

РОСЖЕЛДОР
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Ростовский государственный университет путей сообщения»
(ФГБОУ ВО РГУПС)

ОСНОВЫ ВЕБ-РАЗРАБОТКИ И ВЕРСТКИ

Учебно-методическое пособие
для выполнения лабораторных работ

для специальности
09.02.09 Веб-разработка

Ростов-на-Дону
2025

Рецензент – кандидат технических наук, доцент А.А. Феденко

Муконина, М. И.

Основы веб-разработки и верстки: [Электронный ресурс] учебно-методическое пособие для лабораторных работ / М. И. Муконина; ФГБОУ ВО РГУПС. – Ростов-на-Дону : РГУПС, 2025. – 202 с.

Учебно-методическое пособие разработано для обучающихся по программам среднего профессионального образования (СПО) в сфере информационных технологий, веб-разработки и смежных дисциплин. Его ключевая задача – сформировать у студентов практические компетенции в области вёрстки и дизайна веб-страниц с опорой на современные стандарты HTML5 и CSS.

В процессе работы с пособием обучающиеся последовательно осваивают весь цикл создания адаптивных веб-ресурсов: от первичной разметки до комплексной вёрстки с использованием CSS-технологий и фреймворков. Материал охватывает широкий спектр тем – от базовых понятий (основы HTML-разметки, рабочее окружение веб-разработчика, форматирование текста, работа с гиперссылками и мультимедийным контентом) до продвинутых техник. Студенты учатся строить таблицы и формы для сбора данных, применять семантическую вёрстку через теги HTML5, осваивать базовые и углублённые методы стилизации с помощью CSS (включая работу с селекторами, блочной моделью и позиционированием). Отдельное внимание уделено современным инструментам: вёрстке посредством Flexbox и CSS Grid, принципам адаптивного дизайна (с использованием медиа-запросов и burger-меню), а также работе с CSS-фреймворками – в частности, на примере Bootstrap. Предназначено для студентов специальности среднего профессионального образования 09.02.09 «Веб-разработка», изучающих дисциплину «Основы web-разработки и верстки», а также для студентов всех специальностей, изучающих указанную и смежные дисциплины по веб-разработке. Изложенный материал будет полезен при курсовом и дипломном проектировании.

Одобрено к внесению в «Электронный университет» кафедрой «Вычислительная техника и автоматизированные системы управления».

СОДЕРЖАНИЕ

Лабораторная работа № 1 «Создание первой веб-страницы. Рабочее окружение и базовая HTML-разметка»	5
Лабораторная работа № 2 «Форматирование текста и создание гиперссылок»	13
Лабораторная работа № 3 «Встраивание мультимедийного контента и создание списков»	17
Лабораторная работа № 4 «Построение таблицы данных»	21
Лабораторная работа № 5 «Разработка HTML-формы для сбора данных»...	28
Лабораторная работа № 6 «Применение семантических тегов HTML5 для разметки страницы»	38
Лабораторная работа № 7 «Подключение CSS и применение базовых стилей к семантической странице»	47
Лабораторная работа № 8 «Отработка различных типов CSS-селекторов»..	53
Лабораторная работа № 9 «Работа с блочной моделью: создание карточки товара»	64
Лабораторная работа № 10 «Оформление текстового блока и фоновое изображения»	71
Лабораторная работа № 11 «Позиционирование элементов: создание навигационного меню с выпадающим списком»	77
Лабораторная работа № 12 «Позиционирование элементов: создание фиксированной панели и модального окна»	87
Лабораторная работа № 13 «Верстка макета на Flexbox: шапка сайта и галерея изображений»	98
Лабораторная работа № 14 «Верстка сложного макета на Flexbox: карточка товара и «липкий» футер»	107
Лабораторная работа № 15 «Создание макета страницы с использованием CSS Grid (Часть 1)»	118
Лабораторная работа № 16 «Создание макета страницы с использованием CSS Grid (Часть 2)»	122
Лабораторная работа № 17 «Адаптивная верстка: применение медиа-запросов (@media)»	126
Лабораторная работа № 18 «Создание адаптивной навигации (бургер-меню)»	130

Лабораторная работа № 19 «Создание адаптивного макета с использованием CSS-фреймворка (на примере локального Bootstrap)».....	137
Лабораторная работа № 20 «Разработка пользовательского интерфейса с использованием готовых CSS-компонентов»	144
Лабораторная работа № 21 «Практическая работа: верстка макета главной страницы по готовому шаблону (этап 1 - HTML)»	151
Лабораторная работа № 22 «Практическая работа: верстка макета главной страницы по готовому шаблону (этап 2 - Базовая стилизация)»	169
Лабораторная работа № 23 «Практическая работа: верстка макета главной страницы по готовому шаблону (этап 3 - Детальная стилизация)».....	171
Лабораторная работа № 24 «Практическая работа: верстка макета главной страницы по готовому шаблону (этап 4 - Адаптация)»	173
Лабораторная работа № 25 «Практическая работа: верстка внутренней страницы по готовому шаблону»	175
Лабораторная работа № 26 «Индивидуальный проект: выбор темы и проектирование структуры сайта»	190
Лабораторная работа № 27 «Индивидуальный проект: семантическая HTML-разметка всех страниц».....	192
Лабораторная работа № 28 «Индивидуальный проект: базовая CSS-стилизация и создание общей сетки».....	194
Лабораторная работа № 29 «Индивидуальный проект: детальная стилизация и адаптивность»	196
Лабораторная работа № 30 «Индивидуальный проект: финальная отладка, тестирование и презентация».....	198
БИБЛИОГРАФИЧЕСКИЙ СПИСОК	200

Лабораторная работа № 1

«Создание первой веб-страницы. Рабочее окружение и базовая HTML-разметка»

Цель работы: освоить базовые навыки создания веб-страниц, познакомиться с обязательной структурой HTML-документа и основными элементами разметки.

Методические указания

HTML (HyperText Markup Language – «язык гипертекстовой разметки») - стандартизированный язык разметки веб-страниц во Всемирной паутине.

HTML-документ – это обычный текстовый документ, может быть создан как в обычном текстовом редакторе (Блокнот), так и в специализированном, с подсветкой кода (Notepad++, Visual Studio Code и т.п.). HTML-документ имеет расширение .html.

HTML теги – это специальные команды для браузера. Они говорят ему, что, к примеру, следует считать заголовком страницы, а что абзацем.

Теги строятся по такому принципу: <имя тега>. Имя тега может состоять из английских букв и цифр. Примеры тегов: <h1>, <p>, .

Теги обычно пишутся парами – открывающий тег и соответствующий ему закрывающий. Закрывающий тег образуется путем добавления слэша / перед именем тега: <имя тега> ... </имя тега>. Между начальным и закрывающим тегами находится содержимое элемента – контент.

Элементы могут вкладываться друг в друга, например, <p><i> Текст </i></p>. При вложении следует соблюдать порядок их закрытия.

В языке HTML использование прописных букв в именах элементов не играет большой роли. Так, для браузера теги , и означают одно и то же.

Атрибуты – специальные команды, позволяющие изменять свойства и поведение элемента, для которого они заданы. Атрибуты прописываются в открывающем теге элемента и содержат имя и значение, указываемые в формате имя атрибута="значение".

Каркас документа на языке HTML имеет следующий вид:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Document</title>
</head>
<body>

</body>
</html>
```

Первая строка – объявление типа документа (также называемое **объявление DOCTYPE**), определяющее этот файл как документ на языке HTML5. Его добавление сообщает браузерам: документ необходимо интерпретировать в соответствии со спецификацией HTML5.

Весь документ заключен в элемент `html`. Элемент `html` называется корневым элементом, поскольку он содержит все элементы в документе и не может содержаться ни в каком другом элементе.

Содержимое внутри элемента `html` состоит из головы (раздел заголовка) и тела.

Элемент **head** (раздел заголовка) содержит описательную информацию о документе, такую, как заголовок, используемые таблицы стилей, сценарии и прочие метаданные.

Элементы **meta**, расположенные внутри элемента `head`, предоставляют данные о самом документе. Элемент `meta` можно использовать для предоставления самой разнообразной информации, но в данном случае он определяет кодировку набора символов (стандартизированный набор букв, цифр и символов), применяемую в документе.

Также в разделе заголовка обязателен элемент **title**. В соответствии со спецификацией HTML каждый документ должен содержать описательное название.

Элемент **body** (тело) содержит весь контент, который мы хотим отобразить в окне браузера.

HTML предоставляет множество тегов, каждый из которых выполняет свою функцию. Основными тегами являются:

<h1>, <h2>, ..., <h6> - заголовки разного уровня, используются для структурирования заголовков страницы;

<p> - параграф, используется для отображения текстового контента;

<a> - гиперссылка, создает ссылки на другие веб-страницы или ресурсы;

**** - вставка изображений на веб-страницу;

**, ** - нумерованные и нумерованные списки;

<table> - создание таблиц для представления данных в упорядоченной форме.

Задания для выполнения лабораторной работы

1. Установка и настройка редактора кода.

Скачайте и установите редактор Visual Studio Code с официального сайта.

Создайте новую папку для вашего учебного проекта
ФамилияСтудента_lab_1

Откройте эту папку в VS Code через меню File -> Open Folder...

2. Создание файла index.html

В боковой панели рядом с названием папки нажмите "New File..." и назовите файл index.html.

В открывшейся вкладке с файлом введите восклицательный знак ! и нажмите клавишу Tab. VS Code автоматически сгенерирует базовую структуру HTML5-документа.

Измените язык документа на русский (lang="ru") внутри тега <html>.

Измените текст внутри тега <title> на «Моя первая страница». Этот текст отобразится на вкладке браузера.

3. Добавление заголовка и нескольких параграфов.

Внутри тега <body>, между его открывающей и закрывающей частью, добавьте главный заголовок страницы:

```
<h1>Добро пожаловать в мир веб-разработки!</h1>
```

Под заголовком добавьте два абзаца текста, используя тег `<p>`:

```
<p>Это моя самая первая веб-страница. Здесь я практикую базовую разметку документа.</p>
```

```
<p>HTML – это каркас любого сайта. Он определяет структуру и содержание страницы.</p>
```

4. Просмотр результата.

Сохраните файл (Ctrl+S).

Найдите файл `index.html` в папке проекта и откройте его двойным кликом. Он запустится в вашем браузере по умолчанию (рис. 1.1).

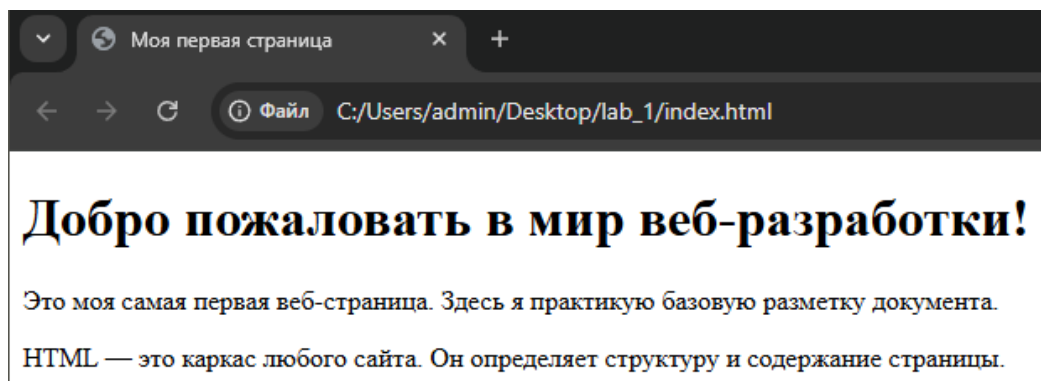


Рисунок 1.1 - Создание первого html-документа

Задание для всех вариантов: Создайте папку с названием `lab1_varN` (где `N` - номер вашего варианта, в заголовке окна браузера укажите "Вариант `N`", где `N` - номер вашего варианта. В теле документа разместите текст по вариантам.

Вариант 1.

Заголовок: История Интернета

Параграф 1: Идея глобальной сети была предложена еще в 1960-х годах.

Параграф 2: Первый успешный обмен данными по ARPANET состоялся в 1969 году.

Параграф 3: Создание World Wide Web (WWW) Тимом Бернерсом-Ли в 1989 году стало революцией.

Вариант 2.

Заголовок: Язык разметки HTML

Параграф 1: HTML расшифровывается как HyperText Markup Language.

Параграф 2: Основная задача HTML – описать структуру веб-страницы.

Параграф 3: Элементы HTML состоят из тегов, атрибутов и содержимого.

Вариант 3.

Заголовок: Каскадные таблицы стилей

Параграф 1: CSS отвечает за внешний вид и оформление веб-страниц.

Параграф 2: Стили можно подключать разными способами: inline, internal, external.

Параграф 3: Использование CSS отделяет содержание документа от его представления.

Вариант 4.

Заголовок: Язык программирования JavaScript

Параграф 1: JavaScript добавляет интерактивность на веб-страницы.

Параграф 2: Этот язык выполняется непосредственно в браузере пользователя.

Параграф 3: С помощью JS можно реагировать на действия пользователя, изменять содержимое страницы.

Вариант 5.

Заголовок: Редакторы кода

Параграф 1: Редактор кода – основной инструмент разработчика.

Параграф 2: Они обеспечивают подсветку синтаксиса и автодополнение.

Параграф 3: Примеры популярных редакторов: VS Code, Sublime Text, WebStorm.

Вариант 6.

Заголовок: Веб-браузеры

Параграф 1: Браузер – это программа для просмотра веб-страниц.

Параграф 2: Он интерпретирует HTML, CSS и JavaScript код.

Параграф 3: Популярные браузеры: Chrome, Firefox, Safari, Edge.

Вариант 7.

Заголовок: Клиент-серверная архитектура

Параграф 1: В основе работы веба лежит модель "клиент-сервер".

Параграф 2: Клиент (браузер) отправляет запрос, сервер его обрабатывает и возвращает ответ.

Параграф 3: Ответом сервера часто является HTML-документ.

Вариант 8.

Заголовок: Структура HTML-документа

Параграф 1: Каждый документ начинается с декларации `<!DOCTYPE html>`.

Параграф 2: Элемент `<html>` является корневым контейнером.

Параграф 3: Документ делится на секции `<head>` (метаданные) и `<body>` (содержимое).

Вариант 9.

Заголовок: Теги заголовков в HTML

Параграф 1: Для заголовков используются теги от `<h1>` до `<h6>`.

Параграф 2: `<h1>` – самый важный заголовок, на странице рекомендуется использовать один.

Параграф 3: Заголовки помогают структурировать текст и важны для SEO.

Вариант 10.

Заголовок: Теги форматирования текста

Параграф 1: Для выделения текста жирным используется тег `` или ``.

Параграф 2: Для курсива – тег `` или `<i>`.

Параграф 3: Смысловые теги (``, ``) предпочтительнее визуальных (``, `<i>`).

Вариант 11.

Заголовок: Атрибуты HTML-элементов

Параграф 1: Атрибуты предоставляют дополнительную информацию об элементе.

Параграф 2: Они указываются внутри открывающего тега в формате имя="значение".

Параграф 3: Примеры: атрибут lang в <html>, charset в <meta>, src в .

Вариант 12.

Заголовок: Гиперссылки в HTML

Параграф 1: Ссылки создаются с помощью тега <a> (anchor).

Параграф 2: Обязательный атрибут href указывает адрес, на который ведет ссылка.

Параграф 3: Ссылки могут вести на другие страницы, файлы или якоря внутри документа.

Вариант 13.

Заголовок: Изображения на веб-странице

Параграф 1: Для вставки изображений используется одиночный тег .

Параграф 2: Атрибут src задает путь к файлу изображения.

Параграф 3: Атрибут alt содержит альтернативный текст, который отображается, если картинка не загрузилась.

Вариант 14.

Заголовок: Списки в HTML

Параграф 1: Неупорядоченный список создается тегом , а пункты в нем – .

Параграф 2: Упорядоченный список создается тегом , а пункты также – .

Параграф 3: Списки используются для навигации, перечисления свойств и т.д.

Вариант 15.

Заголовок: Валидация HTML-кода

Параграф 1: Валидация – это проверка кода на соответствие стандартам.

Параграф 2: Валидный код гарантирует кроссбраузерность и доступность.

Параграф 3: Для проверки можно использовать онлайн-сервисы, например, от W3C.

Лабораторная работа № 2

«Форматирование текста и создание гиперссылок»

Цель работы: приобретение практических навыков структурирования и оформления текстового контента на веб-странице с помощью языка HTML.

Методические указания

Язык разметки HTML предоставляет набор тегов для определения структуры и смысла контента веб-страницы. При оформлении текстовой информации, такой как статья, важно разделять задачи: HTML отвечает за структуру и логическое выделение, а визуальное оформление (цвет, шрифт, размер) – это область CSS.

Заголовки используются для организации контента на странице, создания иерархии разделов. В HTML существует шесть уровней заголовков, от самого важного <h1> до наименее важного <h6>. Тег <h1> должен использоваться один раз на странице и, как правило, содержит её основную тему. Последующие заголовки (<h2>, <h3> и т.д.) служат для обозначения подразделов (Рис. 2.1).

Заголовок Н1

Заголовок Н2

Заголовок Н3

Заголовок Н4

Заголовок Н5

Заголовок Н6

Рисунок 2.1 – Реализация заголовков.

Логические теги описывают назначение текста, а не его внешний вид.

**** - важность или срочность текста. По умолчанию браузеры отображают его **жирным** шрифтом.

**** - логическое ударение или интонационный акцент. По умолчанию отображается *курсивом*.

<cite> - используется для выделения названия. По умолчанию *курсив*.

<mark> - выделяет текст как релевантный в определенном контексте (например, как результат поиска). По умолчанию желтый фон.

Физические теги напрямую задают внешний вид текста, не неся явной семантической нагрузки.

**** - **жирный** текст, без дополнительной важности (например, ключевые слова в аннотации).

<i> - *курсив*, без акцента (например, технический термин, иностранное слово).

<u> - подчеркивает текст.

Тег <a> создает гиперссылку, связывающую текущий документ с другим ресурсом или частью текущего документа. Самый важный атрибут ссылки – href (hypertext reference), который содержит URL-адрес цели.

Внешняя ссылка ведет на страницу другого сайта. Обязательно должен включать протокол (обычно https://).

```
<a href="https://www.example.com">Посетите Example</a>
```

Внутренняя ссылка осуществляют навигацию между страницами в пределах одного сайта.

```
<a href="web/index.html">Главная страница</a>
```

Внутридокументная (якорная) ссылка ссылается на конкретный элемент внутри текущей страницы. Для этого элементу, на который нужно сослаться, присваивается уникальный id (идентификатор). В href ставится символ #, за которым следует значение этого id.

```
<h2 id="chapter2">Глава вторая</h2>
```

```
<a href="#chapter2">Перейти ко второй главе</a>
```

Атрибут title добавляет всплывающую подсказку при наведении курсора на ссылку, поясняя её содержание или цель.

