* отображение новостей и постановок с возможностью перехода на полное описание.

Задачи, реализуемые программным продуктом:

* загрузка и отображение слайдера с постановками в зависимости от выбранной даты;
* взаимодействие с DOM и создание динамических элементов;
* использование современных стандартов, вёрстки и стилизации (SCSS, SVG).

1.3 Выбор средств реализации программного продукта

Для реализации веб-сайта использован следующий стек технологий:

* HTML5 — язык разметки для создания структуры веб-страниц;
* SCSS (Sass) — препроцессор CSS для модульной и удобной стилизации;
* JavaScript— язык программирования для динамического взаимодействия с DOM;
* JSON — формат для хранения и загрузки данных о событиях и новостях;
* SVG — векторная графика для отображения иконок и декоративных элементов.

Используемые инструменты:

* Visual Studio Code — основная среда разработки;
* Figma — создание макетов пользовательского интерфейса;
* GitHub — хостинг исходного кода и версионный контроль;
* Live Server — локальный просмотр сайта в процессе разработки.

Библиотеки не используются: весь функционал реализуется на "чистом" JavaScript для соблюдения требований дисциплины и демонстрации владения базовыми веб-технологиями.

1.4 Вывод

В результате анализа аналогичных сайтов были определены основные функции, которые должны быть реализованы в проекте: динамическая афиша, новостной раздел, адаптивный дизайн и удобный интерфейс. Сформировано техническое задание с определением структуры и функционала сайта. Для реализации проекта был выбран современный и лёгкий стек технологий: HTML5, SCSS, JavaScript и JSON. Акцент сделан на пользовательском удобстве, доступности и визуальной привлекательности интерфейса.

2 Проектирование страниц веб-сайта

Проектирование веб-сайта началось с разработки макета в графическом редакторе Figma, где были определены ключевые зоны интерфейса, стилистика, структура и расположение элементов. Макет строился на основе 12-колоночной сетки с шириной колонок 80 пикселей и отступами 20 пикселей, что обеспечило визуальную согласованность и гибкость верстки. Все блоки были спроектированы с учётом адаптивности для двух основных разрешений: 1024px, и 480px.

Сайт разрабатывался в тёмной цветовой теме, с фокусом на визуальную контрастность, строгую типографику и акценты на ключевых интерактивных элементах — афише и новостях.

2.1 Выбор способа вёрстки

При вёрстке всех страниц активно использовалось свойство display: flex, которое позволяло задавать направление, выравнивание и распределение элементов в контейнерах. Flex-контейнеры применялись для построения навигации, расположения карточек событий, элементов футера и шапки.

Медиа-запросы (@media) были задействованы для адаптации сайта под различные устройства. Использовались три ключевых точки перелома:

* 1024px — для планшетов и ноутбуков;
* 768px — для стандартных планшетов и малых экранов;
* 480px — для смартфонов.

Вёрстка не использовала CSS Grid или сторонние библиотеки и была построена вручную на чистом HTML и SCSS с использованием Flex-модели.

2.2 Выбор стилевого оформления

Общее стилевое решение сайта — минималистичное, выполненное в тёмной цветовой гамме с контрастными элементами. Интерфейс не перегружен деталями и обеспечивает акцент на функциональных блоках. На главной странице и в разделах между блоками предусмотрены отступы, сегментирующие контент и создающие визуальную иерархию.

Используемая цветовая палитра представлена на рисунке 2.1:

* основной фон: #010000;
* акценты: # A80000 — для кнопок, выделений;
* ссылки: #6b6bff, #d4a373;
* заголовки: #ffffff;
* Изображение выглядит как снимок экрана, текст, Красочность

  Контент, сгенерированный ИИ, может содержать ошибки.обычный текст: #828282.