```
<a href="#chapter2" title="Глава вторая">Перейти ко второй главе</a>
```

Задания для выполнения лабораторной работы

Создайте HTML-страницу, представляющую собой статью на заданную по вариантам тему. Статья должна содержать:

1. Заголовок 1-го уровня.
2. Не менее трех разделов, каждый с заголовком 2-го уровня.
3. Одному из разделов присвойте id для якорной ссылки.
4. Одну якорную ссылку на созданный раздел.
5. Абзацы текста, в которых необходимо использовать минимум по одному разу каждый тег:

Логическое выделение: , , <mark>, <cite>, <code>

Физическое выделение: , <i>

6. Создайте внешнюю гиперссылку на официальный или авторитетный интернет-ресурс, релевантный теме статьи. Ссылка должна открываться в новой вкладке.

Вариант 1. Что такое Интернет и как он работает.

Вариант 2. Зачем нужен Python: примеры использования.

Вариант 3. Как защитить свои аккаунты в интернете.

Вариант 4. Искусственный интеллект в нашей жизни.

Вариант 5. Плюсы и минусы социальных сетей.

Вариант 6. Основы правильного питания.

Вариант 7. Как люди изучают космос: от телескопов до МКС.

Вариант 8. Почему графика в играх стала такой реалистичной.

Вариант 9. Как получают электричество из природы и виды энергии

Вариант 10. Как студенту управлять своими деньгами.

Вариант 11. Открытия, которые изменили мир: пенициллин, электричество, интернет.

Вариант 12. Основные жанры литературы.

Вариант 13. Основные виды архитектур.

Вариант 14. Современные направления в музыке.

Вариант 15. Правила композиции для хорошей фотографии.

Лабораторная работа № 3

«Встраивание мультимедийного контента и создание списков»

Цель работы: освоение практических приёмов по встраиванию графических изображений на веб-страницу с соблюдением принципов доступности и созданию структурированных списков различных типов.

Методические указания

Для передачи информации, навигации и улучшения пользовательского опыта используются изображения и списки. В HTML для этих целей существуют специальные элементы.

Встраивание изображений

**Тег ** является одиночным тегом и используется для вставки графики на страницу. Ключевая его особенность – он не оборачивает контент, а загружает внешний ресурс. Важнейшими атрибутами этого тега являются src и alt.

Атрибут src задаёт путь к файлу изображения. Путь может быть относительным (относительно местоположения текущего HTML-документа, images/photo.jpg) или абсолютным (полный URL-адрес в интернете, https://site.com/photo.jpg).

```

```

Атрибут alt - это обязательное текстовое описание изображения. Если изображение не загрузилось, браузер покажет этот текст на его месте.

```

```

Дополнительные атрибуты :

1. **width и height** задают ширину и высоту изображения в пикселях. Рекомендуется их указывать, чтобы браузер зарезервировал место под картинку до её загрузки, предотвращая резкие смещения контента на странице.

2. **title** задаёт всплывающую подсказку, которая появляется при наведении курсора мыши на изображение.

Создание списков

Списки в HTML используются для группировки связанных пунктов. Существует два основных типа списков, каждый из которых имеет свою семантику и применение.

**Маркированный список ** используется для перечисления элементов, порядок которых не важен.

Тег - это контейнер для всего списка.

Тег - обозначает каждый отдельный пункт списка.

```
<ul>
  <li>Главная</li>
  <li>О компании</li>
  <li>Услуги</li>
  <li>Контакты</li>
</ul>
```

**Нумерованный список ** используется для перечисления элементов, где важен порядок или последовательность.

Тег - контейнер для нумерованного списка.

Тег - обозначает каждый пункт, который автоматически нумеруется.

Атрибуты :

type: Меняет стиль нумерации (1 – цифры, A – заглавные буквы, a – строчные буквы, I – большие римские цифры, i – маленькие римские цифры).

start: Задает номер, с которого начнётся отсчет.

Вложенность списков

Любой список может содержать внутри себя другой список. Это используется для создания сложных многоуровневых структур.

```
<ol>
  <li>Введение
    <ul>
      <li>Актуальность темы</li>
      <li>Цели исследования</li>
    </ul>
  </li>
  <li>Основная часть
    <ol type="a">
      <li>Теоретические основы</li>
      <li>Практическое исследование</li>
    </ol>
  </li>
</ol>
```

```
</li>
<li>Заключение</li>
</ol>
```

Задания для выполнения лабораторной работы

Создайте HTML-страницу, представляющую собой информационную страницу на заданную по вариантам тему. Страница должна содержать:

1. Заголовок 1-го уровня.
2. Не менее одного изображения, с корректно заполненными обязательными атрибутами.
3. Маркированный список, содержащий не менее 4 пунктов.
4. Нумерованный список, содержащий не менее 5 пунктов. Для списка задайте атрибут type со значением, отличным от «1».
5. Сделайте один из пунктов в маркированном или нумерованном списке вложенным.

Вариант 1. Тема: Как правильно посадить комнатное растение.

Вариант 2. Тема: Топ-5 достопримечательностей моего родного города.

Вариант 3. Тема: Сборка игрового ПК: основные компоненты.

Вариант 4. Тема: Утренняя зарядка для бодрости (комплекс упражнений).

Вариант 5. Тема: Подготовка к походу в горы.

Вариант 6. Тема: Правила ухода за аквариумными рыбками.

Вариант 7. Тема: Что взять с собой на пикник: список вещей и продуктов.

Вариант 8. Тема: Что должно быть в домашней аптечке: минимальный набор.

Вариант 9. Тема: Как получают электричество из природы и виды энергии

Вариант 10. Тема: Основные жесты и команды для дрессировки щенка.

Вариант 11. Тема: Основные этапы мойки автомобиля своими руками.

Вариант 12. Тема: Как почистить клавиатуру компьютера от пыли и мусора.

Вариант 13. Тема: Как собрать аптечку в дорогу на море.

Вариант 14. Тема: Как вести себя при пожаре в многоэтажном доме

Вариант 15. Тема: Как найти Полярную звезду на ночном небе

Лабораторная работа № 4

«Построение таблицы данных»

Цель работы: освоение принципов семантического и структурного построения таблиц в HTML для отображения табличных данных.

Методические указания

Таблица в HTML – это структура для представления данных, организованных в строки и столбцы. Для создания таблиц используется семейство связанных тегов, каждый из которых описывает свою часть таблицы.

Основная структура таблицы

Любая таблица начинается с тега **<table>**, который является контейнером для всех остальных элементов. Внутри него информация организуется построчно с помощью тега **<tr>**. Каждая строка **<tr>** содержит ячейки.

Существует два типа ячеек:

1. **<td>** - Стандартная ячейка, содержащая фактические данные.
2. **<th>** - Ячейка-заголовок для столбца или строки. По умолчанию текст в **<th>** выделяется жирным и выравнивается по центру.

```
<table>
  <tr> <th>Заголовок 1</th> <th>Заголовок 2</th> </tr> <!-- Строка
заголовков -->
  <tr> <td>Данные A1</td> <td>Данные A2</td> </tr> <!-- Строка
данных -->
  <tr> <td>Данные B1</td> <td>Данные B2</td> </tr>
</table>
```

Секционирование таблицы

Для логического структурирования больших таблиц и независимого стилизования их частей используются специальные теги-секции:

1. **<thead>** - определяет шапку таблицы. Внутри размещаются строки (**<tr>**), содержащие заголовки столбцов (**<th>**). В таблице может быть только один **<thead>**.

2. **<tbody>** - определяет тело таблицы. Внутри размещаются основные строки с данными (<td>). В таблице может быть несколько <tbody>, что удобно для логической группировки данных.

3. **<tfoot>** - определяет подвал таблицы. Обычно содержит итоговые строки (суммы, средние значения). Согласно спецификации HTML, тег <tfoot> в разметке может идти сразу после <thead>, даже перед <tbody>, но браузер всегда отобразит его внизу таблицы. Это позволяет браузеру начать рендеринг итогов еще до загрузки всех данных.

Объединение ячеек

Для создания сложных таблиц, где одна ячейка должна занимать место нескольких, используются атрибуты **colspan** и **rowspan**. Они применяются к тегам <td> или <th>.

colspan - указывает, сколько столбцов по горизонтали должна занимать ячейка. Значение – целое число (colspan="2").

rowspan - указывает, сколько строк по вертикали должна занимать ячейка. Значение – целое число (rowspan="3").

При объединении ячеек, ячейки <td> или <th> в последующих строках просто пропускаются.

```
<table border="1">
  <tr>
    <th rowspan="2">Регион</th> //Эта ячейка займет 2 строки
    <th colspan="2">Продажи</th> //Эта ячейка займет 2 столбца
  </tr>
  <tr>
    //В этой строке ячейка для "Регион" уже занята rowspan'ом
    сверху
    <th>План</th> <th>Факт</th> //Это подзаголовки под "Продажи"
  </tr>
  <tr>
    <td>Центральный</td> <td>100</td> <td>120</td>
  </tr>
</table>
```

Задания для выполнения лабораторной работы

Создайте HTML-страницу, содержащую сложную таблицу по вариантам.

Вариант 1. «Расписание экзаменов группы АИБ-01-23»

№	Дисциплина	Время проведения		Ответственные	
		Дата	Час	Аудитория	Преподаватель
1	Базы данных	25.12.26	10:00	А310	Иванов А.С.
2	Веб-разработка	26.12.26	14:00	Б415	Петров В.В.
3	Алгоритмизация	27.12.26	9:00	Г210	Сидоров А.И.
Консультации за день до экзамена					

Вариант 2. «Турнирная таблица»

Место	Команда	Статистика игр					Очки
		И	В	Н	П	Мячи	
1	Спартак	5	4	1	0	12-3	13
2	ЦСКА	5	3	2	0	10-4	11
3	Динамо	5	2	1	2	8-7	7
Лидер: Спартак							

Вариант 3. «Учет расходов»

Категория	Статья расходов	Финансы, руб.		
		План	Факт	Отклонение
Питание	Продукты	8000	8500	+500
	Кафе	3000	3500	+500
Транспорт	Метро	2000	2000	0
	Такси	1000	1500	+500
Всего расходов		14000	15500	+1500

Вариант 4. «Характеристики ноутбуков»

Модель	Ключевые характеристики				Цена, руб.
	процессор	ОЗУ	Накопитель	Экран	
Asus VivoBook	i5-12450H	16GB	512GB SSD	15.6	65 000

HP Laptop	Ryzen 5	8GB	256GB SSD	14	45 000
Lenovo IdeaPad	i3-1215U	8GB	512GB SSD	15.6	50 000
Рекомендовано: Asus VivoBook					

Вариант 5. «Производственный календарь»

Пн	Вт	Ср	Чт	Пт	Сб	Вс
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	Выходных: 8		
Рабочих дней: 22						

Вариант 6. «Учет рабочего времени»

Сотрудник	Рабочие часы по дням					Итого
	Пн	Вт	Ср	Чт	Пт	
Иванов И.И.	8	8	7	8	8	39
Петрова А.С.	8	8	8	4	8	36
Сидоров П.С.	8	6	8	8	8	38
Всего часов по отделу						113

Вариант 7. «Меню ресторана»

Позиция	Порция	Цена, руб.
Холодные закуски		
Селедка под шубой	150 г	350
Салат Цезарь	200 г	450
Основные блюда		
Стейк	300 г	1200
Карбонара	350 г	600

Средний чек	650
-------------	-----

Вариант 8. «План конференции»

Этап	Задача	Ответственный	Срок	Статус
Подготовка	Разослать приглашения	Иванов И.И.	01.11	Выполнено
	Заказать помещение	Петрова А.С	05.11	Выполнено
	Напечатать программы	Сидоров В.В.	10.11	Не выполнено
Общий прогресс				67%

Вариант 9. «Смета ремонта»

№	Наименование	Калькуляция		
		Ед.	Кол-во	Цена, руб.
1	Краска интерьерная	л	5	600
2	Грунтовка	л	3	300
3	Работа маляра	м ²	50	200
Итого материалов				3 900
Итого работ				10 000
Общая стоимость				13 900

Вариант 10. «Успеваемость 9 «А» класса»

Ученик	Предметы				Средний балл
	Алгебра	Физика	Химия	Литература	
Иванов	5	4	4	5	4.5
Петрова	5	5	5	5	5
Сидоров	3	4	4	3	3.5
Лучший балл: Петрова					

Вариант 11. «Прогноз погоды»

Город	Дни недели						
	Пн	Вт	Ср	Чт	Пт	Сб	Вс
Москва	+5	+3	+1	+4	+6	+7	+8
	Облачно	Снег	Снег	Облачно	Ясно	Ясно	Ясно
Сочи	+15	+16	+17	+16	+18	+19	+20
	Дождь	Дождь	Облачно	Ясно	Ясно	Ясно	Ясно

Вариант 12. «График дежурств»

Неделя	Пара	Дни недели				
		Пн	Вт	Ср	Чт	Пт
1	1	Иванов	Петров	Сидоров	Козлов	Федоров
	2	Петров	Сидоров	Козлов	Федоров	Иванов
	3	Сидоров	Козлов	Федоров	Иванов	Петров

Вариант 13. «План отпусков»

Сотрудник	Месяц						
	Янв	Фев	Мар	Апр	Май	Июнь	Июль
Иванов				Отпуск			
Петров			Отпуск				
Сидоров	Отпуск						
Пик отпусков: апрель-май							

Вариант 14. «Характеристики автомобилей»

Модель	Технические параметры				Цена, руб.
	Двигатель	КПП	Привод	Расход топлива на 100 км	
Kia Sportage	2.0 л/150 л.с.	Автомат	Передний	8.2 л	2 500 000

Toyota RAV4	2.5 л/203 л.с.	Вариатор	Полный	9.0 л	3 200 000
Nissan X-Trail	1.5 л/204 л.с.	Робот	Полный	7.5 л	2 800 000

Вариант 15. «Тарифы мобильной связи»

Оператор	Тарифные параметры			Цена, руб/мес
	Минуты	Интернет, ГБ	SMS	
МТС	500	20	100	500
Билайн	2000	30	50	600
Мегафон	300	15	200	450
Средняя стоимость				517

Лабораторная работа № 5

«Разработка HTML-формы для сбора данных»

Цель работы: освоение принципов создания интерактивных HTML-форм для взаимодействия с пользователем.

Методические указания

HTML-формы - это основной инструмент для сбора информации от пользователя на веб-странице: от простого поискового запроса до сложной регистрационной анкеты. Все элементы формы размещаются внутри тега `<form>`, который определяет область и настраивает механизм отправки данных на сервер.

Тег `<form>` и его атрибуты

Тег `<form>` является контейнером для всех элементов формы. Два ключевых атрибута определяют, куда и как отправляются данные:

```
<form action="#" method="post">  
  //Элементы формы здесь  
</form>
```

1. `action` - задает URL-адрес сервера или страницы, которая будет обрабатывать данные формы. Для учебных целей можно использовать значение `#`, что означает отправку на ту же страницу.

2. `method` - определяет метод HTTP-запроса для отправки данных. Основные значения: `get` - данные добавляются прямо в адресную строку. Подходит для поиска и фильтров. `Post` - данные уходят «внутри» запроса, и в адресе их не видно. Используется для форм регистрации, входа и других форм с личной информацией.

Элемент `<label>`: Текстовое описание для элемента формы.

Простейшая форма может быть такой:

```
<form action="/submit" method="post">  
  <label for="name">Ваше имя:</label>  
  <input type="text" id="name" name="name" required>  
  <label for="email">Email:</label>  
  <input type="email" id="email" name="email" required>  
  <button type="submit">Отправить</button>  
</form>
```

Элементы ввода и их атрибуты

Главное в форме - элементы, принимающие данные от пользователя. У каждого элемента есть ключевой атрибут `name`, который служит идентификатором пары «ключ-значение» при отправке данных на сервер.

Поля ввода **<input>** - основной элемент формы. С его помощью пользователь вводит имя, пароль, email и другие данные.

Текстовое поле `<input type="text">`. Стандартное поле для ввода одной строки текста (имя, email).

```
<input type="text" name="username" id="username"
placeholder="Введите имя" value="Иван" maxlength="50" required>
```

Атрибуты: `placeholder` – подсказка внутри поля, `value` – предзаполненное значение, `maxlength` – максимальное количество символов, `required` – обязательное поле.

Поле с паролем `<input type="password">`. Аналогично текстовому, но вводимые символы маскируются.

```
<input type="password" name="user_password" id="user_password"
required>
```

Числовое поле `<input type="number">`. Позволяет вводить только числа.

```
<input type="number" name="age" id="age" min="0" max="120"
step="1">
```

Атрибуты: `min` и `max` задают диапазон, `step` – шаг изменения, например, 0.5 для дробных чисел.

Поле для email `<input type="email">`. Предназначено для ввода адреса электронной почты.

```
<input type="email" name="email" id="email">
```

Чекбокс `<input type="checkbox">`. Позволяет пользователю выбрать ноль, один или несколько вариантов.

```
<input type="checkbox" name="agree" id="agree">
<label for="agree">Согласен с условиями</label>
```

Радиокнопка `<input type="radio">`. Позволяет выбрать только один вариант из группы.

```
<input type="radio" name="gender" value="male" id="male">
<label for="male">Мужской</label>
```

```
<input type="radio" name="gender" value="female" id="male">
<label for="male">Женский</label>
```

Все переключатели одной группы должны иметь одинаковое name, но разные value

Многострочное поле <textarea>. Для текста большого объема: комментариев, отзывов и тому подобного.

```
<textarea name="message" rows="5" cols="40" placeholder="Ваше
сообщение"></textarea>
```

Кнопки <button>. Управляют отправкой или очисткой формы.

```
<button type="submit">Отправить форму</button> //Кнопка отправки
<button type="reset">Очистить</button> //Кнопка сброса
<button type="button" onclick="doSomething()">Выполнить
действие</button> //Обычная кнопка для JavaScript
```

Раскрывающийся список <select>. Позволяет выбрать вариант из меню.

```
<select name="country">
  <option value="">Выберите страну</option>
  <option value="ru">Россия</option>
  <option value="kz">Казахстан</option>
  <option value="by">Беларусь</option>
</select>
```

Группировка полей

Для логической организации сложных форм используются теги <fieldset> и <legend>.

```
<fieldset>
  <legend>Личная информация</legend>
  <label for="firstName">Имя:</label>
  <input type="text" id="firstName" name="firstName">

  <label for="lastName">Фамилия:</label>
  <input type="text" id="lastName" name="lastName">
</fieldset>
```

<fieldset> - группирует связанные элементы, рисуя вокруг них рамку.

<legend> - определяет заголовок группы. Размещается первым внутри <fieldset>

Задания для выполнения лабораторной работы

Вариант 1. Создайте HTML-страницу, содержащую форму. Форма должна иметь заголовок «Форма заявки на кредит». Создайте форму с 3 группами полей:

1. Группа «Личные данные»: Текстовое поле «ФИО», email, текстовое поле «Номер паспорта».
2. Группа «Параметры кредита»: Числовое поле «Сумма кредита» (min=10000, max=5000000), список вариантов «Срок кредита» (1 год, 3 года, 5 лет), радиокнопки «Цель кредита» (ипотека, авто, рефинансирование).
3. Группа «Дополнительно»: Многострочное поле «Комментарий», чекбокс «Согласен на рассылку», чекбокс «Имею постоянную работу».

Вариант 2. Создайте HTML-страницу, содержащую форму. Форма должна иметь заголовок «Анкета для нового сотрудника». Создайте форму с 3 группами полей:

1. Группа «Основная информация»: Текстовое поле «ФИО», email, числовое поле "Возраст" (min=18, max=70).
2. Группа «Профессиональные данные»: Список вариантов «Должность» (разработчик, тестировщик, менеджер), многострочное поле «Опыт работы», чекбоксы «Знание языков» (английский, немецкий, китайский).
3. Группа «Условия работы»: Радиокнопки «График работы» (полный день, удаленно, гибкий), многострочное поле «Пожелания к работе», чекбокс «Ознакомлен с политикой компании».

Вариант 3. Создайте HTML-страницу, содержащую форму. Форма должна иметь заголовок «Бронирование столика в ресторане». Создайте форму с 3 группами полей:

1. Группа «Контактные данные»: Текстовое поле «Имя», email, текстовое поле «Телефон».

2. Группа «Параметры бронирования»: Текстовое поле «Дата посещения», список вариантов «Время» (18:00, 19:00, 20:00, 21:00), числовое поле «Количество гостей» (min=1, max=10).

3. Группа «Дополнительно»: Радиокнопки «Тип столика» (у окна, в центре зала, VIP), многострочное поле «Особые пожелания», чекбокс «Подтверждаю бронирование».

Вариант 4. Создайте HTML-страницу, содержащую форму. Форма должна иметь заголовок «Заявка на ремонт техники». Создайте форму с 3 группами полей:

1. Группа «Данные клиента»: Текстовое поле «ФИО», email, текстовое поле «Телефон».

2. Группа «Информация об устройстве»: Список вариантов «Тип устройства» (ноутбук, смартфон, холодильник), текстовое поле «Модель», многострочное поле «Описание проблемы».

3. Группа «Условия ремонта»: Радиокнопки «Срочность» (обычная, срочно), чекбокс «Нужен выезд мастера», чекбокс «Согласен с диагностикой».

Вариант 5. Создайте HTML-страницу, содержащую форму. Форма должна иметь заголовок «Регистрация на онлайн-курс». Создайте форму с 3 группами полей:

1. Группа «Личные данные»: Текстовое поле «Имя», email, числовое поле «Возраст».

2. Группа «Информация о курсе»: Список вариантов «Выберите курс» (Веб-разработка, Data Science, Дизайн), радиокнопки «Формат обучения» (утро, день, вечер), чекбокс «Нужен сертификат».

3. Группа «Дополнительно»: Многострочное поле «Опыт в теме курса», чекбокс-группа «Источник информации» (соц. сети, друзья, поиск), чекбокс «Принимаю условия».

Вариант 6. Создайте HTML-страницу, содержащую форму. Форма должна иметь заголовок «Форма обратной связи». Создайте форму с 3 группами полей:

1. Группа «Контактные данные»: Текстовое поле «Имя», email, текстовое поле «Тема обращения».
2. Группа «Содержание обращения»: Список вариантов «Тип обращения» (жалоба, предложение, вопрос), многострочное поле «Текст обращения», чекбокс «Нужен сертификат».
3. Группа «Оценка»: Радиокнопки «Оцените наш сервис» (1-5 баллов), чекбокс «Хочу получить ответ на email», чекбокс «Обращение конфиденциально».

Вариант 7. Создайте HTML-страницу, содержащую форму. Форма должна иметь заголовок «Регистрация на спортивное мероприятие». Создайте форму с 3 группами полей:

1. Группа «Участник»: Текстовое поле «ФИО», числовое поле «Возраст» (min=18, max=70), список вариантов «Вид спорта» (бег, плавание, велоспорт).
2. Группа «Данные для участия»: Радиокнопки «Дистанция» (5 км, 10 км, 21 км), чекбокс-группа «Дополнительные услуги» (футболки, медаль, фото).
3. Группа «Медицинская информация»: Многострочное поле «Медицинские противопоказания», чекбокс «Имею допуск врача», чекбокс «Согласен с правилами».

Вариант 8. Создайте HTML-страницу, содержащую форму. Форма должна иметь заголовок «Подбор тура в путешествие». Создайте форму с 3 группами полей:

1. Группа «Параметры тура»: Список вариантов «Страна назначения» (Россия, Турция, Египет), числовое поле «Количество человек», чекбокс-группа «Тип отдыха» (пляжный, экскурсионный, активный).

2. Группа «Даты и бюджет»: Текстовое поле «Дата вылета», числовое поле «Бюджет на человека», радиокнопки «Категория отеля» (3*, 4*, 5*).

3. Группа «Пожелания»: Многострочное поле «Особые требования», чекбокс «Рассмотреть все варианты», чекбокс «Отправить предложения на email».

Вариант 9. Создайте HTML-страницу, содержащую форму. Форма должна иметь заголовок «Заказ доставки еды». Создайте форму с 3 группами полей:

1. Группа «Данные для доставки»: Текстовое поле «Имя», текстовое поле «Адрес», текстовое поле «Телефон».

2. Группа «Заказ»: Список вариантов «Блюдо 1», список вариантов «Блюдо 2», числовое поле «Количество приборов».

3. Группа «Дополнительно»: Радиокнопки «Способ оплаты» (карта, наличные), многострочное поле «Комментарий для курьера», чекбокс «Оставить у двери».

Вариант 10. Создайте HTML-страницу, содержащую форму. Форма должна иметь заголовок «Запись на прием к врачу». Создайте форму с 3 группами полей:

1. Группа «Пациент»: Текстовое поле «ФИО», числовое поле «Возраст», текстовое поле «Номер полиса».

2. Группа «Параметры приема»: Список вариантов «Специализация врача» (Терапевт, Офтальмолог, Дерматолог), текстовое поле «Дата приема», радиокнопки «Время» (утро, день, вечер).

3. Группа «Симптомы»: Многострочное поле «Описание симптомов», чекбокс «Срочный прием», чекбокс «Согласен на обработку данных».

Вариант 11. Создайте HTML-страницу, содержащую форму. Форма должна иметь заголовок «Форма жалобы в ЖКХ». Создайте форму с 3 группами полей:

1. Группа «Заявитель»: Текстовое поле «ФИО», текстовое поле «Адрес», текстовое поле «Телефон».

2. Группа «Суть проблемы»: Список вариантов «Тип проблемы» (протечка, отопление, электричество), многострочное поле «Подробное описание».

3. Группа «Дополнительно»: Радиокнопки «Срочность» (не срочно, срочно), чекбокс «Требуется обратный звонок», чекбокс «Данные достоверны».

Вариант 12. Создайте HTML-страницу, содержащую форму. Форма должна иметь заголовок «Заявка на консультацию юриста». Создайте форму с 3 группами полей:

1. Группа «Клиент»: Текстовое поле «ФИО», email, текстовое поле «Телефон».

2. Группа «Вопрос»: Список вариантов «Тема вопроса» (семейное право, наследство, договоры), многострочное поле «Описание ситуации».

3. Группа «Условия»: Радиокнопки «Формат консультации» (онлайн, офис), чекбокс «Нужны документы», чекбокс «Согласен на конфиденциальность».

Вариант 13. Создайте HTML-страницу, содержащую форму. Форма должна иметь заголовок «Опрос о продукте». Создайте форму с 3 группами полей:

1. Группа «Респондент»: Текстовое поле «Имя», числовое поле «Возраст», список вариантов «Пол» (мужской, женский).
2. Группа «Использование продукта»: Радиокнопки «Частота использования» (ежедневно, еженедельно, редко), чекбокс-группа «Что нравится» (качество, цена, дизайн).
3. Группа «Предложения»: Многострочное поле «Что улучшить?», чекбокс «Готов участвовать в тестировании», чекбокс «Разрешаю отправку уведомлений».

Вариант 14. Создайте HTML-страницу, содержащую форму. Форма должна иметь заголовок «Заявка на размещение рекламы». Создайте форму с 3 группами полей:

1. Группа «Рекламодатель»: Текстовое поле «Название компании», email, текстовое поле «Контактное лицо».
2. Группа «Параметры рекламы»: Список вариантов «Тип рекламы» (баннер, статья, видео), числовое поле «Бюджет» радиокнопки «Срок размещения» (месяц, квартал, год).
3. Группа «Дополнительно»: Многострочное поле «Пожелания к размещению», чекбокс «Нужен медиаплан», чекбокс «Согласен с условиями».

Вариант 15. Создайте HTML-страницу, содержащую форму. Форма должна иметь заголовок «Форма для волонтеров». Создайте форму с 3 группами полей:

1. Группа «Волонтер»: Текстовое поле «ФИО», email, текстовое поле «Телефон».
2. Группа «Направления работы»: Список вариантов «Направление» (помощь детям, помощь животным, экология), чекбокс-группа «Навыки» (водительские права, медицинские знания, языки).

3. Группа «График»: Радиокнопки «Доступность» (выходные, будни, любое время), многострочное поле «Опыт волонтерства», чекбокс «Готов пройти обучение».

Лабораторная работа № 6

«Применение семантических тегов HTML5 для разметки страницы»

Цель работы: освоение принципов семантической верстки с использованием HTML5.

Методические указания

HTML5 - язык для структурирования и представления содержимого всемирной паутины, который является пятой версией HTML. До появления HTML5 разработчики использовали теги `<div>` с классами (`<div class="header">`, `<div class="footer">`) для обозначения частей страницы. HTML5 представил набор специальных семантических тегов, которые делают структуру документа понятной не только людям, но и браузерам, поисковым системам и вспомогательным технологиям.

Ниже представлен некоторые теги HTML5:

Тег `<header>` – определяет «шапку» сайта или раздела. Обычно содержит логотип, заголовок, навигационное меню, поисковую форму.

Тег `<nav>` - предназначен для основной навигации по сайту. Содержит блок ссылок, ведущих к другим разделам сайта или страницам.

Тег `<main>` – этот элемент предназначен для основного содержимого документа.

Тег `<article>` – определяет статью.

Тег `<section>` – задаёт раздел документа, может применяться для блока новостей, контактной информации, вкладок в диалоговом окне и т.д.

Тег `<footer>` – определяет низ сайта или раздела.

Создание макета страницы

Шапка содержит вводную информацию о странице или разделе. На уровне всей страницы это обычно логотип, название сайта и краткое описание.

```
<header>  
  <h1>Название сайта или компании</h1>  
  <p>Краткий слоган или описание деятельности</p>  
</header>
```

Навигационный блок содержит главное меню сайта. Рекомендуется оформлять меню как список ссылок.

```
<nav>
  <a href="#" style="margin-right: 15px;">Главная</a>
  <a href="#" style="margin-right: 15px;">О нас</a>
  <a href="#" style="margin-right: 15px;">Услуги</a>
  <a href="#">Контакты</a>
```

```
</nav>
```

style="..." – атрибут для задания стилей прямо в HTML.

style="margin-right: 15px;" – создает отступы между текстом.

Основной контент находится в теге <main>. Для создания двух колонок мы разместим внутри <main> две основные семантические единицы: <article> (для основной статьи) и <aside> (для боковой панели). Чтобы они стали колонками, мы применим к ним встроенные стили, задав им свойство display: inline-block; и ширину (width). inline-block позволяет блочным элементам располагаться в строку, как слова в тексте.

```
<main style="padding: 20px;">
  // Основная статья - левая колонка
  <article style="width: 70%; display: inline-block; padding-
right: 20px;">
    <h2>Заголовок статьи</h2>
    <p>Основной текст статьи. Здесь размещается уникальный
контент страницы: новость, описание, аналитика и т.д. Этот блок занимает
около 70% ширины контейнера.</p>
    <section>
      <h3>Подраздел статьи</h3>
      <p>Текст подраздела, который логически связан с
основной статьей.</p>
    </section>
  </article>
  // Боковая панель - правая колонка
  <aside style="width: 25%; display: inline-block; background-
color: #f4f4f4; padding: 15px;">
    <h3>Боковая панель</h3>
    <p>Дополнительная информация, связанная с темой сайта:
архив, рекламный баннер, популярные посты.</p>
  </aside>
</main>
```

display: inline-block; – ключевое свойство для создания колонок. Оно заставляет блочные элементы (<article>, <aside>) вести себя как строчные, позволяя им встать рядом.

Подвал – нижняя часть страницы. Внутри обычно размещается информация об авторских правах и контакты (Рис. 6.1).

```
<footer style=" text-align: center; padding: 15px;">
  <p>&copy; 2023 Мой сайт. Все права защищены.</p>
  <p>Контактная информация: info@mysite.com</p>
</footer>
```

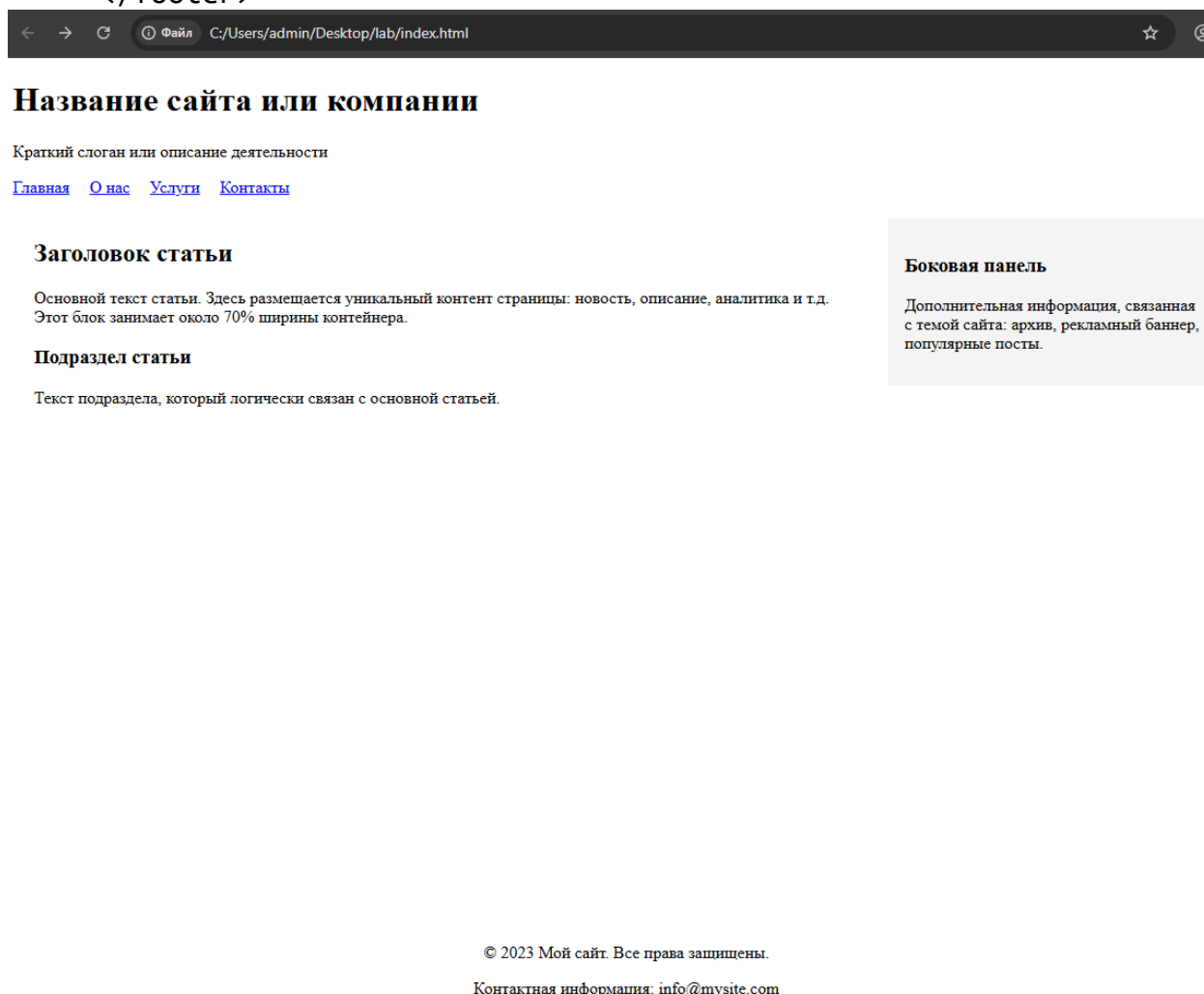


Рисунок 6.1 – Макет сайта.

Задания для выполнения лабораторной работы

Вариант 1. Создать разметку страницы согласно макету (шапка, основной блок с 2 колонками, подвал), используя семантические теги.

Сайт-портфолио веб-разработчика.

Шапка: «Портфолио [Ваше имя]».

Навигация: «Главная», «Мои работы», «Обо мне», «Блог», «Контакты».

Левая колонка: Заголовок «Последний проект: Корпоративный сайт».

Подробное описание проекта (технологии, задачи).

Правая колонка: Заголовок «Мои навыки». Список ключевых технологий HTML5.

Подвал: «© [Год] [Ваше имя]. Сайт-портфолио. Для связи: email@example.com».

Вариант 2. Создать разметку страницы согласно макету, используя семантические теги.

Блог о путешествиях.

Шапка: «Дневник путешествий».

Навигация: «Главная», «Страны», «Фотогалерея», «Советы», «Обратная связь».

Левая колонка: Заголовок «Неделя в горах Алтая». Рассказ о походе.

Правая колонка: Заголовок «Предстоящие поездки». Список из 2-3 планируемых направлений.

Подвал: «© Блог о путешествиях. Все истории основаны на реальных событиях. @travel_diary».

Вариант 3. Создать разметку страницы согласно макету, используя семантические теги.

Сайт кулинарных рецептов

Шапка: «Домашняя кухня».

Навигация: «Завтраки», «Супы», «Основные блюда», «Десерты», «Быстрое».

Левая колонка: Заголовок «Классический шоколадный торт». Пошаговый рецепт с ингредиентами.

Правая колонка: Заголовок «Лайфхак недели». Краткий совет (например, «Как взбить идеальные сливки»).

Подвал: «© Кулинарный сайт. Рецепты проверены лично. Приятного аппетита!».

Вариант 4. Создать разметку страницы согласно макету, используя семантические теги.

Новостной портал

Шапка: «IT News Daily».

Навигация: «Главные новости», «Гаджеты», «Игры», «Наука», «Интервью».

Левая колонка: Заголовок «Представлен новый процессор». Новость о технологическом анонсе.

Правая колонка: Заголовок «Самое популярное». Список из 3 заголовков самых читаемых статей.

Подвал: «© IT News Daily, 2023. Все права на материалы принадлежат изданию. info@itnews.ru».

Вариант 5. Создать разметку страницы согласно макету, используя семантические теги.

Сайт книжного клуба.

Шапка: «Книжный уголок».

Навигация: «Каталог», «Рецензии», «Авторы», «События», «Обсуждения».

Левая колонка: Заголовок «Обзор: «1984» Джорджа Оруэлла». Ваше мнение о книге.

Правая колонка: Заголовок «Книга месяца». Название и автор выбранной книги.

Подвал: «© Книжный клуб. Читаем и обсуждаем вместе. Все обзоры субъективны».

Вариант 6. Создать разметку страницы согласно макету, используя семантические теги.

Спортивный сайт.

Шапка: «Всё о футболе».

Навигация: «Чемпионаты», «Команды», «Статистика», «Трансферы», «Видео».

Левая колонка: Заголовок «Итоги прошедшего тура». Краткий обзор матчей.

Правая колонка: Заголовок «Турнирная таблица». Мини-таблица из 3-4 строк (место, команда, очки).

Подвал: «© Футбольный портал. Актуальные новости и аналитика. Поддержка: support@football.ru».

Вариант 7. Создать разметку страницы согласно макету, используя семантические теги.

Сайт образовательного центра.

Шапка: «IT Academy».

Навигация: «Курсы», «Преподаватели», «Расписание», «Отзывы», «Записаться».

Левая колонка: Заголовок «Новый курс: «Введение в Python»». Описание программы курса, для кого подходит.

Правая колонка: Заголовок «Бесплатный вебинар». Тема и дата ближайшего мероприятия.

Подвал: «© IT Academy. Лицензия на образовательную деятельность №XXXXX. Адрес: г. Город, ул. Примерная, 1».

Вариант 8. Создать разметку страницы согласно макету, используя семантические теги.

Личный дневник.

Шапка: «Мысли вслух».

Навигация: «Сегодня», «Архив», «Цитаты», «О проекте».

Левая колонка: Заголовок «[Сегодняшняя дата]». Запись в свободной форме о событиях дня или размышлениях.

Правая колонка: Заголовок «Настроение дня». Одно слово, описывающее день.

Подвал: «Личный блог. Здесь я делюсь своими мыслями.»

Вариант 9. Создать разметку страницы согласно макету, используя семантические теги.

Сайт о домашних растениях.

Шапка: «Мой Зелёный Мир».

Навигация: «Каталог растений», «Уход», «Проблемы и решения», «Форум», «Магазин».

Левая колонка: Заголовок «Как ухаживать за Монстерой». Подробная инструкция по поливу, свету, пересадке.

Правая колонка: Заголовок «Растение недели». Название и фото одного неприхотливого растения.

Подвал: «© Сайт для любителей комнатных растений. Зелёный вам настрой! Контакты для сотрудничества: plants@mail.com».

Вариант 10. Создать разметку страницы согласно макету, используя семантические теги.

Сайт музыкальной группы.

Шапка: «[Название группы] – Официальный сайт».

Навигация: «Новости», «Музыка», «Видео», «Гастроли», «Магазин».

Левая колонка: Заголовок «Анонс нового альбома». Текст о готовящемся к выпуску альбоме, возможное название.

Правая колонка: Заголовок «Ближайший концерт». Дата, город, название площадки.

Подвал: «© [Год] [Название группы]. Все права на музыку защищены. Booking: manager@band.com».

Вариант 11. Создать разметку страницы согласно макету, используя семантические теги.

Сайт городского мероприятия.

Шапка: «Фестиваль уличной еды «Вкусная улица»».

Навигация: «Программа», «Участники», «Карта», «Билеты», «Партнеры».

Левая колонка: Заголовок «Программа на главной сцене». Расписание выступлений

Правая колонка: Заголовок «Как добраться». Краткое описание места проведения и общественного транспорта.

Подвал: «© Организатор фестиваля «Вкусная улица». По вопросам участия: info@fest.ru».

Вариант 12. Создать разметку страницы согласно макету, используя семантические теги.

Сайт про здоровый образ жизни.

Шапка: «Путь к здоровью».

Навигация: «Питание», «Тренировки», «Ментальное здоровье», «Мотивация», «Рецепты».

Левая колонка: Заголовок «10 минут утренней зарядки для бодрости». Комплекс простых упражнений.

Правая колонка: Заголовок «Факт дня». Короткий факт о пользе воды или сна.

Подвал: «© Информационный ресурс о ЗОЖ. Проконсультируйтесь с врачом. Мы не даем медицинских советов».

Вариант 13. Создать разметку страницы согласно макету, используя семантические теги.

Сайт кинотеатра.

Шапка: «Киномир».

Навигация: «Афиша», «Скоро», «Цены», «Контакты», «Правила».

Левая колонка: Заголовок «Главная премьера недели: «Название фильма»». Описание сюжета, жанр, рейтинг.

Правая колонка: Заголовок «Сеансы на сегодня». Список времени 2-3 сеансов для этого фильма.

Подвал: «© Кинотеатр «Киномир». Адрес: ул. Киношная, 7. Телефон: +7 (XXX) XXX-XX-XX».

Вариант 14. Создать разметку страницы согласно макету, используя семантические теги.

Сайт ремонта квартир.

Шапка: «Уютный дом – ремонт под ключ».

Навигация: «Услуги», «Портфолио», «Цены», «Отзывы», «Рассчитать».

Левая колонка: Заголовок «Этапы капитального ремонта». Последовательность работ от черновой отделки до чистовой.

Правая колонка: Заголовок «Акция!». Условия акции.

Подвал: «© Строительная компания «Уютный дом». Работаем с 2010 года. Гарантия на работы 3 года. phone: +7...».

Вариант 15. Создать разметку страницы согласно макету, используя семантические теги.

Сайт погодного сервиса.

Шапка: «Уютный дом – ремонт под ключ».

Навигация: «Сегодня», «Завтра», «На неделю», «Карты», «Архив».

Левая колонка: Заголовок «Погода в [Ваш город] на сегодня». Температура, облачность, ветер, давление.

Правая колонка: Заголовок «Рекорды дня». Самая высокая и самая низкая температура, зафиксированная в этот день в истории наблюдений.

Подвал: «© Метеосервис. Данные обновляются каждую минуту. Источник данных: ...».

Лабораторная работа № 7

«Подключение CSS и применение базовых стилей к семантической странице»

Цель работы: освоить практические навыки подключения внешних каскадных таблиц стилей CSS к HTML-документу, научиться создавать и структурировать внешний файл стилей. Получить опыт применения базовых CSS-свойств для визуального оформления семантического макета страницы

Методические указания

Стиль – набор правил оформления и форматирования, который может быть применен к различным элементам страницы, является основным понятием CSS.

CSS (Cascading Style Sheets) – это язык, содержащий набор свойств для описания внешнего вида любых HTML документов.

Существует три вида таблиц стилей:

Внутренние таблицы стилей (Inline Style Sheets) – при помощи специального атрибута помещаются прямо в HTML теги.

Глобальные таблицы стилей (Global Style Sheets) – определяют стиль элементов во всем документе.

Связанные таблицы стилей (Linked Style Sheets) – могут быть использованы для нескольких документов сразу и хранятся во внешнем файле.

Существует четыре способа связывания документа и таблицы стилей:

Связывание – позволяет использовать одну таблицу стилей для форматирования многих страниц HTML.

Для того чтобы расположить описания стилей в отдельном файле, нужно создать обычный текстовый файл, описать с помощью языка CSS необходимые стили, разместить этот файл на Web-сервере, а в коде Web-страниц, которые будут использовать стили из этого файла, нужно будет сделать ссылку на него. Делается это с помощью тега <link>, располагающегося внутри тега <head>:

```
<link rel=stylesheet type="text/css" href="url">
```

Первые два атрибута этого тега требуются для того, чтобы сообщить браузеру, что на этой страничке используется CSS. Атрибут href содержит ссылку на файл css.

Внедрение – позволяет задавать все правила таблицы стилей непосредственно в самом документе.

Описание стилей располагается в коде Web-страницы, внутри тега HEAD, в теге <style>. В этом случае вы можете использовать эти стили для элементов, располагающихся в пределах html файла. Атрибут type="text/css" является обязательным и служит для указания браузеру использовать CSS.

```
<style type="text/css">
  body { color: black; }
</style>
```

Встраивание в теги документа – позволяет изменять форматирование конкретных элементов страницы.

Описание стиля располагается внутри тега элемента, который вы описываете с помощью атрибута style.

```
<p style="color: red;">Текст</p>
```

Импортирование – позволяет встраивать в документ таблицу стилей, расположенную на сервере.

В теге <STYLE> можно импортировать внешнюю таблицу стилей с помощью свойства @import таблицы стилей:

```
@import: url(mystyles.css);
```

Его следует задавать в начале стилевых блока или связываемой таблицы стилей перед заданием остальных правил. Значение свойства @import является URL файла таблицы стилей.

Правила каскадных таблиц состоят из селектора и определения.

CSS селектор указывает, к каким элементам применять объявления стилей. В качестве селектора могут выступать:

- элементы (имена тегов);
- класс;
- идентификатор;

- псевдоклассы и псевдоэлементы.

Свойства селектора – стиль элемента, для которого он определен.

Синтаксис: селектор {свойства}

```
p{
color: blue;
size: 12pt;
}
```

Все элементы p в документе будут синего цвета, размером 12pt.

Правила каскадных таблиц стилей, в которых в качестве селектора используются теги HTML, влияют на отображение всех элементов заданного типа в документе.

Также существует **универсальный селектор ***. Он выбирает все элементы на странице.

```
* {
margin: 0;
}
```

Ниже представлены базовые свойства CSS для работы с текстом:

```
color: #333; // задает цвет текста
font-family: 'Arial', sans-serif; // определяет шрифт
font-size: 16px; // устанавливает размер шрифта
text-align: center; // выравнивает текст
line-height: 1.5; // устанавливает высоту строки
letter-spacing: 1px; // задает расстояние между буквами
```

Свойства для работы с блочными элементами:

```
width: 300px; height: 200px; // задают ширину и высоту
margin: 10px 20px; // устанавливает внешние отступы
padding: 15px; // задает внутренние отступы
border: 1px solid #ddd; // создает границу
background-color: #f9f9f9; // устанавливает цвет фона
box-shadow: 0 2px 5px rgba(0,0,0,0.1); // добавляет тень
border-radius: 5px; // скругляет углы
display: flex; // определяет тип отображения элемента
```

Разделение HTML и CSS - основной принцип веб-разработки. Это делается для организации кода и упрощения навигации по проекту.

Внутри папки проекта создается два файла - HTML-документ и CSS-файл.

```
index.html // HTML-документ с семантической разметкой
styles.css // Внешний файл стилей
```

В HTML-документе необходимо сообщить браузеру, что мы используем внешний файл стилей. В секции <head> необходимо добавить <link> с указанием пути к файлу CSS (Рис 7.1).

```
<link rel="stylesheet" href="styles.css">
```

Пример файла CSS:

```
* {
    margin: 0;
    padding: 0; } // Сброс стандартных отступов
body {
    font-size: 16px;
    line-height: 1.6;
    color: #333;
    background-color: #f9f9f9; } // Стили для body
header {
    background-color: #667eea;
    color: white;
    padding: 40px 20px;
    text-align: center;
    border-radius: 10px;
    margin-bottom: 20px; } // Стили для header
nav {
    padding: 15px;
    border-radius: 5px;
    margin-bottom: 20px; }
nav a {
    color: black;
    text-decoration: none;
    font-weight: bold;
    padding: 5px 10px;
    transition: color 0.3s; } // Стили для navigation
main {
    background-color: white;
    padding: 30px;
    border-radius: 8px;
    box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
    margin-bottom: 20px;
    display: flex;
    gap: 20px; } // Стили для main
article {
    background: white;
    flex: 3; } // Стили для article
aside {
    padding: 20px;
    border-radius: 8px;
    background: #bdc3c7;
    margin-bottom: 20px;
    flex: 1; } // Стили для aside
```

```

footer {
  background-color: #2d3748;
  color: #cbd5e0;
  padding: 25px;
  text-align: center;
  position: fixed;
  bottom: 0;
  left: 0;
  right: 0;
  height: 50px; } // Стили для footer

```

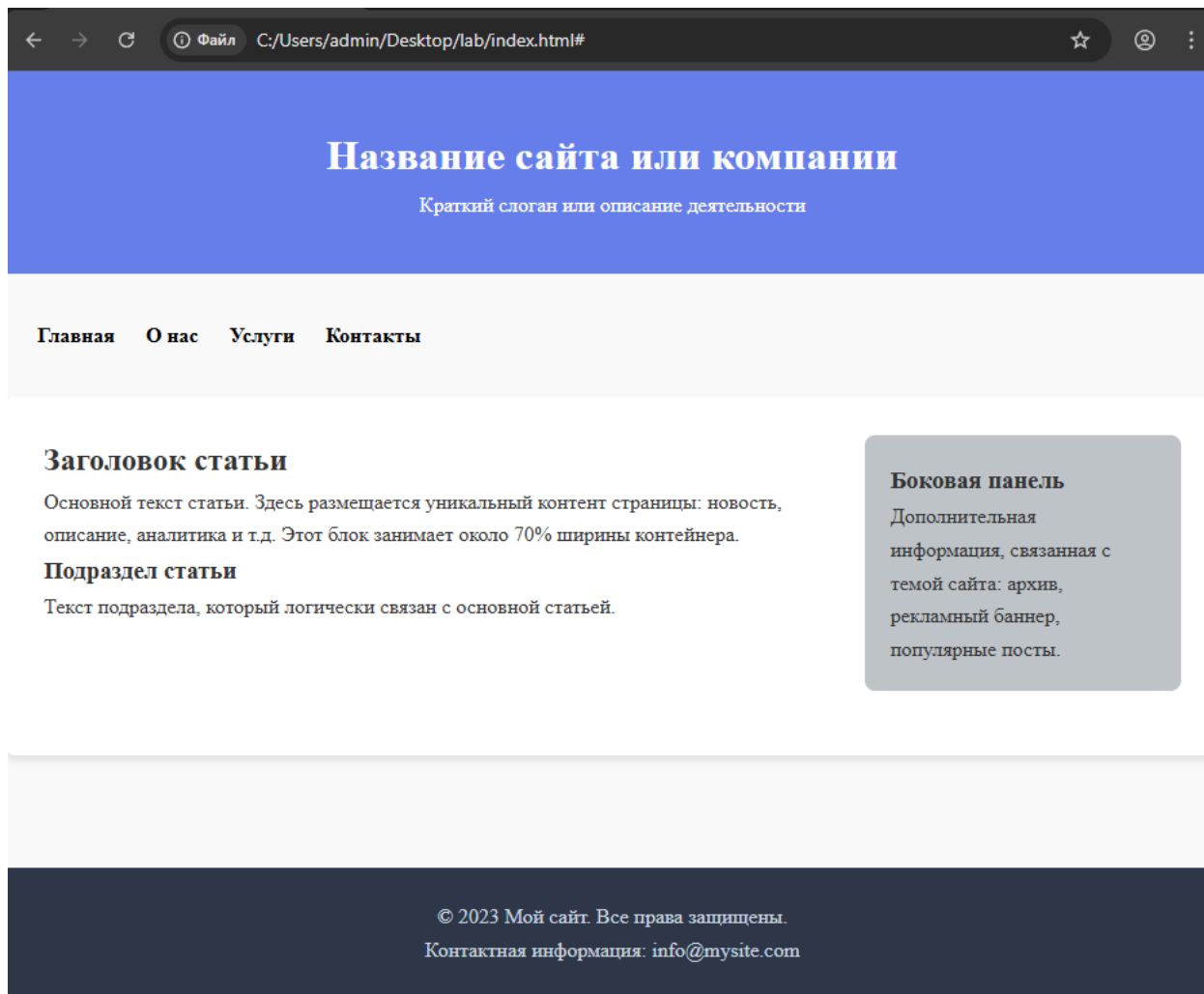


Рисунок 7.1 – Стилизация сайта.

Задания для выполнения лабораторной работы

Для сайта, созданного в лабораторной работе № 6, удалите стили внутри html-документа и создайте внешний файл стилей. Сделайте оформление сайта по вариантам.

Вариант 1. Сайт-портфолио веб-разработчика.

Вариант 2. Блог о путешествиях.

- Вариант 3.** Сайт кулинарных рецептов.
- Вариант 4.** Новостной портал.
- Вариант 5.** Сайт книжного клуба.
- Вариант 6.** Спортивный сайт.
- Вариант 7.** Сайт образовательного центра.
- Вариант 8.** Личный дневник.
- Вариант 9.** Сайт о домашних растениях.
- Вариант 10.** Сайт музыкальной группы.
- Вариант 11.** Сайт городского мероприятия.
- Вариант 12.** Сайт про здоровый образ жизни.
- Вариант 13.** Сайт кинотеатра.
- Вариант 14.** Сайт ремонта квартир.
- Вариант 15.** Сайт погодного сервиса.

Лабораторная работа № 8

«Отработка различных типов CSS-селекторов»

Цель работы: получить практические навыки разработки программ на языке CSS с использованием псевдоэлементов и псевдоклассов, классов и идентификаторов, блочной верстки.

Методические указания

Классы – наиболее часто используемые селекторы в CSS, которые используются для выбора одного или нескольких HTML-элементов с одинаковым значением атрибута CSS класса class.

Для того чтобы назначить класс элементу, необходимо использовать **атрибут class**.

```
<div class="rgups"></div>
```

После чего в CSS объявляется селектор класса с использованием точки перед именем класса.

```
.rgups{  
font: bold italic 120% Arial;  
}
```

Идентификаторы – селекторы, которые используются для выбора одного HTML-элемента с уникальным значением атрибута id.

Атрибут элемента id – идентификатор, который уникальн в во всем документе.

```
<div id="foot"></div>
```

Чтобы ссылаться на элементы по их идентификатору в CSS, используйте префикс # перед идентификатором:

```
#foot {  
border:silver solid 2px;  
}
```

Псевдоэлемент задает стиль элементов на определённых в дереве элементах документа, а также генерирует содержимое, которого нет в исходном коде текста.

Синтаксис использования псевдоэлементов:

Селектор:Псевдоэлемент { Описание правил стиля }

Сначала следует имя селектора, затем пишется двоеточие, после которого идёт имя псевдоэлемента.

Все псевдоэлементы представлены ниже:

1) **Псевдоэлемент :after** – применяется для вставки назначенного контента после содержимого элемента. Этот псевдоэлемент работает совместно со стилевым свойством content, которое определяет содержимое для вставки.

2) **Псевдоэлемент :before** – по своему действию :before аналогичен псевдоэлементу :after, но вставляет контент до содержимого элемента.

3) **Псевдоэлемент :first-letter** – определяет стиль первого символа в тексте элемента, к которому добавляется.

4) **Псевдоэлемент :first-line** – определяет стиль первой строки блочного текста. Длина этой строки зависит от многих факторов, таких как используемый шрифт, размер окна браузера, ширина блока, языка и т.д.

Псевдоклассы определяют динамическое состояние элементов, которое изменяется с помощью действий пользователя, а также положение в дереве документа. Синтаксис применения:

Селектор: Псевдокласс {Описание правил стиля}

Вначале указывается селектор, к которому добавляется псевдокласс, затем следует двоеточие, после которого идёт имя псевдокласса.

Условно все псевдоклассы делятся на три группы:

- определяющие состояние элементов;
- имеющие отношение к дереву элементов;
- указывающие язык текста.

Псевдоклассы, определяющие состояние элементов, – псевдоклассы, которые распознают текущее состояние элемента и применяют стиль только для этого состояния.

Псевдокласс :active – происходит при активации пользователем элемента.

Псевдокласс :link – применяется к непосещенным ссылкам, т.е. к таким ссылкам, на которые пользователь ещё не нажимал.

Псевдокласс :focus – применяется к элементу при получении им фокуса.

Псевдокласс :hover – активизируется, когда курсор мыши находится в пределах элемента, но щелчка по нему не происходит.

Псевдокласс :visited – применяется к посещённым ссылкам.

Псевдоклассы, имеющие отношение к дереву документа, – псевдоклассы, которые определяют положение элемента в дереве документа и применяют к нему стиль в зависимости от его статуса.

Псевдокласс:nth-child() – выбирает элементы на основе их позиции среди соседних элементов. Принимает различные аргументы.

Псевдокласс :first-child – применяется к первому дочернему элементу селектора, который расположен в дереве элементов документа.

Псевдоклассы, задающие язык текста – псевдоклассы для документов, одновременно содержащих тексты на нескольких языках, имеет значение соблюдение правил синтаксиса, характерных для того или иного языка.

Комбинаторы селекторов показывают связь между элементами:

1. Пробел – выбирает всех потомков:

`nav a { } /* Все ссылки внутри nav на любом уровне */`

2. `>` – выбирает только прямых детей:

`nav > a { } /* Только ссылки, которые прямо внутри nav */`

3. `+` – выбирает соседний элемент:

`h2 + p { } /* Параграф сразу после h2 */`

Пример сложной структуры сайта, добавление многоуровневого меню и блок с цитатами, стилизация с помощью CSS-селекторов (Рис 8.1):

В файле html:

```
<!DOCTYPE html>
<html lang="ru">
<head>
```

```

    <meta charset="UTF-8">
    <title>Обработка CSS-селекторов</title>
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <header id="main-header" class="header">
        <h1 class="site-title">Изучение CSS-селекторов</h1>
        <nav class="main-nav">
            <ul class="menu">
                <li class="menu-item">
                    <a href="#" class="menu-link">Главная</a>
                </li>
                <li class="menu-item has-submenu">
                    <a href="#" class="menu-link">Услуги</a>
                </li>
                <li class="menu-item">
                    <a href="#" class="menu-link">О нас</a>
                </li>
                <li class="menu-item has-submenu">
                    <a href="#" class="menu-link">Блог</a>
                </li>
                <li class="menu-item">
                    <a href="#" class="menu-link">Контакты</a>
                </li>
            </ul>
        </nav>
    </header>
    <main class="content">
        <section class="quotes-section">
            <h2 class="section-title">Избранные цитаты</h2>
            <div class="quotes-container">
                <blockquote class="quote important-quote">
                    <p class="quote-text">Код – это поэзия,
которую понимают компьютеры.</p>
                    <footer class="quote-author">– Анонимный
программист</footer>
                </blockquote>
                <blockquote class="quote">
                    <p class="quote-text">Веб-дизайн на 95%
состоит из типографики.</p>
                    <footer class="quote-author">– Оливер
Райхенштейн</footer>
                </blockquote>
                <blockquote class="quote">
                    <p class="quote-text">Сначала делайте
правильно, потом делайте красиво.</p>
                    <footer class="quote-author">– Майк
Тайсон</footer>
                </blockquote>
            </div>
        </section>

```

```

    </main>
    <footer class="main-footer">
        <p>© 2023 Лабораторная работа по CSS-селекторам</p>
    </footer>
</body>
</html>

```

В файле css:

```

* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}
body { /* Селектор по тегу */
    font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
    line-height: 1.6;
    color: #333;
    background-color: #f8f9fa;
}
#main-header { /* Селектор по id */
    background: linear-gradient(135deg, #2c3e50, #4a6491);
    color: white;
    padding: 30px 20px;
    text-align: center;
    margin-bottom: 30px;
}
.site-title { /* Селектор по классу */
    font-size: 2.5rem;
    margin-bottom: 20px;
    font-weight: 300;
}
.main-nav {
    background-color: #34495e;
    padding: 15px 0;
    margin-bottom: 40px;
}
.menu {
    list-style: none;
    display: flex;
    justify-content: center;
    gap: 30px;
}
.menu-item {
    padding: 10px 0;
}
.menu-link {
    color: white;
    text-decoration: none;
    padding: 10px 20px;
    display: inline-block;
}

```

```

        border-radius: 4px;
        font-weight: 500;
        transition: all 0.3s ease;
    }
    .menu-link:hover { /* Псевдокласс :hover для эффекта при наведении
*/
        background-color: #3498db;
        transform: scale(1.05);
    }
    .menu-item.has-submenu .menu-link { /* Селектор для элементов с
классом has-submenu */
        background-color: #2c3e50;
    }
    .content {
        max-width: 1200px;
        margin: 0 auto;
        padding: 0 20px;
    }
    .quotes-section {
        background-color: white;
        padding: 30px;
        border-radius: 10px;
        box-shadow: 0 4px 15px rgba(0,0,0,0.1);
    }
    .section-title {
        color: #2c3e50;
        font-size: 2rem;
        margin-bottom: 30px;
        padding-bottom: 10px;
        border-bottom: 2px solid #3498db;
    }
    .quotes-container {
        display: grid;
        grid-template-columns: repeat(auto-fit, minmax(300px, 1fr));
        gap: 25px;
    }
    .quote {
        background-color: #f8f9fa;
        padding: 25px;
        border-radius: 8px;
        border-left: 5px solid #3498db;
        transition: all 0.3s ease;
    }
    .quote:hover { /* Псевдокласс :hover для цитат */
        transform: translateY(-5px);
        box-shadow: 0 8px 20px rgba(0,0,0,0.12);
        background-color: white;
    }
    .quote-text {
        font-size: 1.1rem;
        font-style: italic;
    }

```

```

        color: #555;
        margin-bottom: 15px;
        line-height: 1.8;
    }
    .quote-author {
        font-weight: bold;
        color: #2c3e50;
        text-align: right;
        font-size: 0.9rem;
    }
    .quote:nth-child(even) { /* Четные цитаты */
        border-left-color: #2ecc71;
        background-color: #f0f8f0;
    }
    .quote:nth-child(even):hover {
        background-color: #e8f5e8;
    }
    .quote:nth-child(odd):not(.important-quote) { /* Нечетные цитаты */
        border-left-color: #9b59b6;
    }
    .quote:first-child { /* Первая цитата псевдокласс :first-child */
        border-left-color: #e74c3c;
        background-color: #fff0f0;
    }
    .quote:first-child:hover {
        background-color: #ffe8e8;
    }
    .quote:last-child { /* Последняя цитата псевдокласс :last-child */
        border-left-color: #f39c12;
    }
    .important-quote { /* Цитата с классом important-quote */
        border-width: 3px;
        font-weight: bold;
    }
    .quotes-container > .quote { /* Дочерний селектор */
        margin-bottom: 10px;
    }
    .quote footer {
        margin-top: 10px;
    }
    .quote + .quote { /* Соседний селектор */
        margin-top: 5px;
    }
    .main-footer {
        background-color: #2c3e50;
        color: white;
        text-align: center;
        padding: 25px 20px;
        margin-top: 50px;
    }
    .main-footer p {

```

```

        margin: 0;
        font-size: 0.9rem;
        opacity: 0.8;
    }
    .menu-item:nth-child(3n) { /* Стили для каждого третьего пункта меню
*/
        font-weight: bold;
    }

```

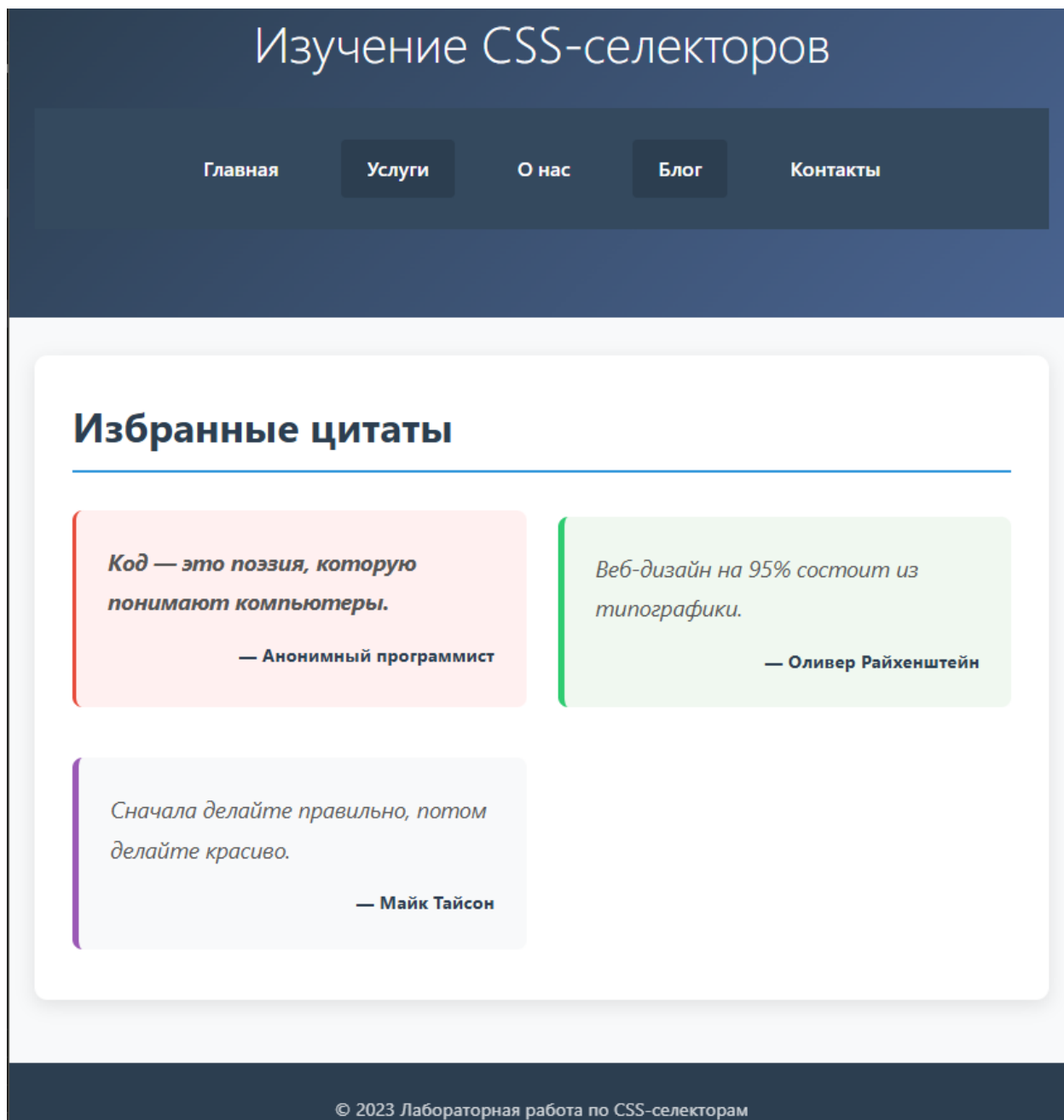


Рисунок 8.1 – Сложная структура сайта.

Задания для выполнения лабораторной работы

Создать сайт (шапка, основной блок, подвал) с использованием селекторов по классу, вложенности, псевдоклассов.

Вариант 1.

Сайт пиццерии

Меню: "Главная", "Меню", "Акции", "Доставка", "Контакты" (пункты "Меню" и "Акции" имеют особое оформление через классы).

Блок: 6 видов пиццы с названием и ценой. Чередовать цвета фона для пицц, выделить первую пиццу.

Вариант 2.

Сайт библиотеки

Меню: "Главная", "Жанры", "Авторы", "Новинки", "Контакты" (пункты "Жанры" и "Авторы" имеют особое оформление).

Блок: 6 книг с названием и автором. Книги по четным номерам с серым фоном, при наведении меняют цвет.

Вариант 3.

Сайт фильмов

Меню: "Главная", "Категории", "Кинотеатры", "Афиша", "О нас" (пункты "Категории" и "Кинотеатры" с особым оформлением).

Блок: 7 фильмов с названием и годом. Каждый третий фильм выделен, при наведении появляется тень.

Вариант 4.

Сайт цветочного магазина

Меню: "Главная", "Цветы", "Букеты", "Доставка", "Контакты" (пункты "Цветы" и "Букеты" с особыми классами).

Блок: 6 видов цветов с названием и ценой. Четные позиции с зеленым фоном.

Вариант 5.

Сайт компьютерных игр

Меню: "Главная", "Жанры", "Платформы", "Новинки", "Поддержка" (пункты "Жанры" и "Платформы" с классовыми отличиями).

Блок: 9 игр с названием и рейтингом. Игры с высоким рейтингом выделены, при наведении увеличиваются.

Вариант 6.

Сайт спортивных товаров

Меню: "Главная", "Виды спорта", "Бренды", "Акции", "Магазины" (пункты "Виды спорта" и "Бренды" выделены классами).

Блок: 8 товаров с названием и ценой. Товары по акции красным, четные с светлым фоном.

Вариант 7.

Сайт курсов

Меню: "Главная", "Направления", "Форматы", "Преподаватели", "Запись" (пункты "Направления" и "Форматы" с классовым оформлением).

Блок: 7 курсов с названием и длительностью. Короткие курсы выделены, при наведении меняют фон.

Вариант 8.

Сайт ресторана

Меню: "Главная", "Блюда", "Меню дня", "Бронирование", "Контакты" (пункты "Блюда" и "Меню дня" с особыми классами).

Блок: 7 блюд с названием и составом. Десерты розовым фоном.

Вариант 9.

Сайт телефонов

Меню: "Главная", "Бренды", "Ценовые категории", "Аксессуары", "Гарантия" (пункты "Бренды" и "Ценовые категории" с классами).

Блок: 8 моделей с названием и характеристиками. Четные модели выделены, при наведении показывают цену.

Вариант 10.

Сайт одежды

Меню: "Главная", "Одежда", "Коллекции", "Распродажа", "Размеры" (пункты "Одежда" и "Коллекции" с классовым оформлением).

Блок: 8 вещей с названием и размером. Четные с серым фоном.

Вариант 11.

Сайт автомобилей

Меню: "Главная", "Марки", "Типы", "Финансирование", "Тест-драйв" (пункты "Марки" и "Типы" с особыми классами).

Блок: 7 машин с моделью и годом. Четные машины зеленым, при наведении меняют цвет.

Вариант 12.

Сайт музыки

Меню: "Главная", "Жанры", "Исполнители", "Чарты", "События" (пункты "Жанры" и "Исполнители" с классами).

Блок: 7 альбомов с названием и исполнителем. Четные с темным фоном.

Вариант 13.

Сайт строительных материалов

Меню: "Главная", "Материалы", "Категории", "Доставка", "Скидки" (пункты "Материалы" и "Категории" с особым оформлением).

Блок: 8 материалов с названием и характеристиками. Популярные материалы выделены, при наведении показывают цену.

Вариант 14.

Сайт косметики

Меню: "Главная", "Типы", "Типы", "Советы", "Оплата" (пункты "Типы" и "Линии" с классовым оформлением).

Блок: 8 продуктов с названием и объемом. Четные продукты выделены.

Вариант 15.

Сайт мебели

Меню: "Главная", "Мебель", "Типы", "Доставка", "Оплата" (пункты "Мебель" и "Стили" с особыми классами).

Блок: 7 предметов с названием и размером. Предметы из новой коллекции выделены, при наведении появляется тень.

Лабораторная работа № 9

«Работа с блочной моделью: создание карточки товара»

Цель работы: освоить практические навыки работы с блочной моделью CSS для создания современных UI-карточек. Научиться точно управлять размерами, отступами, границами и позиционированием элементов внутри карточки товара.

Методические указания

Блочная модель CSS (CSS Box Model) – это фундаментальная концепция, которая описывает, как браузер рассчитывает размеры и расположение каждого элемента на веб-странице. Каждый HTML-элемент представляется в виде прямоугольного блока, состоящего из нескольких слоев, расположенных от центра к краю.

Структура блочной модели включает четыре основных компонента:

Контент – это центральная область, содержащая фактическое содержимое элемента: текст, изображения или другие медиа-объекты. Размеры контента определяются свойствами width (ширина) и height (высота).

Внутренний отступ (Padding) – прозрачная область между контентом и границей. Padding расширяет фон элемента, создавая визуальное пространство вокруг контента. Управляется свойствами padding, padding-top, padding-right, padding-bottom, padding-left.

Граница (Border) – линия, которая окружает padding и контент. Граница может иметь различные стили, толщину и цвет. Контролируется свойствами border, border-width, border-style, border-color, а также отдельными свойствами для каждой стороны (border-top, border-right и т.д.).

Внешний отступ (Margin) – прозрачная область за пределами границы, которая отделяет элемент от соседних элементов. Margin не является частью самого элемента и не принимает его фоновые свойства. Управляется свойствами margin, margin-top, margin-right, margin-bottom, margin-left.

Важное свойство **box-sizing** определяет, как вычисляются общие размеры элемента. По умолчанию используется значение content-box, при

котором `width` и `height` задают размер только контента, а `padding` и `border` добавляются к этим размерам. Значение `border-box` меняет это поведение: `width` и `height` определяют общий размер элемента, включая `padding` и `border`, что значительно упрощает расчеты и верстку.

Карточка товара – это самодостаточный UI-компонент, представляющий информацию о товаре в сжатом, визуально привлекательном формате. Типичная карточка включает:

- Изображение товара
- Название товара
- Краткое описание
- Цену
- Кнопку действия ("В корзину")

Свойство **`border-radius`** позволяет скруглять углы элементов. Можно задавать одно значение для всех углов или отдельные значения для каждого угла в порядке: верхний-левый, верхний-правый, нижний-правый, нижний-левый. Например, `border-radius: 10px;` или `border-radius: 10px 5px 15px 0;`. Значения могут быть в пикселях, процентах или других единицах измерения.

Тени и эффекты с помощью **`box-shadow`** добавляют карточке глубину и визуальный интерес. Синтаксис: `box-shadow: h-offset v-offset blur spread color inset;`. Например, `box-shadow: 0 4px 8px rgba(0,0,0,0.1);` создает легкую тень снизу.

Создадим адаптивную карточку товара с использованием всех принципов блочной модели.

Создаем файл `index.html` с каталогом товаров, разделенным на классы:

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Карточка товара - Блочная модель</title>
  <link rel="stylesheet" href="styles.css">
  <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.0.0/css/all.min.css">
</head>
```

```

<body>
  <div class="container">
    <h1>Каталог товаров</h1>
    <div class="products-grid">
      <div class="product-card"><!-- Карточка товара -->
        <div class="product-image">
          
        </div>
        <div class="product-content">
          <div class="product-category">Электроника</div>
          <h2 class="product-title">Смартфон XYZ Pro</h2>
          <p class="product-description">
            Современный смартфон с большим экраном,
            мощным процессором
            и тройной камерой для качественных фото.
          </p>
          <div class="product-price">
            <span class="current-price">24 999 ₺</span>
            <span class="old-price">31 249 ₺</span>
          </div>
          <button class="add-to-cart"> <!-- Кнопка действия
            <i class="fas fa-shopping-cart"></i>
            В корзину
          </button>
        </div>
      </div>
      <div class="product-card">
        <div class="product-image">
          
        </div>
        <div class="product-content">
          <div class="product-
            category">Электроника</div>
          <h2 class="product-title">Смартфон XYZ
            Pro</h2>
          <p class="product-description">
            Современный смартфон с большим
            экраном, мощным процессором
            и тройной камерой для
            качественных фото.
          </p>
          <div class="product-price">
            <span class="current-price">24
            999 ₺</span>
            <span class="old-price">31 249
            ₺</span>
          </div>
          <button class="add-to-cart">

```

```

        <i class="fas fa-shopping-
cart"></i>
        В корзину
    </button>
</div>
</div>
</div>
</div>
</div>
</body>
</html>
В файле css:
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box; /*включает border-box для всех
элементов*/
}
body {
    font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
    line-height: 1.6;
    color: #333;
    background-color: #f8f9fa;
    padding: 20px;
}
.container {
    max-width: 1200px;
    margin: 0 auto; /* Автоматические отступы слева/справа
центрируют блок */
    padding: 20px;
}
h1 {
    text-align: center;
    margin-bottom: 40px;
    color: #2c3e50;
    font-size: 2.5rem;
}
.product-card {
    width: 320px; /* Фиксированная ширина карточки */
    background-color: white;
    border-radius: 12px; /* Скругление всех углов */
    overflow: hidden; /* Скрывает выступающие за границы элементы
*/
    box-shadow: 0 5px 15px rgba(0, 0, 0, 0.08);
    transition: all 0.3s ease; /* Плавные переходы для hover-
эффектов */
    margin: 0 auto;
    border: 1px solid #eaeaea;
}
.product-image {
    position: relative;

```

```

        height: 300px; /* Фиксированная высота изображения */
        overflow: hidden; /* Скрывает части изображения, выходящие за
границы */
    }
    .product-image img {
        width: 100%;
        height: 100%;
        object-fit: cover; /* Сохраняет пропорции, обрезая при
необходимости */
        transition: transform 0.5s ease; /* Плавное увеличение при
наведении */
    }
    .product-card:hover .product-image img {
        transform: scale(1.05); /* Увеличиваем изображение при
наведении на 5% */
    }
    .product-content {
        padding: 20px;
    }
    .product-category {
        display: inline-block;
        color: #7f8c8d;
        font-size: 0.85rem;
        margin-bottom: 8px;
        text-transform: uppercase; /* Все буквы заглавные */
        letter-spacing: 0.5px;
    }
    .product-title {
        font-size: 1.4rem;
        color: #2c3e50;
        margin-bottom: 12px; /* Отступ снизу перед описанием */
        line-height: 1.4;
        border-bottom: 1px solid #eee;
        padding-bottom: 12px; /* Отступ между текстом и границей */
    }
    .product-description {
        color: #666;
        font-size: 0.95rem;
        line-height: 1.6;
        display: -webkit-box; /* Ограничение высоты с многоточием для
длинных описаний */
        -webkit-line-clamp: 3; /* Максимум 3 строки */
        -webkit-box-orient: vertical;
        overflow: hidden;
    }
    .product-price {
        margin-bottom: 25px;
        display: flex;
        align-items: center;
        gap: 15px;
    }

```

```

.current-price {
  font-size: 1.8rem;
  font-weight: bold;
  color: #e74c3c; /* Красный цвет для акционной цены */
}
.old-price {
  font-size: 1.2rem;
  color: #95a5a6;
  text-decoration: line-through; /* Перечеркнутый текст */
}
.add-to-cart {
  width: 100%; /* Ширина 100% от родительского блока */
  background-color: #3498db;
  color: white;
  border: none;
  padding: 14px 20px; /* Внутренние отступы: 14px сверху/снизу,
20px слева/справа */
  border-radius: 8px; /* Скругление углов кнопки */
  font-size: 1rem;
  font-weight: 600;
  cursor: pointer;
  transition: all 0.3s ease;
  display: flex;
  align-items: center;
  justify-content: center;
  gap: 10px;
}
.add-to-cart:hover {
  background-color: #2980b9; /* Темнее при наведении */
  transform: translateY(-2px); /* подъем */
}
.products-grid {
  display: grid;
  grid-template-columns: repeat(auto-fill, minmax(300px, 1fr));
  gap: 30px; /* Расстояние между карточками в сетке */
  margin-top: 40px;
}

```

Задания для выполнения лабораторной работы

Создать UI-карточку товара/услуги/продукта с использованием всех принципов блочной модели CSS.