Рисунок 2.1 – Основная цветовая палитра

2.3 Выбор шрифтового оформления

В проекте используются три шрифта, каждый из которых выполняет свою функциональную роль в визуальной иерархии интерфейса:

* **Molot** — применяется для выделения акцентных элементов, таких как контактный номер номера и ссылка на список билетов;
* **PT Sans** — используется в качестве основного текста на всех страницах сайта;
* **Cormorant Italic** — применяется для оформления заголовков, подзаголовков, дат в новостях, а также в шапке сайта.

Рисунок 2.2 – Используемые шрифты и начертания

Выбранные шрифты обладают поддержкой кириллицы и представлены в различных начертаниях: *regular*, *bold*, *italic*, что позволяет гибко управлять стилистикой и акцентами контента. Основной размер текста составляет 24 px, заголовков — 48 px, в зависимости от уровня вложенности и важности информации. Цветовое оформление текста варьируется от белого до светло-серого, в зависимости от контраста с фоном.

2.4 Разработка логотипа

На рисунке 2.3 представлен логотип проекта, он был взят из открытого источника и доработан в векторном редакторе. В процессе адаптации были:

* удалены лишние элементы (текст и графические детали);
* изменены цвета в соответствии с тёмной темой сайта;
* улучшен визуальный баланс и читаемость.

Логотип выполнен в виде стилизованного изображения театрального здания, включающего архитектурные элементы — арочные окна, декоративные уровни, центральный вход. Композиция заключена в круглую рамку с декоративными штрихами, напоминающими античные орнаменты. Благодаря векторному формату (SVG), логотип сохраняет чёткость на любых разрешениях.

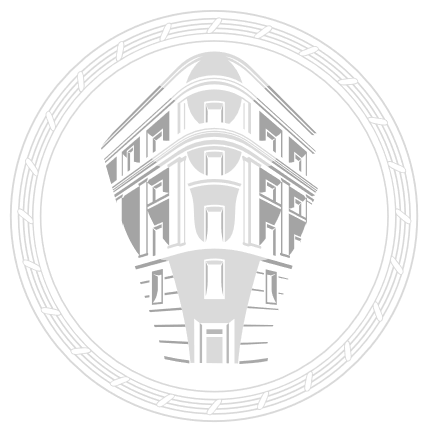


Рисунок 2.3 – Логотип

2.5 Разработка пользовательских элементов

В проекте будут представлены элементы пользовательского интерфейса для взаимодействия: навигационное меню с адаптацией под мобильные устройства (гамбургер-меню) и календарь выбора даты с отображением афиши – слайдера постановок.

Навигация – ключевой элемент удобства сайта, обеспечивающий перемещение между разделами. На мобильных устройствах навигационное меню представлено в виде гамбургер-иконки, раскрывающей пункты («Главная», «О театре», «Билеты», «Новости»), как показано на рисунке 2.2.

Изображение выглядит как текст, снимок экрана, черный

Контент, сгенерированный ИИ, может содержать ошибки.а б

в

а – десктопная версия, б – мобильная свернутая, в – мобильная раскрытая

Рисунок 2.2 – Навигационное меню

На рисунке 2.3 представлен календарь выбора даты интегрированный со слайдером – афишей, отображающим постановки на выбранную дату.

Элемент включает:

* выбор даты в сетке календаря с выделением активной даты (например, 12 мая 2025);
* слайдер с постерами, названиями и временем событий (например, «Премьера 'Ромео и Джульетта'»).

Изображение выглядит как текст, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.а

б в

а – десктопная версия, б – мобильная версия, в – мобильная версия

Рисунок 2.4 – интерактивный календарь и афиша – слайдер

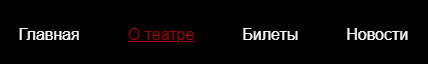
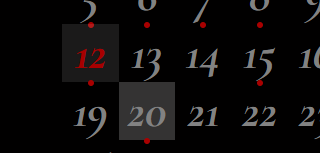
Эти пользовательские элементы обеспечивают удобство навигации и быстроту доступа к информации о подходящих постановках.