Обязательные требования к карточке:

- Использовать box-sizing: border-box для всех элементов
- Задать точные размеры карточки
- Использовать внутренние отступы для всех контентных блоков
- Использовать внешние отступы для разделения элементов

- Добавить границы с разными стилями
- Применить скругление углов к карточке и некоторым внутренним элементам
- Добавить тень
- Реализовать hover-эффекты с transform
- Карточка должна содержать минимум: изображение, название, описание, цену, кнопку действия.

Вариант 1. Карточка пиццы для доставки.

Вариант 2. Карточка книги для интернет-магазина.

Вариант 3. Карточка фильма для киносайта.

Вариант 4. Карточка букета для цветочного магазина

Вариант 5. Карточка игры для игрового магазина.

Вариант 6. Карточка спортивного товара.

Вариант 7. Карточка онлайн-курса.

Вариант 8. Карточка блюда для ресторана.

Вариант 9. Карточка смарт-часов.

Вариант 10. Карточка одежды.

Вариант 11. Карточка автомобиля.

Вариант 12. Карточка музыкального альбома.

Вариант 13. Карточка строительного материала.

Вариант 14. Карточка косметического продукта.

Вариант 15. Карточка мебели.

Лабораторная работа № 10

«Оформление текстового блока и фонового изображения»

Цель работы: освоить практические навыки работы с фоновыми изображениями и наложения текста поверх них. Научиться создавать визуально привлекательные баннеры и шапки статей, используя свойства background-image.

Методические указания

Фоновое изображение (background-image) – это CSS-свойство, которое позволяет установить изображение в качестве фона для HTML-элемента. В отличие от тега , который является частью контента, фоновое изображение является декоративным элементом оформления. Это важное различие: изображения, несущие смысловую нагрузку, должны быть в контенте, а декоративные фоны – через CSS.

Основные свойства для работы с фоновыми изображениями:

1. **background-image:** url('путь/к/изображению.jpg') – задает фоновое изображение. Можно использовать абсолютные или относительные пути. Современный CSS также поддерживает градиенты как фоновое изображение, например: background-image: linear-gradient(...).

2. **background-size** – определяет размер фонового изображения:

- cover – масштабирует изображение с сохранением пропорций так, чтобы оно полностью покрывало элемент (может обрезаться)
- contain – масштабирует изображение так, чтобы оно полностью помещалось в элемент (может оставаться пустое пространство)
- Конкретные значения в пикселях или процентах: background-size: 100% 100% или background-size: 300px 200px

3. **background-position** – определяет позицию фонового изображения:

- Ключевые слова: center, top, bottom, left, right
- Конкретные значения в пикселях или процентах: background-position: 50% 50% или background-position: 20px 10px

- Можно комбинировать: background-position: right bottom

4. **background-repeat** – управляет повторением фонового изображения:

- no-repeat – не повторять
- repeat – повторять по обоим осям (как плитка)
- repeat-x – повторять только по горизонтали

5. **background-attachment** – определяет поведение фона при прокрутке:

- scroll – фон прокручивается вместе с элементом, по умолчанию
- fixed – фон фиксируется относительно окна просмотра
- local – фон прокручивается вместе с содержимым элемента

Полупрозрачный фон (overlay) – это техника наложения полупрозрачного слоя поверх фонового изображения для улучшения читаемости текста.

Создадим шапку для статьи с использованием фонового изображения (Рис 10.1).

Файл index.html:

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Статья</title>
  <link rel="stylesheet" href="styles.css">
  <link
href="https://fonts.googleapis.com/css2?family=Playfair+Display:wght@700&family=Roboto:wght@300;400;700&display=swap" rel="stylesheet">
</head>
<body>
  <header class="article-header">
    <div class="overlay"></div>
    <div class="header-content">
      <h1 class="article-title">Тайны древних городов:
Потерянные цивилизации Азии</h1>
      <p class="article-subtitle">Исследование забытых
культур и их наследия в современном мире</p>
      <div class="article-meta">
        <span class="author">
          <i class="fas fa-user"></i> Иванов Иван
        </span>
      </div>
    </div>
  </header>
</body>
</html>
```

МИН

```
        <span class="date">
            <i class="fas fa-calendar"></i> 15 ноября 2026
        </span>
        <span class="reading-time">
            <i class="fas fa-clock"></i> Время чтения: 1
        </span>
    </div>
    <a href="#content" class="cta-button">
        <span>Начать чтение</span>
        <i class="fas fa-arrow-down"></i>
    </a>
</div>
</header>
<main class="article-content" id="content">
    <p>Здесь будет основное содержание статьи...</p>
</main>
</body>
</html>
```

В файле css:

```
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}
body {
    font-family: 'Roboto', sans-serif;
    line-height: 1.6;
    color: #333;
}
.article-header {
    height: 100vh;
    min-height: 600px;
    display: flex;
    align-items: center;
    justify-content: center;
    text-align: center;
    color: white;
    overflow: hidden;
    background-image: url('back.jpg'); /* Фоновое изображение */
    background-size: cover; /* Масштабировать чтобы покрыть весь
элемент */
    background-position: center; /* Центрировать изображение */
    background-repeat: no-repeat; /* Не повторять изображение */
}
.overlay {
    position: absolute;
    top: 0;
    left: 0;
    right: 0;
```

```

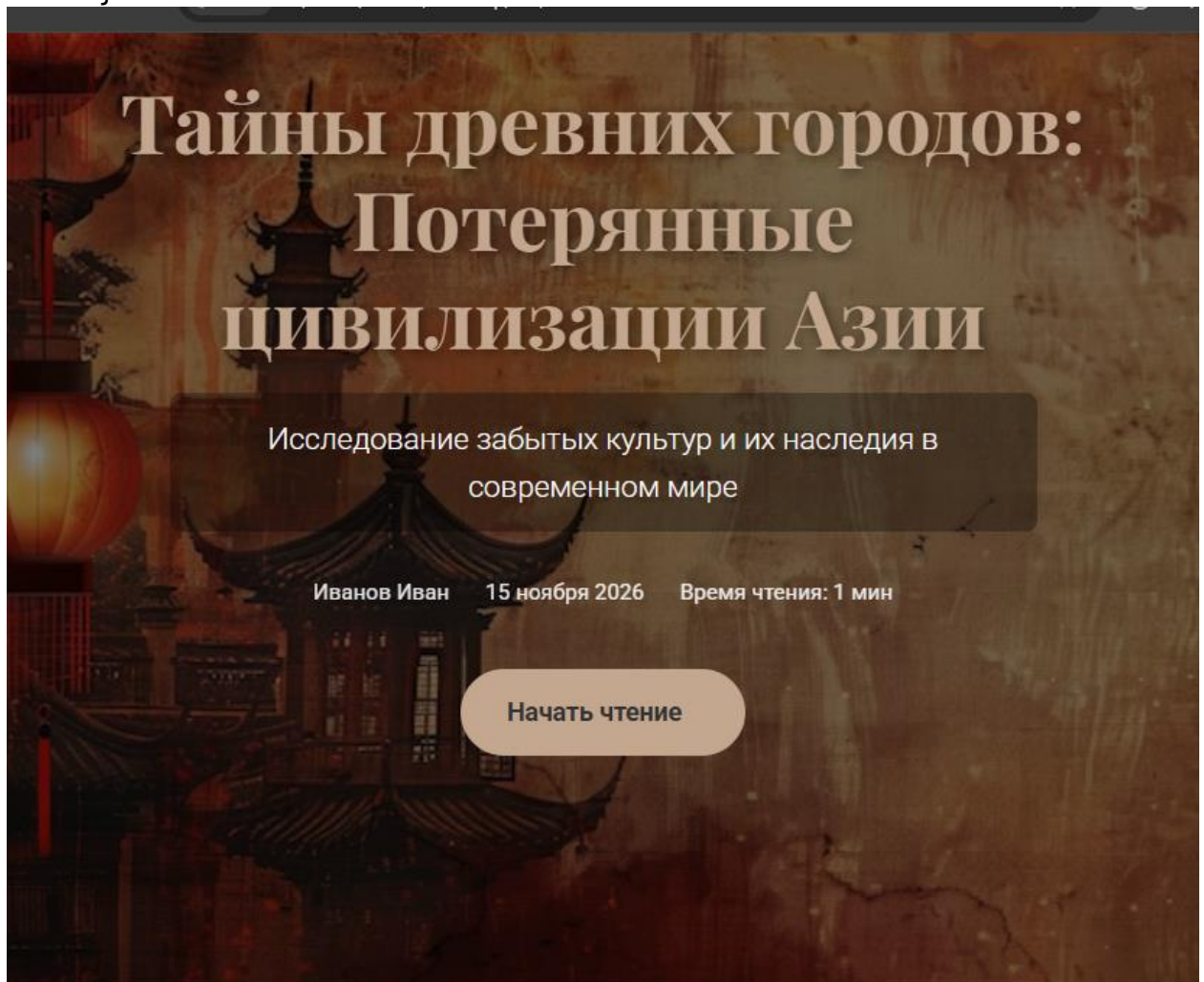
        bottom: 0;
        background: rgba(0, 0, 0, 0.5) 50%; /* Средняя прозрачность */
    }
    .header-content {
        z-index: 2; /* Над оверлеем */
        max-width: 800px; /* Ограничение ширины текста */
        padding: 40px;
    }
    .article-title {
        font-family: 'Playfair Display', serif;
        font-size: 3.5rem; /* Крупный размер для заголовка */
        font-weight: 700;
        line-height: 1.2;
        margin-bottom: 20px;
        text-shadow: 2px 2px 8px rgba(0, 0, 0, 0.5);
        color: #c3a78f;
    }
    .article-subtitle {
        font-size: 1.3rem;
        font-weight: 300;
        line-height: 1.6;
        margin-bottom: 30px;
        max-width: 600px;
        margin-left: auto;
        margin-right: auto;
        padding: 15px; /* Полупрозрачная подложка */
        background-color: rgba(0, 0, 0, 0.3);
        border-radius: 8px;
    }
    .article-meta {
        display: flex;
        justify-content: center;
        flex-wrap: wrap; /* Перенос на новую строку при нехватке места
*/
        gap: 25px;
        margin-bottom: 40px;
        font-size: 0.95rem;
        opacity: 0.9; /* прозрачность */
    }
    .cta-button {
        display: inline-flex;
        align-items: center;
        gap: 12px;
        background-color: #c3a78f;
        color: #333;
        text-decoration: none;
        padding: 16px 32px;
        border-radius: 50px;
        font-weight: 700;
        font-size: 1.1rem;
        transition: all 0.3s ease;
    }

```

```

}
.cta-button:hover {
    transform: translateY(-3px); /* движение при наведении */
}
.article-content {
    max-width: 800px;
    margin: 80px auto;
    padding: 0 20px;
    font-size: 1.1rem;
    line-height: 1.8;
}

```



Здесь будет основное содержание статьи...

Рисунок 20.1 – Шапка для статьи с использованием фонового изображения.

Задания для выполнения лабораторной работы

Создать баннер или шапку статьи с фоновым изображением, полупрозрачным оверлеем и стилизованным текстом поверх.

- Вариант 1.** Баннер для туристического агентства.
- Вариант 2.** Шапка статьи о здоровом питании.
- Вариант 3.** Баннер для онлайн-курса.
- Вариант 4.** Шапка блога о фотографии.
- Вариант 5.** Баннер для фитнес-клуба.
- Вариант 6.** Шапка статьи о кино.
- Вариант 7.** Баннер для книжного магазина.
- Вариант 8.** Шапка кулинарного блога.
- Вариант 9.** Баннер для музыкального фестиваля.
- Вариант 10.** Шапка статьи о технологии.
- Вариант 11.** Баннер для цветочного магазина.
- Вариант 12.** Шапка спортивного блога.
- Вариант 13.** Баннер для кофейни.
- Вариант 14.** Шапка статьи об архитектуре.
- Вариант 15.** Баннер для игры.

Лабораторная работа № 11

«Позиционирование элементов: создание навигационного меню с выпадающим списком»

Цель работы: освоить практические навыки позиционирования элементов с помощью CSS-свойства `position`. Научиться создавать многоуровневые навигационные меню с выпадающими подменю, используя абсолютное позиционирование и псевдоклассы.

Методические указания

Позиционирование элементов (CSS Positioning) – это система CSS, которая позволяет управлять расположением элементов на веб-странице, выходя за рамки стандартного потока документа. Понимание позиционирования критически важно для создания сложных макетов и интерактивных интерфейсов.

Типы позиционирования в CSS:

1. **`position: static`** – значение по умолчанию. Элемент располагается в обычном потоке документа согласно его порядку в HTML. Свойства `top`, `right`, `bottom`, `left` и `z-index` не применяются к статически позиционированным элементам.

2. **`position: relative`** – элемент позиционируется относительно его нормального положения в потоке документа. После установки `position: relative` можно использовать свойства `top`, `right`, `bottom`, `left` для смещения элемента от его исходной позиции. Важно: пространство, которое занимал элемент до смещения, остается занятым в потоке документа.

3. **`position: absolute`** – элемент извлекается из нормального потока документа и позиционируется относительно его ближайшего позиционированного предка. Если такого предка нет, элемент позиционируется относительно окна браузера. Абсолютно позиционированные элементы не занимают места в потоке документа и могут перекрывать другие элементы.

4. **position: fixed** – элемент фиксируется относительно окна просмотра и не перемещается при прокрутке страницы. Как и абсолютно позиционированные элементы, фиксированные извлекаются из потока документа.

5. **position: sticky** – гибрид **relative** и **fixed**. Элемент ведет себя как **relative** до достижения определенного положения прокрутки.

Контекст позиционирования – это концепция, определяющая, относительно какого элемента происходит позиционирование абсолютно позиционированных потомков. Элемент создает новый контекст позиционирования, если у него установлено **position: relative, absolute, fixed** или **sticky**. Это важно для выпадающих меню: подменю должно позиционироваться относительно своего родительского пункта меню, а не всей страницы.

Свойство **z-index** контролирует вертикальное наложение элементов **stacking order**. Более высокое значение **z-index** помещает элемент поверх элементов с меньшим значением. Важно: **z-index** работает только для позиционированных элементов.

Выпадающее меню – это паттерн пользовательского интерфейса, при котором подменю скрыто по умолчанию и появляется при взаимодействии пользователя, обычно при наведении курсора или клике. Техническая реализация включает:

- Родительский пункт меню с **position: relative**
- Выпадающий список с **position: absolute**
- Скрытие/отображение с помощью CSS (**display: none/block** или **visibility: hidden/visible** с **opacity**)
- Активацию через псевдокласс **:hover**

Псевдокласс :hover применяет стили к элементу, когда пользователь наводит на него указатель мыши, но не активирует его (не нажимает). Для выпадающих меню обычно используется селектор **parent:hover child**, который активирует дочерний элемент при наведении на родителя.

Свойства видимости:

- **display: none** – полностью удаляет элемент из потока, освобождая занимаемое им пространство
- **visibility: hidden** – скрывает элемент, но сохраняет занимаемое им пространство
- **opacity: 0** – делает элемент полностью прозрачным, но сохраняет его функциональность.

Трансформации и переходы transform и transition используются для создания плавных анимаций при открытии/закрытии меню.

Например:

- **transform:** translateY(-10px) для начального смещения
- **transition:** all 0.3s ease для плавного изменения

Создадим горизонтальное навигационное меню с выпадающими подменю.

Файл index.html:

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Навигационное меню с выпадающим списком</title>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <div class="container">
    <header class="header"><p>WebSite</p></header>
    <nav class="main-nav">
      <ul class="menu">
        <li class="menu-item"><a href="#" class="menu-link">Главная</a></li>
        <li class="menu-item dropdown">
          <a href="#" class="menu-link">Услуги</a>
          <ul class="dropdown-menu"><!-- Выпадающее
подменю -->
            <li class="dropdown-item"><a href="#"
class="dropdown-link">Веб-разработка</a></li>
            <li class="dropdown-item"><a href="#"
class="dropdown-link">Веб-дизайн</a></li>
            <li class="dropdown-item"><a href="#"
class="dropdown-link">Продвижение</a></li>
```

```

        <li class="dropdown-item"><a href="#"
class="dropdown-link">Мобильные приложения</a></li>
    </ul>
</li>
<li class="menu-item megamenu">
    <a href="#" class="menu-link">Категории</a>
    <div class="megamenu-content">
        <div class="megamenu-column">
            <h3 class="megamenu-
title">Технологии</h3>
            <ul class="megamenu-list">
                <li><a
href="#">Программирование</a></li>
                <li><a href="#">Базы
данных</a></li>
                <li><a href="#">DevOps</a></li>
                <li><a
href="#">Кибербезопасность</a></li>
            </ul>
        </div>
        <div class="megamenu-column">
            <h3 class="megamenu-
title">Дизайн</h3>
            <ul class="megamenu-list">
                <li><a href="#">UI/UX
Дизайн</a></li>
                <li><a href="#">Графический
дизайн</a></li>
                <li><a href="#">3D
Моделирование</a></li>
                <li><a href="#">Анимация</a></li>
            </ul>
        </div>
        <div class="megamenu-column">
            <h3 class="megamenu-
title">Бизнес</h3>
            <ul class="megamenu-list">
                <li><a
href="#">Маркетинг</a></li>
                <li><a
href="#">Менеджмент</a></li>
                <li><a href="#">Финансы</a></li>
                <li><a href="#">Стартапы</a></li>
            </ul>
        </div>
    </div>
</li>
<li class="menu-item"><a href="#" class="menu-
link">О нас</a></li>
    <li class="menu-item dropdown nested"> <a href="#"
class="menu-link">Контакты</a>

```

```

        <ul class="dropdown-menu">
            <li class="dropdown-item"><a href="#"
class="dropdown-link">Офисы</a></li>
            <li class="dropdown-item"><a href="#"
class="dropdown-link">Телефоны</a></li>
            <li class="dropdown-item"><a href="#"
class="dropdown-link">Email</a></li>
        </ul>
    </li>
</ul>
</nav>
<main class="content">
    <h1>Демонстрация навигационного меню</h1>
    <p>Наведите курсор на пункты "Услуги", "Категории" или
"Контакты" чтобы увидеть выпадающие меню.</p>
    <div class="demo-grid">
        <div class="demo-card">
            <h3>Позиционирование</h3>
            <p>Элементы меню используют position: relative
и absolute</p>
        </div>
        <div class="demo-card">
            <h3>:hover эффекты</h3>
            <p>Подменю появляется при наведении
курсора</p>
        </div>
        <div class="demo-card">
            <h3>Z-index</h3>
            <p>Для управления наложением элементов</p>
        </div>
    </div>
</main>
</div>
</body>
</html>

```

В файле css:

```

* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}
body {
    font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
    line-height: 1.6;
    color: #333;
    background-color: #f8f9fa;
}
.container {
    max-width: 1200px;
    margin: 0 auto;
}

```

```

        padding: 20px;
    }
    .header {
        background: linear-gradient(135deg, #2c3e50, #4a6491);
        color: white;
        padding: 30px;
        border-radius: 10px;
        margin-bottom: 30px;
        text-align: center;
    }
    .header p {
        display: flex;
        align-items: center;
        justify-content: center;
        gap: 15px;
        font-size: 2.5rem;
        font-weight: bold;
        margin-bottom: 15px;
    }
    .main-nav {
        background-color: white;
        border-radius: 10px;
        box-shadow: 0 4px 15px rgba(0, 0, 0, 0.1);
        margin-bottom: 40px;
        padding: 0 20px;
    }
    .menu {
        list-style: none;
        display: flex; /* Горизонтальное расположение пунктов */
        justify-content: center; /* Центрирование меню */
    }
    .menu-item {
        position: relative;
    }
    .menu-link {
        display: flex;
        align-items: center;
        gap: 10px;
        color: #333;
        text-decoration: none;
        padding: 20px 25px;
        font-weight: 500;
        font-size: 1.1rem;
        transition: all 0.3s ease;
        border-bottom: 3px solid transparent; /* Для hover-эффекта */
    }
    .dropdown-arrow {
        font-size: 0.8rem;
        margin-left: 5px;
        transition: transform 0.3s ease;
    }

```

```

.dropdown-menu {
    position: absolute;
    top: 100%;
    left: 0;
    background-color: white;
    min-width: 250px;
    box-shadow: 0 8px 25px rgba(0, 0, 0, 0.15);
    border-radius: 0 0 10px 10px;
    list-style: none;
    padding: 10px 0;
    z-index: 1000; /* поверх других элементов */
    opacity: 0; /* Скрываем подменю по умолчанию */
    visibility: hidden;
    transform: translateY(-10px);
    transition: all 0.3s ease;
}
.dropdown:hover .dropdown-menu {
    opacity: 1;
    visibility: visible;
    transform: translateY(0);
}
.dropdown-link {
    display: flex;
    align-items: center;
    gap: 10px;
    color: #333;
    text-decoration: none;
    padding: 12px 25px;
    transition: all 0.2s ease;
}
.dropdown-link:hover {
    background-color: #f8f9fa;
    color: #3498db;
    padding-left: 30px; /* Сдвигаем текст при наведении */
}
.megamenu-content {
    position: absolute;
    top: 100%;
    left: 0;
    background-color: white;
    width: 800px;
    box-shadow: 0 8px 25px rgba(0, 0, 0, 0.15);
    border-radius: 0 0 10px 10px;
    padding: 30px;
    z-index: 1000;
    opacity: 0;
    visibility: hidden;
    transform: translateY(-10px);
    transition: all 0.3s ease;
    display: grid; /* Распределение на колонки */
    grid-template-columns: repeat(3, 1fr);
}

```

```

        gap: 30px;
    }
    .megamenu:hover .megamenu-content {
        opacity: 1;
        visibility: visible;
        transform: translateY(0);
    }
    .megamenu-list {
        list-style: none;
    }
    .megamenu-list li {
        margin-bottom: 10px;
    }
    .megamenu-list a {
        color: #555;
        text-decoration: none;
        transition: color 0.2s ease;
        display: block;
        padding: 5px 0;
    }
    .megamenu-list a:hover {
        color: #3498db;
        padding-left: 5px;
    }
    .content {
        background-color: white;
        padding: 40px;
        border-radius: 10px;
        box-shadow: 0 4px 15px rgba(0, 0, 0, 0.1);
    }
    .demo-grid {
        display: grid;
        grid-template-columns: repeat(3, 1fr);
        gap: 20px;
        margin-top: 40px;
    }
    .demo-card {
        background-color: #f8f9fa;
        padding: 25px;
        border-radius: 8px;
        border-left: 4px solid #3498db;
    }

```

Задания для выполнения лабораторной работы

Создать сайт с использованием навигационного меню с выпадающим списком.

Вариант 1. Сайт пиццерии. Меню: "Главная", "Меню": [Виды пиццы, Напитки, Десерты], "Акции": [Скидки, Комбо, Сезонные], "Доставка", "Контакты".

Вариант 2. Сайт библиотеки. Меню: "Главная", "Жанры": [Художественная, Научная, Детская, "Авторы": [Русские, Зарубежные, Современные], "Новинки", "Контакты".

Вариант 3. Сайт фильмов. Меню: "Главная", "Категории": [Боевики, Комедии, Драмы], "Кинотеатры": [Центральный, Северный, Западный], "Афиша", "О нас".

Вариант 4. Сайт цветочного магазина. Меню: "Главная", "Цветы": [Розы, Тюльпаны, Орхидеи, "Букеты": [Свадебные, Праздничные, Деловые], "Доставка", "Контакты".

Вариант 5. Сайт компьютерных игр. Меню: "Главная", "Жанры": [Стратегии, RPG, Шутеры], "Платформы": [PC, PlayStation, Xbox], "Новинки", "Поддержка".

Вариант 6. Сайт спортивных товаров. Меню: "Главная", "Виды спорта": [Футбол, Теннис, Бег], "Бренды": [Nike, Adidas, Puma], "Акции", "Магазины".

Вариант 7. Сайт курсов. Меню: "Главная", "Направления": [Программирование, Дизайн, Маркетинг], "Форматы": [Онлайн, Офлайн, Интенсив], "Преподаватели", "Запись".

Вариант 8. Сайт ресторана. Меню: "Главная", "Блюда": [Закуски, Основные, Десерты], "Меню дня": [Бизнес-ланч, Ужин, Детское], "Бронирование", "Контакты".

Вариант 9. Сайт телефонов. Меню: "Главная", "Бренды": [Apple, Samsung, Xiaomi], "Ценовые категории": [Бюджетные, Средние, Флагманы], "Аксессуары", "Гарантия".

Вариант 10. Сайт одежды. Меню: "Главная", "Одежда": [Мужская, Женская, Детская], "Коллекции": [Весна-лето, Осень-зима, Специальные], "Распродажа", "Размеры".

Вариант 11. Сайт автомобилей. Меню: "Главная", "Марки": [Toyota, BMW, Lada], "Типы": [Седаны, Внедорожники, Спорткары], "Финансирование", "Тест-драйв".

Вариант 12. Сайт музыки. Меню: "Главная", "Жанры": [Рок, Поп, Джаз, "Исполнители": [Солисты, Группы, Хоры], "Чарты", "События".

Вариант 13. Сайт строительных материалов. Меню: "Главная", "Материалы": [Дерево, Металл, Пластик], "Категории": [Для дома, Для улицы, Специальные], "Доставка", "Скидки".

Вариант 14. Сайт косметики. Меню: "Главная", "Типы": [Для лица, Для волос, Для тела], "Типы": [Натуральная, Профессиональная, Люкс], "Советы", "Оплата".

Вариант 15. Сайт мебели. Меню: "Главная", "Мебель": [Для кухни, Для гостиной, Для спальни], "Типы": [Классика, Модерн, Лофт], "Доставка", "Оплата".

Лабораторная работа № 12

«Позиционирование элементов: создание фиксированной панели и модального окна»

Цель работы: освоить практические навыки работы с различными типами CSS-позиционирования. Научиться создавать фиксированные элементы интерфейса и модальные окна pop-up.

Методические указания

Модальное окно (Modal Window) – это диалоговое окно, которое появляется поверх основного содержимого страницы и требует взаимодействия с пользователем перед возвращением к основному контенту. Модальные окна используются для форм, подтверждений, уведомлений, детальной информации. Техническая реализация включает:

- Контейнер модального окна с position: fixed для покрытия всего экрана
- Внутреннее окно с контентом
- Фон с полупрозрачностью
- Механизм открытия/закрытия
- Управление фокусом и доступностью

Фиксированная панель (Fixed Panel) – элемент интерфейса, который остается на фиксированной позиции при прокрутке страницы. Чаще всего используется для навигационных панелей (фиксированная шапка), Боковых панелей с действиями, панелей инструментов и уведомлений

Кнопка «Наверх» (Back to Top Button) – фиксированная кнопка, обычно расположенная в правом нижнем углу страницы, которая при клике плавно прокручивает страницу к началу. Особенности реализации:

- position: fixed для фиксации положения
- Плавная прокрутка к началу (через scrollTo или scrollToView)
- Доступность для клавиатурной навигации

Анимации и переходы для модальных окон и фиксированных элементов создают плавный пользовательский опыт:

- Плавное появление/исчезновение через opacity и transform
- Анимация масштабирования для модальных окон
- Плавная прокрутка для кнопок «Наверх»
- Переходы для sticky-элементов

Создадим веб-страницу с фиксированной шапкой и модальным окном, которое открывается по клику.

Файл index.html:

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Фиксированная панель и модальное окно</title>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <header class="fixed-header"><!-- Фиксированная шапка (fixed
positioning) -->
    <div class="header-content">
      <div class="logo">WebLab</div>
      <nav class="main-nav">
        <a href="#" class="nav-link active">Главная</a>
        <a href="#" class="nav-link">О нас</a>
        <a href="#" class="nav-link">Услуги</a>
        <a href="#" class="nav-link">Контакты</a>
      </nav>
      <label for="modal-toggle" class="open-modal-btn"><!--
Кнопка открытия модального окна -->
        Связаться
      </label>
    </div>
  </header>
  <div class="sticky-panel"><!-- Липкая панель (sticky
positioning) -->
    <div class="sticky-content">
      <span>Акция: Скидка 20% на все услуги до конца
месяца!</span>
    </div>
  </div>
  <!-- Основное содержимое -->
  <main class="content">
    <section class="hero">
      <h1>Изучение позиционирования в CSS</h1>
      <p>Фиксированные, абсолютные и sticky-элементы</p>
      <!-- Вторая кнопка открытия модального окна -->
      <label for="modal-toggle" class="cta-button">
        Открыть модальное окно
```

```

        </label>
    </section>
    <section class="info-section">
        <h2>Типы позиционирования</h2>
        <div class="info-grid">
            <div class="info-card">
                <h3>Fixed</h3>
                <p>Фиксированное позиционирование относительно
окна браузера</p>
            </div>
            <div class="info-card">
                <h3>Sticky</h3>
                <p>Липкое позиционирование - гибрид relative и
fixed</p>
            </div>
            <div class="info-card">
                <h3>Absolute</h3>
                <p>Абсолютное позиционирование относительно
ближайшего позиционированного предка</p>
            </div>
        </div>
    </section>
    <section class="long-content">
        <h2>Контент для демонстрации прокрутки</h2>
        <div class="content-text">
            <p>CSS предлагает несколько типов позиционирования
элементов на веб-странице. Понимание этих типов критически важно для
создания современных интерфейсов.</p>

            <p><strong>Position: static</strong> - это
значение по умолчанию. Элементы располагаются в нормальном потоке
документа.</p>

            <p><strong>Position: relative</strong> - элемент
позиционируется относительно своего нормального положения. Можно смещать
с помощью top, right, bottom, left.</p>
        </div>
    </section>
</main>
<input type="checkbox" id="modal-toggle" class="modal-
toggle"><!-- Скрытый чекбокс для управления модальным окном -->
<div class="modal-overlay"><!-- Модальное окно (fixed
positioning с абсолютным позиционированием внутри) -->
    <div class="modal-container">
        <label for="modal-toggle" class="modal-close"><!--
Кнопка закрытия -->
            x
        </label>
        <div class="modal-content">
            <h2>Связаться с нами</h2>

```

```

        <p>Заполните форму ниже и мы свяжемся с вами в
ближайшее время</p>
        <form class="contact-form">
            <div class="form-group">
                <label for="name">Имя</label>
                <input
placeholder="Введите ваше имя">
                    type="text" id="name"
                </div>
            <div class="form-group">
                <label for="email">Email</label>
                <input
placeholder="Введите ваш email">
                    type="email" id="email"
                </div>
            <div class="form-group">
                <label for="message">Сообщение</label>
                <textarea
placeholder="Введите ваше сообщение"></textarea>
                    id="message" rows="4"
                </div>
            <button type="submit" class="submit-btn">
                Отправить
            </button>
        </form>
    </div>
</div>
</body>
</html>

```

В файле css:

```

* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}
body {
    font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
    line-height: 1.6;
    color: #333;
    background-color: #f8f9fa;
    padding-top: 70px; /* Отступ для фиксированной шапки */
}
.fixed-header {
    position: fixed; /* ФИКСИРОВАННОЕ ПОЗИЦИОНИРОВАНИЕ */
    top: 0; /* Прижимаем к верхнему краю окна */
    left: 0; /* Прижимаем к левому краю */
    width: 100%; /* На всю ширину окна */
    background-color: white;
    box-shadow: 0 2px 15px rgba(0, 0, 0, 0.1);
    z-index: 1000; /* Высокий z-index чтобы быть поверх других
элементов */
    padding: 15px 0;
}

```

```

}
.header-content {
    max-width: 1200px;
    margin: 0 auto;
    justify-content: space-between;
    align-items: center;
    padding: 0 20px;
}
.logo {
    font-size: 1.5rem;
    font-weight: bold;
    color: #2c3e50;
}
.main-nav {
    gap: 30px;
}
.nav-link {
    color: #333;
    text-decoration: none;
    font-weight: 500;
    padding: 8px 15px;
    border-radius: 5px;
    transition: all 0.3s ease;
}
.nav-link:hover,
.nav-link.active {
    background-color: #3498db;
    color: white;
}
.open-modal-btn {
    background-color: #3498db;
    color: white;
    border: none;
    padding: 10px 20px;
    border-radius: 5px;
    font-weight: 500;
    cursor: pointer;
    transition: all 0.3s ease;
    display: inline-block;
    text-decoration: none;
}
.open-modal-btn:hover {
    background-color: #2980b9;
    transform: translateY(-2px);
}
.sticky-panel {
    position: sticky; /* ЛИПКОЕ ПОЗИЦИОНИРОВАНИЕ */
    top: 70px; /* Прилипает когда достигает 70px от верха окна */
    background: linear-gradient(90deg, #e74c3c, #c0392b);
    color: white;
    z-index: 999; /* Ниже шапки но выше контента */
}

```

```

        padding: 15px 0;
    }
    .sticky-content {
        max-width: 1200px;
        margin: 0 auto;
        justify-content: space-between;
        align-items: center;
        padding: 0 20px;
    }
    .sticky-close {
        background: none;
        border: none;
        color: white;
        cursor: pointer;
        font-size: 1.5rem;
        padding: 0 10px;
        transition: color 0.3s ease;
    }
    .sticky-close:hover {
        color: #ffcccc;
    }
    .content {
        max-width: 1200px;
        margin: 0 auto;
        padding: 40px 20px;
    }
    .hero {
        text-align: center;
        padding: 80px 0;
        background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
        color: white;
        border-radius: 15px;
        margin-bottom: 60px;
    }
    .hero h1 {
        font-size: 3rem;
        margin-bottom: 20px;
    }
    .hero p {
        font-size: 1.2rem;
        margin-bottom: 30px;
        opacity: 0.9;
    }
    .cta-button {
        background-color: white;
        color: #667eea;
        border: none;
        padding: 15px 40px;
        font-size: 1.1rem;
        font-weight: bold;
        border-radius: 50px;
    }

```

```

        cursor: pointer;
        transition: all 0.3s ease;
        display: inline-block;
        text-decoration: none;
    }
    .cta-button:hover {
        transform: translateY(-3px);
        box-shadow: 0 10px 20px rgba(0, 0, 0, 0.2);
    }
    .info-section {
        margin-bottom: 60px;
    }
    .info-section h2 {
        text-align: center;
        font-size: 2.5rem;
        color: #2c3e50;
        margin-bottom: 40px;
    }
    .info-grid {
        display: grid;
        grid-template-columns: repeat(3, 1fr);
        gap: 30px;
    }
    .info-card {
        background-color: white;
        padding: 30px;
        border-radius: 10px;
        box-shadow: 0 5px 15px rgba(0, 0, 0, 0.1);
        transition: all 0.3s ease;
        text-align: center;
    }
    .info-card:hover {
        transform: translateY(-10px);
        box-shadow: 0 15px 30px rgba(0, 0, 0, 0.15);
    }
    .info-card h3 {
        color: #3498db;
        font-size: 1.5rem;
        margin-bottom: 15px;
    }
    .long-content {
        padding: 60px 0;
    }
    .long-content h2 {
        text-align: center;
        color: #2c3e50;
        margin-bottom: 40px;
    }
    .content-text {
        background-color: white;
        padding: 40px;
    }

```

```

        border-radius: 10px;
        box-shadow: 0 5px 15px rgba(0, 0, 0, 0.1);
    }
    .content-text p {
        margin-bottom: 20px;
        line-height: 1.8;
    }
    .modal-toggle {/* СКРЫТЫЙ ЧЕКБОКС ДЛЯ УПРАВЛЕНИЯ МОДАЛЬНЫМ ОКНОМ
*/
        display: none;
    }
    .modal-overlay {/* МОДАЛЬНОЕ ОКНО (fixed + absolute positioning)
*/
        position: fixed;
        top: 0;
        left: 0;
        right: 0;
        bottom: 0;
        background-color: rgba(0, 0, 0, 0.7);
        align-items: center;
        justify-content: center;
        z-index: 2000; /* Самый высокий z-index - поверх всего */
        opacity: 0; /* Скрываем по умолчанию */
        visibility: hidden; /* Скрываем от взаимодействия */
        transition: all 0.3s ease;
    }
    .modal-toggle:checked ~ .modal-overlay {/* Показываем модальное
окно когда чекбокс отмечен */
        opacity: 1; /* Показываем оверлей */
        visibility: visible; /* Делаем видимым */
    }

    .modal-container {
        position: relative; /* СОЗДАЕМ КОНТЕКСТ ПОЗИЦИОНИРОВАНИЯ */
        background-color: white;
        width: 90%;
        max-width: 500px;
        border-radius: 15px;
        padding: 40px;
        box-shadow: 0 20px 60px rgba(0, 0, 0, 0.3);
        transform: translateY(-50px); /* Для анимации появления */
        transition: transform 0.3s ease;
    }
    .modal-toggle:checked ~ .modal-overlay .modal-container {/*
Анимация появления модального окна */
        transform: translateY(0);
    }
    .modal-close {/* Кнопка закрытия модального окна*/
        position: absolute;
        top: 15px;
        right: 15px;

```

```

        background: none;
        border: none;
        font-size: 1.8rem;
        color: #7f8c8d;
        cursor: pointer;
        padding: 5px;
        transition: color 0.3s ease;
        text-decoration: none;
        display: inline-block;
    }
    .modal-content h2 {
        color: #2c3e50;
        margin-bottom: 15px;
    }
    .modal-content > p {
        color: #7f8c8d;
        margin-bottom: 30px;
    }
    .contact-form {
        display: flex;
        flex-direction: column;
        gap: 20px;
    }
    .form-group {
        display: flex;
        flex-direction: column;
        gap: 8px;
    }
    .form-group label {
        font-weight: 500;
        color: #2c3e50;
    }
    .form-group input,
    .form-group textarea {
        padding: 12px 15px;
        border: 1px solid #ddd;
        border-radius: 8px;
        font-family: inherit;
        font-size: 1rem;
        transition: border-color 0.3s ease;
    }
    .form-group input:focus,
    .form-group textarea:focus {
        outline: none;
        border-color: #3498db;
    }
    .submit-btn {
        background-color: #3498db;
        color: white;
        border: none;
        padding: 15px;
    }

```