2.6 Разработка спецэффектов

Для повышения интерактивности и улучшения визуального восприятия веб-сайта были реализованы динамические CSS-эффекты, применяемые при наведении курсора на элементы. Эти спецэффекты делают взаимодействие с интерфейсом более живым и интуитивным для пользователя, а также подчёркивают активные области на странице.

На сайте использовались следующие виды анимаций:

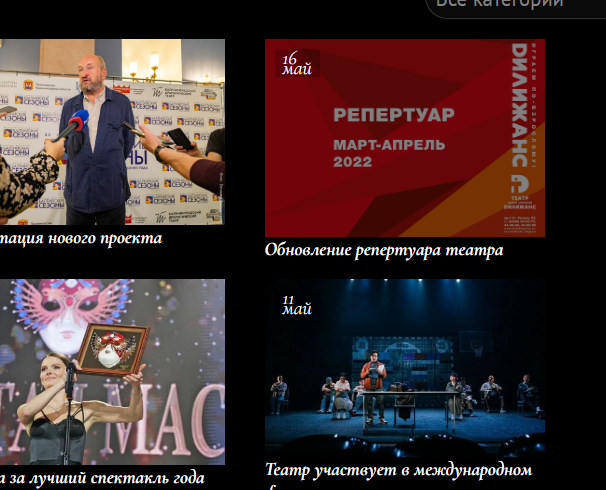
* Изменение цвета текста и фона — позволяет визуально выделить наведение и сделать интерфейс отзывчивым. Примеры представлены на рисунках 2.4, 2.5, 2.6, 2.9, 2.11.
* Масштабирование элементов (transform: scale()) — используется для стрелок и иконок соцсетей, визуально увеличивая их при наведении. Примеры приведены на рисунках 2.6, 2.7.
* Изменение прозрачности (opacity) — применяется для акцента на выборе новости и улучшении видимости даты, иконок или фона. Показано на рисунке 2.8.
* Смещение элементов (transform: translate()) — создаёт эффект лёгкого движения объекта, используется в списке билетов и блоках последних новостей. Отражено на рисунках 2.9, 2.10.

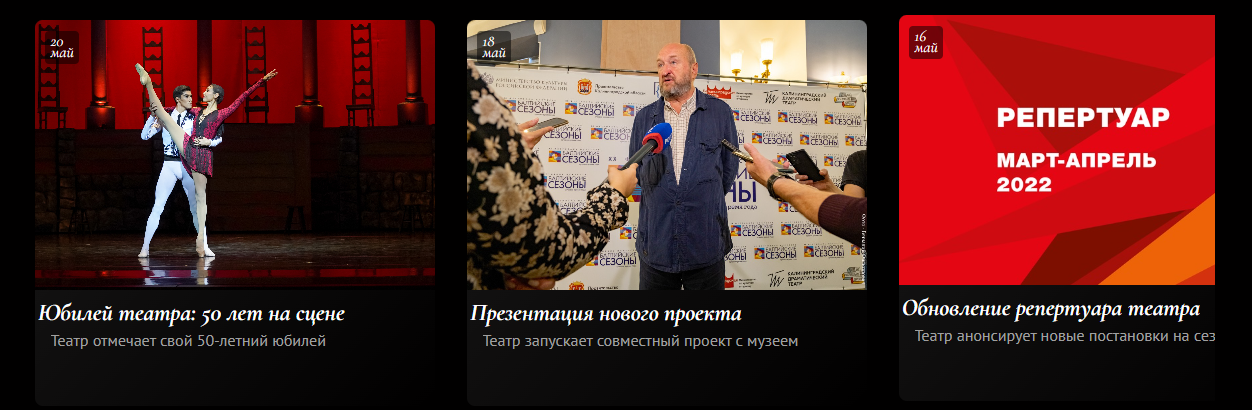
Рисунок 2.5 – Анимация навигационного меню

Изображение выглядит как текст, Шрифт, снимок экрана, Графика

Контент, сгенерированный ИИ, может содержать ошибки. Рисунок 2.6 – Анимация выбора даты

Рисунок 2.7 – Анимация иконок соцсетей Рисунок 2.8 – Анимация слайдера

Рисунок 2.9 – Анимация картинки новости

Рисунок 2.10 – Анимация билета и кнопки бронирования

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.Рисунок 2.11 – Анимация блока последней новости

Рисунок 2.12 – Анимация списка других новостей и ссылки «Назад»

Все анимации реализованы с использованием свойства transition, что обеспечивает плавность и не перегружает интерфейс. Эффекты активируются при наведении и не мешают восприятию информации на мобильных устройствах.

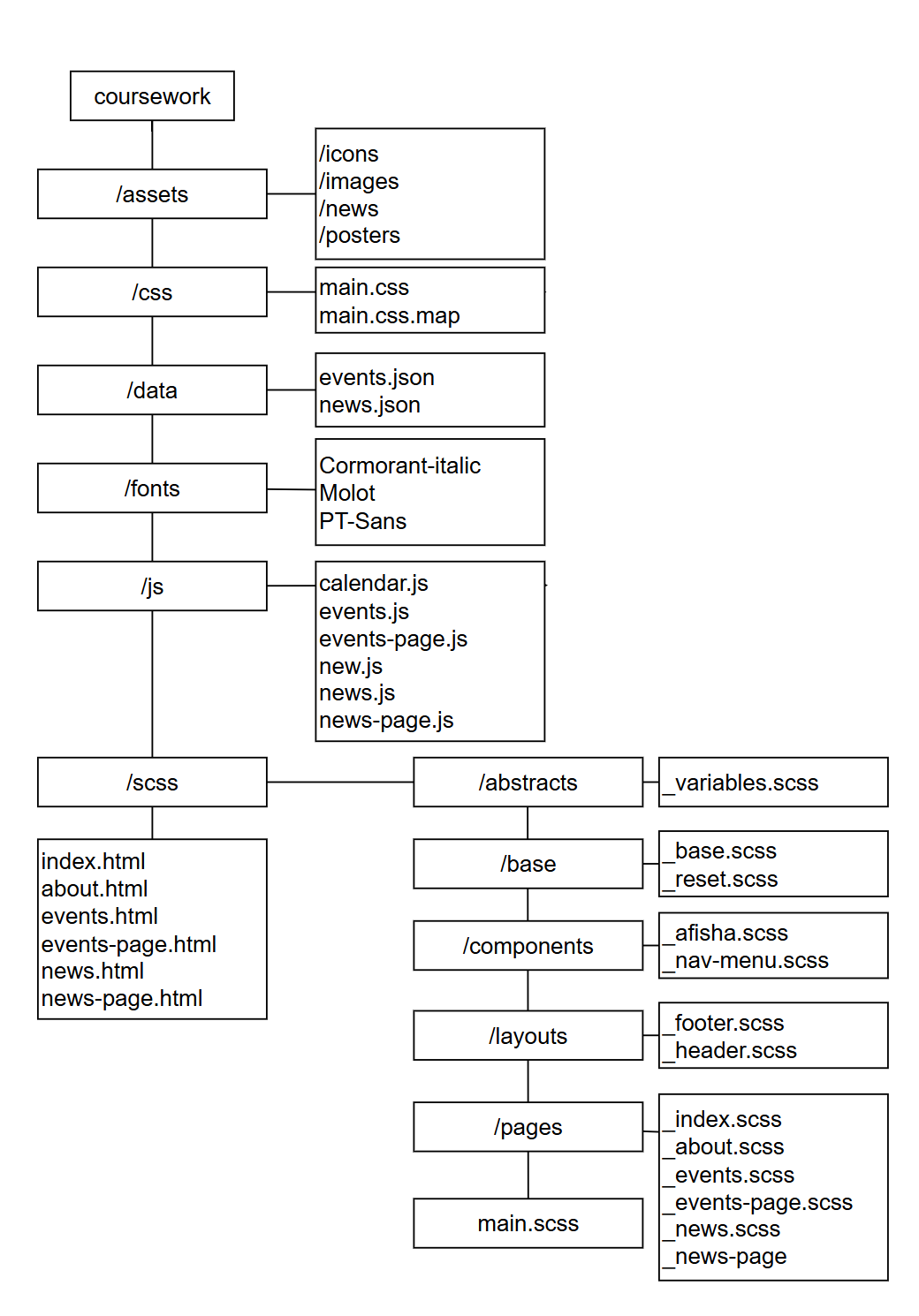
2.7 Выводы

Разработка макета в Figma и чёткое проектирование структуры сайта обеспечили единообразие интерфейса, логичное расположение блоков и согласованность визуальных решений. Выбранные инструменты и стилистика (Flexbox, SCSS, шрифтовая и цветовая палитра) позволили реализовать удобный, эстетичный и функциональный веб-сайт в рамках заданной тематики. Методология БЭМ и модульная организация стилей обеспечили чистоту кода и его дальнейшую масштабируемость.

3 Реализация структуры веб-сайта

Разрабатываемый веб-сайт представляет собой многостраничный проект, в котором каждая HTML-страница выполняет строго определённую функцию: отображение главной страницы, списка билетов, новостей, общей информации о театре и подробностей по конкретным событиям. Структура папок и файлов организована логично и модульно, что значительно упростило как текущую разработку, так и последующее сопровождение и масштабирование проекта, схематически она продемонстрирована на рисунке 3.1.

Весь исходный код проекта представлен в приложениях В–Ж, где приведены листинги HTML-документов, SCSS-стилей, JSON-файлов с данными и JavaScript, обеспечивающий динамическую функциональность сайта.

Рисунок 3.1 – Схематическое изображение структуры папок и файлов проекта

3.1 Структура HTML-документа

Веб-сайт реализован в виде многостраничной структуры, где каждая страница отвечает за конкретную функциональность: отображение информации о театре, списке билетов, новостей и их подробностей. Все страницы используют единый HTML-шаблон, основанный на принципах семантической вёрстки HTML5. Общий принцип его структуры представлен в листинге 3.1

<!DOCTYPE html>

<html lang="ru">

<head>...</head>

<body>

<header class="header">...</header>

<nav class="nav"> ...</nav>

<main class="main">

<section>...</section>

...

</main>

<footer class="footer">...</footer>

<script src="js/calendar.js"></script>

<script src="js/new.js"></script>

</body>

</html>

Листинг 3.1 – Общая HTML структура главной страницы

Обоснование выбранной структуры:

* Использование семантических HTML5-элементов (<header>, <nav>, <main>, <section>,<article>, <footer>) улучшает читаемость кода, облегчает навигацию и повышает SEO-оптимизацию сайта.
* Применение мета-тегов в <head> (кодировка, масштаб, описание, key-words) способствует отзывчивости сайта, улучшает его индексирование поисковыми системами и корректное отображение в социальных сетях.
* Ясное разделение на функциональные зоны (шапка, навигация, контент, подвал) обеспечивает структурированность и логичную архитектуру страниц.
* Весь HTML-код оформлен с применением методологии BEM, классы имеют вид block\_\_element\_modifier. Такой подход обеспечивает модульность, лёгкость в сопровождении и масштабировании проекта, а также высокую читаемость кода.

Навигация между страницами схематически представлена на рисунке 3.2, и устроена следующим образом:

* главная страница (index.html) содержит ссылки на разделы "О театре", «Список билетов» и «Новости.
* раздел "Список билетов" ведёт к карточкам спектаклей, каждая из которых может быть раскрыта на отдельной странице с подробным описанием (events-page.html).
* раздел "Новости" включает список новостных карточек с переходом на полную новость (news-page.html).
* раздел "О театре" предоставляет информацию о деятельности, истории и составе театра.

Изображение выглядит как текст, диаграмма, линия, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.Рисунок 3.2 – Схематической изображение структуры веб-сайта

Кроме того, на каждой странице сайта присутствует ссылка на главную страницу, реализованная через логотип театра, представленный в формате SVG и обернут в элемент <a> с атрибутом href="index.html", что позволяет пользователям быстро вернуться на главную страницу из любой точки сайта. Эта функция обеспечивает единообразную навигацию. Так же на странице события предусмотрена ссылка «Назад», которая позволяет пользователю вернуться на предыдущую страницу (например, главную страницу или страницу билетов). Ссылка реализована через элемент <a> с атрибутом href="javascript:history.back()", что вызывает метод history.back() для возврата в истории браузера. Это обеспечивает простую и интуитивную навигацию, соответствующую ожиданиям пользователей.

3.2 Добавление таблиц стилей SCSS и CSS

В ходе разработки проекта был применен внешний подходы к подключению таблиц стилей, представленный на листинге 3.3 и строчный на листинге 3.4.

<!DOCTYPE html>

<html lang="ru">

<head>

...

<link rel="stylesheet" href="css/main.css" />

</head>

<body>...</body>

</html>

Листинг 3.3 – Пример внешнего подключения

<main id="newsPage" style="display: none">

Листинг 3.4 – Пример строчного подключения

Стилизация проекта организована в SCSS-файлах, разделённых на следующие модули:

* abstracts/ — переменные и общие настройки (\_variables.scss);
* base/ — сброс и базовые стили (\_reset.scss, \_base.scss);
* layouts/ — стили для header и footer;
* components/ — отдельные элементы, такие как интерактивный календарь и навигация;
* pages/ — стили конкретных страниц (\_index.scss, \_events.scss, и т.д.).

Файл main.scss объединяет все модули и компилируется в main.css, который подключается к страницам. Все стили написаны вручную, без сторонних библиотек. Использование SCSS позволило применить вложенность, сокращения и изоляцию компонентов.

3.3 Использование стандартов XML (SVG)

В проекте используются векторные изображения формата SVG — в логотипе, иконках меню и других декоративных элементах. SVG-графика подключалась как через теги <img>, как на листинге 3.4, так и через inline – вставки, как на листинге 3.5.

<img src="assets/icons/instagram.svg" alt="Instagram" />

Листинг 3.4 – Подключение графики с помощью тега img

<path

                  d="M 45.46644,98.758703 c 0.379518,-0.443927 1.308722,-1.818548 -0.0237,-3.478641 C 34.020448,80.626907 22.392465,66.174792 11.006561,51.488137 c -0.549627,-0.750106 -0.549612,-1.896596 0,-2.646714 C 22.470045,34.232022 33.961985,19.642514 45.443205,5.0479173 46.4656,3.7000743 46.18991,2.2987443 45.704828,1.5485503 45.154656,0.80928733 43.578236,0.21832333 42.541821,1.5070303 30.268782,17.466036 17.439538,32.893359 5.2667372,48.943056 c -0.463199,0.718135 -0.463395,1.725972 0,2.443973 12.3033598,15.746967 24.6892308,31.436443 37.0435938,47.140718 1.344021,1.779003 2.726055,0.672688 3.156109,0.230956 z"

                  style="fill: #828282"

                />

Листинг 3.5 – Подключение графики с помощью inline SVG

Преимущества SVG:

* масштабируемость без потери качества;
* возможность анимации и стилизации через CSS;
* низкий вес и высокая скорость загрузки.

3.4. Управление элементами DOM

На главной странице реализован календарь, позволяющий пользователям просматривать события на разные даты. Код находится в файле calendar.js, и его включает следующую функциональность.

Загрузка данных: Функция loadEvents() асинхронно загружает события из файла events.json с помощью fetch. Данные структурируются в объект eventsByDate, где ключами являются даты в формате YYYY-MM-DD, а значениями — массивы событий. Это обеспечивает эффективный доступ к событиям по дате.

async function loadEvents() {

try {

const response = await fetch('./data/events.json')

if (!response.ok) {

throw new Error(`HTTP error! Status: ${response.status}`)

}

const data = await response.json()

eventsByDate = {}

if (!Array.isArray(data)) {

throw new Error('Данные событий не являются массивом')

}

data.forEach(event => {

if (!eventsByDate[event.date]) {

eventsByDate[event.date] = []

}

eventsByDate[event.date].push(event)

})

return data

} catch (error) {

console.error('Ошибка загрузки событий:', error)

return []

}

}

Листинг 3.6 – Загрузка данных из JSON

Генерация календаря: Функция createCalendar() создает HTML-код для календаря текущего месяца. Она включает навигационные кнопки для перехода к предыдущему и следующему месяцам, таблицу с днями недели и датами, а также выделение сегодняшней даты и дат с событиями. Каждая дата является кликабельной ячейкой, и при клике на нее обновляется слайдер событий. Код представлен в Приложении \* Листинг \*.

Слайдер событий: Функция updateEventSlider() отображает события для выбранной даты. Если событий нет, показывается сообщение "Нет событий на эту дату". В противном случае, события отображаются в слайдере с возможностью переключения между ними, используя классы для анимации, такие как slide-in-left и slide-in-right.Навигация по слайдеру: Функции setupSliderNavigation() и updateSlides() обрабатывают клики на кнопках "Предыдущее событие" и "Следующее событие", обеспечивая плавную смену слайдов. Используются события transitionend для управления анимацией.

Обновление календаря: Функция updateCalendar(direction) позволяет переключаться между месяцами при клике на навигационные кнопки, обновляя отображение и сбрасывая выбранную дату.

function updateCalendar(direction) {

  if (direction === 'prev') {

    currentMonth--

    if (currentMonth < 1) {

      currentMonth = 12

      currentYear--

    }

  } else if (direction === 'next') {

    currentMonth++

    if (currentMonth > 12) {

      currentMonth = 1

      currentYear++

    }

  }

  createCalendar('#calendar', currentYear, currentMonth)

  selectedDate = null

  updateEventSlider()

}

Листинг 3.7 – Функция обновления календаря

Вставка данных в шаблонную страницу, например (events-page.html) реализована через JavaScript: параметр title извлекается из URL, данные события загружаются из events.json с помощью fetch, и элементы страницы (.event\_\_image, .event\_\_title и др.) заполняются соответствующими значениями. При ошибке отображается сообщение. Пример JavaScript кода представлен в листинге 3.8, а JSON файла в листинге 3.9.

document.addEventListener('DOMContentLoaded', () => {

const urlParams = new URLSearchParams(window.location.search)

const eventTitle = urlParams.get('title')

if (!eventTitle) return console.error('No title in URL')

  fetch('./data/events.json')

    .then(response => {

      if (!response.ok) throw new Error(`HTTP error: ${response.status}`)

      return response.json()

    })

    .then(data => {

      const event = data.find(e => e.title === decodeURIComponent(eventTitle))

if (!event) return console.error('Event not found')

    document.querySelector('.event\_\_image').src = event.image || ''

...

    })

    .catch(error => {

      console.error('Error:', error)

      document.querySelector('.event').innerHTML =

        '<p>Failed to load event.</p>'

    })

})

Листинг 3.8 – Вставка данных

[

  {

    "image": "./assets/posters/spring\_awakening.jpg",

    "title": "Премьера 'Весеннее пробуждение'",

    "type": "опера",

    "age": "12+",

    "date": "2025-05-05",

    "time": "19:00",

    "director": "Иван Петров",

    "price": "20 - 25р.",

    "description": "Новая опера о возрождении и надежде, вдохновляющая зрителей своим музыкальным размахом.",

    "cast": "Анна Сидорова, Михаил Козлов, Елена Борисова"

  },

...

]

Листинг 3.9– Структура данных Новостей

Код использует современные возможности JavaScript, такие как async/await для асинхронной загрузки данных и DOM-манипуляцию для динамического обновления содержимого страницы. Это обеспечивает интерактивность и удобство использования для пользователей.