```

border-radius: 8px;
font-size: 1rem;
font-weight: 500;
cursor: pointer;
transition: all 0.3s ease;
}
.submit-btn:hover {
background-color: #2980b9;
transform: translateY(-2px);
}

```

Задания для выполнения лабораторной работы

Создать веб-страницу по вариантам с фиксированной шапкой, липкой панелью и модальным окном, используя различные типы CSS-позиционирования.

Обязательные требования для всех вариантов:

1. Использовать position: fixed для фиксированной шапки
2. Использовать position: sticky для липкой панели
3. Создать модальное окно с position: fixed для оверлея и position: absolute для внутренних элементов
4. Реализовать открытие/закрытие модального окна без JavaScript (через чекбокс)
5. Добавить минимум 2 кнопки открытия модального окна
6. Использовать z-index для управления наложением элементов
7. Добавить плавные переходы и анимации
8. Создать достаточно длинный контент для демонстрации прокрутки

Вариант 1. Сайт магазина электроники.

Вариант 2. Сайт фитнес-центра.

Вариант 3. Сайт туристического агентства.

Вариант 4. Сайт ресторана.

Вариант 5. Сайт строительной компании.

Вариант 6. Сайт языковой школы.

Вариант 7. Сайт фотостудии.

Вариант 8. Сайт автосервиса.

Вариант 9. Сайт юридической фирмы.

Вариант 10. Сайт клининговой компании.

Вариант 11. Сайт цветочного магазина.

Вариант 12. Сайт медицинского центра.

Вариант 13. Сайт дизайн-студии.

Вариант 14. Сайт образовательного центра с курсами.

Вариант 15. Сайт салона красоты.

Лабораторная работа № 13

«Верстка макета на Flexbox: шапка сайта и галерея изображений»

Цель работы: освоить практические навыки работы с CSS Flexbox для создания современных адаптивных макетов. Научиться использовать Flexbox для построения одномерных компоновок.

Методические указания

Flexbox (CSS Flexible Box Layout) – это модуль CSS, предназначенный для создания одномерных макетов, которые могут эффективно распределять пространство между элементами в контейнере, даже когда их размер неизвестен или динамичен. Основная идея Flexbox заключается в предоставлении контейнеру возможности изменять ширину/высоту своих элементов для наилучшего заполнения доступного пространства. Это особенно полезно для создания адаптивных интерфейсов, которые должны работать на устройствах с разными размерами экрана.

Flex-контейнер (Flex Container) – это HTML-элемент, у которого установлено свойство `display: flex` или `display: inline-flex`. Все непосредственные дочерние элементы этого контейнера автоматически становятся flex-элементами. Контейнер определяет основную ось и поперечную ось, которые являются фундаментальными для понимания работы Flexbox.

Flex-элементы (Flex Items) – это непосредственные дочерние элементы flex-контейнера. Эти элементы располагаются вдоль основной оси контейнера и могут растягиваться или сжиматься для заполнения доступного пространства. Каждый элемент может иметь свои собственные свойства, управляющие его поведением внутри контейнера.

Свойства flex-контейнера:

1. **display: flex** – превращает элемент в flex-контейнер блочного уровня. Элементы внутри него становятся flex-элементами.

2. **display: inline-flex** – превращает элемент в flex-контейнер строчного уровня. Внешнее поведение как у inline-элемента, но внутри работает как flex-контейнер.

3. **flex-direction** – определяет направление основной оси и, соответственно, расположение flex-элементов. Может принимать значения:

- row – слева направо
- row-reverse – справа налево
- column – сверху вниз
- column-reverse – снизу вверх

4. **justify-content** – определяет выравнивание элементов вдоль основной оси. Распределяет свободное пространство между элементами или вокруг них:

- flex-start – элементы прижимаются к началу оси
- flex-end – элементы прижимаются к концу оси
- center – элементы центрируются по оси
- space-between – первый элемент в начале, последний в конце, остальные равномерно распределены
- space-around – элементы равномерно распределены с равными отступами вокруг каждого
- space-evenly – элементы распределены так, чтобы расстояние между любыми двумя элементами было одинаковым

5. **align-items** – определяет выравнивание элементов вдоль поперечной оси. Управляет тем, как элементы располагаются перпендикулярно основной оси:

- stretch – элементы растягиваются, чтобы заполнить контейнер
- flex-start – элементы выравниваются к началу поперечной оси
- flex-end – элементы выравниваются к концу поперечной оси
- center – элементы центрируются по поперечной оси
- baseline – элементы выравниваются по базовой линии текста

6. **flex-wrap** – определяет, могут ли элементы переноситься на новую строку, если не хватает места в контейнере:

- nowrap – все элементы располагаются в одну строку
- wrap – элементы переносятся на несколько строк при необходимости
- wrap-reverse – элементы переносятся на несколько строк в обратном порядке

7. **align-content** – работает только когда есть несколько строк (при flex-wrap: wrap). Определяет выравнивание строк вдоль поперечной оси, аналогично тому, как justify-content выравнивает элементы вдоль основной оси.

Преимущества Flexbox для верстки:

- Простое выравнивание элементов по горизонтали и вертикали
- Автоматическое распределение доступного пространства
- Легкое создание адаптивных макетов без медиа-запросов
- Управление порядком элементов независимо от HTML-структуры
- Возможность создания равномерных сеток и колонок

Применение Flexbox в реальных задачах:

1. **Шапка сайта** – логотип слева, навигация справа, с адаптивным поведением
2. **Галерея изображений** – равномерная сетка с переносом элементов
3. **Карточки товаров** – выравнивание контента внутри карточек
4. **Формы** – выравнивание меток и полей ввода
5. **Навигационные меню** – горизонтальное или вертикальное расположение

Адаптивность с Flexbox: Flexbox сам по себе обладает адаптивными свойствами. Элементы автоматически перестраиваются при изменении размера контейнера. В сочетании с flex-wrap: wrap можно создавать сетки,

которые автоматически переносят элементы на новую строку при недостатке места, что идеально подходит для мобильных устройств.

Создадим веб-страницу с шапкой и галереей изображений.

Файл index.html:

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Flexbox: Шапка и галерея</title>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <header class="header"><!-- Шапка сайта -->
    <div class="container">
      <div class="logo">
        <span class="logo-text">FlexBox</span>
      </div>
      <nav class="nav">
        <a href="#" class="nav-link active">Главная</a>
        <a href="#" class="nav-link">Галерея</a>
        <a href="#" class="nav-link">О нас</a>
        <a href="#" class="nav-link">Контакты</a>
      </nav>
    </div>
  </header>
  <main class="main">
    <div class="container">
      <h1>Галерея изображений</h1>
      <p>Использование Flexbox для создания адаптивной
сетки</p>
      <div class="gallery"><!-- Галерея -->
        <div class="gallery-item">
          
          <h3>Изображение 1</h3>
```

```

        <p>Описание изображения</p>
    </div>
    <div class="gallery-item">
        
        <h3>Изображение 2</h3>
        <p>Описание изображения</p>
    </div>
    <div class="gallery-item">
        
        <h3>Изображение 3</h3>
        <p>Описание изображения</p>
    </div>
    <div class="gallery-item">
        
        <h3>Изображение 4</h3>
        <p>Описание изображения</p>
    </div>
    <div class="gallery-item">
        
        <h3>Изображение 5</h3>
        <p>Описание изображения</p>
    </div>
    <div class="gallery-item">
        
        <h3>Изображение 6</h3>
        <p>Описание изображения</p>
    </div>
</div>
</div>
</main>
</body>
</html>

```

В файле css:

```
* {
```

```

        margin: 0;
        padding: 0;
        box-sizing: border-box;
    }
    body {
        font-family: Arial, sans-serif;
        line-height: 1.6;
        color: #333;
    }
    .container {
        max-width: 1200px;
        margin: 0 auto;
        padding: 0 20px;
    }
    .header {
        background-color: white;
        box-shadow: 0 2px 10px rgba(0,0,0,0.1);
        padding: 20px 0;
    }
    .header .container { /* Контейнер шапки - flex-контейнер */
        display: flex; /* Включаем Flexbox */
        justify-content: space-between; /* Лого слева, меню справа */
        align-items: center; /* Выравниваем по вертикали */
    }
    .logo {
        font-size: 1.5rem;
        font-weight: bold;
        color: #2c3e50;
    }
    .nav {
        display: flex; /* Горизонтальное меню */
        gap: 20px; /* Отступы между ссылками */
    }
    .nav-link {

```

```

        color: #333;
        text-decoration: none;
        padding: 8px 16px;
        border-radius: 4px;
        transition: background-color 0.3s;
    }
    .nav-link:hover,
    .nav-link.active {
        background-color: #3498db;
        color: white;
    }
    .main {
        padding: 40px 0;
    }
    .main h1 {
        color: #2c3e50;
        margin-bottom: 10px;
    }
    .main > .container > p {
        color: #666;
        margin-bottom: 30px;
    }
    .gallery {
        display: flex; /* Flex-контейнер для галереи */
        flex-wrap: wrap; /* Разрешаем перенос элементов */
        gap: 20px; /* Отступы между элементами */
        margin-bottom: 40px;
    }
    .gallery-item {
        flex: 1 1 300px; /* Гибкие элементы */
        background-color: white;
        border-radius: 8px;
        overflow: hidden;
        box-shadow: 0 3px 10px rgba(0,0,0,0.1);
    }

```

```

}
.gallery-item img {
    width: 100%;
    height: 200px;
    object-fit: cover;
    display: block;
}
.gallery-item h3 {
    padding: 15px 15px 5px;
    color: #2c3e50;
}
.gallery-item p {
    padding: 0 15px 15px;
    color: #666;
}

```

Задания для выполнения лабораторной работы

Создать веб-сайт по вариантам с адаптивной шапкой с логотипом слева и меню справа. Построение равномерной сетки для галереи изображений с помощью flex-wrap.

Вариант 1. Сайт фотографа.

Вариант 2. Сайт интернет-магазина.

Вариант 3. Сайт рецептов.

Вариант 4. Сайт путешествий.

Вариант 5. Сайт художника.

Вариант 6. Сайт спортивного клуба.

Вариант 7. Сайт цветочного магазина.

Вариант 8. Сайт автосалона.

Вариант 9. Сайт библиотеки.

Вариант 10. Сайт музыкальной группы.

Вариант 11. Сайт кофейни.

Вариант 12. Сайт строительной фирмы.

Вариант 13. Сайт дизайн-студии.

Вариант 14. Сайт образовательного центра с курсами.

Вариант 15. Сайт салона красоты.

Лабораторная работа № 14

«Верстка сложного макета на Flexbox: карточка товара и «липкий» футер»

Цель работы: освоить продвинутые техники работы с CSS Flexbox для создания сложных адаптивных макетов. Научиться создавать детализированные карточки товаров с гибким расположением элементов, управлять отдельными flex-элементами с помощью свойств.

Методические указания

Управление отдельными flex-элементами – это ключевой аспект работы с Flexbox, который позволяет тонко настраивать поведение каждого элемента внутри flex-контейнера. В то время как свойства flex-контейнера управляют всеми элементами сразу, свойства отдельных flex-элементов позволяют изменять их поведение независимо от соседей.

Свойство **order** определяет порядок отображения flex-элементов внутри контейнера. По умолчанию все элементы имеют `order: 0`, что означает их отображение в том порядке, в котором они расположены в HTML-коде. Элементы с меньшим значением `order` отображаются раньше, с большим значением – позже. Например, если установить элементу `order: -1`, он переместится в самое начало, а при `order: 1` – в конец. Это свойство особенно полезно при создании адаптивных макетов, когда на мобильных устройствах нужно изменить порядок отображения элементов без изменения HTML-структуры.

Свойства **flex-grow**, **flex-shrink** и **flex-basis** образуют основу гибкости flex-элементов. `flex-basis` определяет начальный размер элемента до распределения свободного пространства. Может быть задан в абсолютных единицах (пикселях), относительных (процентах) или ключевым словом `auto`. `flex-grow` определяет, как элемент будет увеличиваться относительно других элементов при наличии свободного пространства. Значение 0 (по умолчанию) означает, что элемент не будет увеличиваться, а положительное число задает пропорцию, в которой элемент займет свободное пространство. `flex-shrink`

работает противоположно `flex-grow` – определяет, как элемент будет уменьшаться при нехватке пространства. Значение 0 предотвращает сжатие элемента, 1 (по умолчанию) позволяет сжиматься пропорционально.

Сокращенное свойство **flex** позволяет задать все три значения одновременно: `flex: <flex-grow> <flex-shrink> <flex-basis>`. Например, `flex: 1 0 200px` означает: элемент может увеличиваться (`flex-grow: 1`), не может уменьшаться (`flex-shrink: 0`), и имеет начальный размер 200px (`flex-basis: 200px`). Существуют также часто используемые сокращения: `flex: 1` (эквивалентно `flex: 1 1 0`), `flex: auto` (эквивалентно `flex: 1 1 auto`), `flex: none` (эквивалентно `flex: 0 0 auto`).

Свойство **align-self** позволяет переопределить выравнивание отдельного элемента по поперечной оси, установленное в контейнере через `align-items`. Это свойство принимает те же значения: `auto` (по умолчанию, наследует значение `align-items` контейнера), `flex-start`, `flex-end`, `center`, `baseline`, `stretch`. Например, если контейнер имеет `align-items: center`, но одному элементу нужно установить `align-self: flex-start`, этот элемент будет выровнен к началу поперечной оси, в то время как остальные останутся центрированными.

«Липкий» футер (**sticky footer**) – это техника верстки, при которой футер всегда остается внизу окна браузера, даже если контента на странице недостаточно для заполнения всей высоты экрана. При достаточном количестве контента футер располагается после него, как обычно. Для реализации этой техники с использованием Flexbox необходимо: установить `display: flex` и `flex-direction: column` для тега `body` или основного контейнера, задать основному контенту `flex: 1` (чтобы он занимал все доступное пространство), а футеру – `flex-shrink: 0` (чтобы он не сжимался). Это заставляет основной контент растягиваться, выталкивая футер вниз при малом количестве контента.

Верстка карточки товара с использованием Flexbox предполагает создание сложной внутренней структуры, где разные элементы должны вести себя по-разному. Например: изображение товара может иметь фиксированные

пропорции, заголовок и описание должны занимать доступное пространство, цена может быть выровнена по правому краю, а кнопка "Купить" – по нижнему краю карточки. Для этого внутри карточки создаются вложенные flex-контейнеры, а отдельным элементам назначаются специфичные flex-свойства.

Адаптивность карточек достигается комбинацией свойств **flex-wrap**, **flex-basis** с медиа-запросами. На десктопах карточки могут располагаться в несколько колонок, на планшетах – в две колонки, на мобильных – в одну. Изменение порядка элементов с помощью `order` позволяет оптимизировать отображение для мобильных устройств, например, переместить цену выше описания или кнопку в более удобное положение.

Выравнивание элементов внутри карточки часто требует комбинированного подхода. Для горизонтального распределения используется **justify-content** с значениями `space-between` или `space-around`. Для вертикального выравнивания – `align-items` или `align-self` для отдельных элементов. Особое внимание уделяется выравниванию кнопок действия, которые обычно должны быть прижаты к нижнему краю карточки независимо от высоты текстового содержимого.

Файл `index.html`:

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Карточка товара и липкий футер</title>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <div class="page-wrapper"><!-- Основной контейнер для липкого
футера -->
    <header class="header">
      <div class="container">
        <h1>Магазин электроники</h1>
```

```

        <p>Изучение Flexbox: карточка товара и липкий
футер</p>
    </div>
</header>
<main class="main-content"><!-- Основной контент -->
    <div class="container">
        <div class="product-card"><!-- Карточка товара --
>
            <div class="product-badge">Скидка 20%</div><!--
- Бейдж скидки -->
            <div class="product-image">
                
            </div>
            <div class="product-content"><!-- Контент
карточки -->
                <div class="product-header"><!-- Заголовок
и категория -->
                    <span class="product-
category">Смартфоны</span>
                    <h2 class="product-title">Смартфон XYZ
Pro 256GB</h2>
                </div>
                <div class="product-description"><!--
Описание товара -->
                    <p>Мощный смартфон с большим экраном,
тройной камерой и длительным временем работы от аккумулятора. Идеальный
выбор для работы и развлечений.</p>
                </div>
                <div class="product-features"><!--
Характеристики -->
                    <div class="feature">
                        <span class="feature-
label">Экран:</span>

```

```

        <span class="feature-value">6.7"
AMOLED</span>
    </div>
    <div class="feature">
        <span class="feature-
label">Память:</span>
        <span class="feature-value">256
ГБ</span>
    </div>
    <div class="feature">
        <span class="feature-
label">Камера:</span>
        <span class="feature-value">48 Мп
+ 12 Мп + 8 Мп</span>
    </div>
</div>
<div class="product-footer"><!-- Нижняя
часть карточки -->
    <div class="product-price">
        <span class="current-price">34 999
₽</span>
        <span class="old-price">43 749
₽</span>
    </div>
    <button class="buy-button">
        В корзину
    </button>
</div>
</div>
</div>
</div>
</main>
<footer class="footer"><!-- Липкий футер -->
    <div class="container">

```

<p>© 2023 Магазин электроники. Все права защищены.</p>

<p>Этот футер всегда остается внизу страницы</p>

</div>

</footer>

</div>

</body>

</html>

В файле css:

```
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}
body {
    font-family: Arial, sans-serif;
    line-height: 1.6;
    color: #333;
    background-color: #f8f9fa;
    min-height: 100vh; /* Минимальная высота - весь экран */
}
.container {
    max-width: 1200px;
    margin: 0 auto;
    padding: 0 20px;
}
.page-wrapper {
    display: flex; /* Делаем весь контейнер flex-контейнером */
    flex-direction: column; /* Располагаем элементы в колонку */
    min-height: 100vh; /* Минимальная высота - весь экран */
}
.main-content {
    flex: 1; /* Это заставляет контент растягиваться, выталкивая
футер вниз */
```

```

        padding: 40px 0;
    }
    .footer {
        background-color: #2c3e50;
        color: white;
        padding: 30px 0;
        text-align: center;
        flex-shrink: 0; /* НЕ ДАЕМ ФУТЕРУ СЖИМАТЬСЯ */
    }
    .header {
        background-color: white;
        box-shadow: 0 2px 10px rgba(0,0,0,0.1);
        padding: 20px 0;
    }
    .header h1 {
        color: #2c3e50;
        margin-bottom: 10px;
    }
    .header p {
        color: #666;
    }
    .product-card {
        background-color: white;
        border-radius: 12px;
        overflow: hidden;
        box-shadow: 0 5px 20px rgba(0,0,0,0.1);
        max-width: 800px;
        margin: 0 auto 40px;
        display: flex; /* Flex-контейнер для карточки */
        flex-direction: column; /* Элементы в колонку */
    }
    .product-badge {
        background-color: #e74c3c;
        color: white;
    }

```

```

padding: 8px 15px;
display: inline-block;
align-self: flex-start; /* ВЫРАВНИВАЕМ ОТДЕЛЬНЫЙ ЭЛЕМЕНТ */
margin: 15px;
border-radius: 4px;
font-weight: bold;
}
.product-image {
    height: 300px;
    overflow: hidden;
}
.product-image img {
    width: 100%;
    height: 100%;
    object-fit: cover;
}
.product-content {
    padding: 25px;
    display: flex; /* Flex-контейнер для контента */
    flex-direction: column; /* Элементы в колонку */
    flex: 1; /* ЗАНИМАЕТ ВСЕ ДОСТУПНОЕ ПРОСТРАНСТВО */
}
.product-header {
    margin-bottom: 15px;
}
.product-category {
    color: #7f8c8d;
    font-size: 0.9rem;
    display: block;
    margin-bottom: 5px;
}
.product-title {
    color: #2c3e50;
    font-size: 1.5rem;

```

```

}
.product-description {
    margin-bottom: 20px;
    flex-grow: 1; /* ОПИСАНИЕ РАСТЯГИВАЕТСЯ */
}
.product-description p {
    color: #666;
}
.product-features {
    margin-bottom: 25px;
    display: flex; /* Flex для характеристик */
    flex-wrap: wrap; /* Разрешаем перенос */
    gap: 15px;
}
.feature {
    flex: 1 1 200px; /* ГИБКИЕ ЭЛЕМЕНТЫ */
    background-color: #f8f9fa;
    padding: 10px 15px;
    border-radius: 6px;
    display: flex;
    justify-content: space-between;
}
.feature-label {
    color: #7f8c8d;
    font-weight: bold;
}
.feature-value {
    color: #2c3e50;
}
.product-footer {
    display: flex; /* Flex для цены и кнопки */
    justify-content: space-between; /* Цена слева, кнопка справа
*/
    align-items: center; /* Выравниваем по центру */

```

```

        margin-top: auto; /* ПРИЖИМАЕМ К НИЗУ КАРТОЧКИ */
        padding-top: 20px;
        border-top: 1px solid #eee;
    }
    .product-price {
        display: flex;
        flex-direction: column;
    }
    .current-price {
        font-size: 1.8rem;
        font-weight: bold;
        color: #e74c3c;
    }
    .old-price {
        color: #95a5a6;
        text-decoration: line-through;
        font-size: 1.2rem;
    }
    .buy-button {
        background-color: #3498db;
        color: white;
        border: none;
        padding: 12px 30px;
        border-radius: 6px;
        font-size: 1rem;
        font-weight: bold;
        cursor: pointer;
        transition: background-color 0.3s;
    }
    .buy-button:hover {
        background-color: #2980b9;
    }
}

```

Задания для выполнения лабораторной работы

Создать страницу с «липким» футером и карточкой товара/услуги, используя продвинутое свойство Flexbox.

Обязательные требования:

1. Реализовать «липкий» футер
2. Создать карточку с использованием flex-direction: column
3. Использовать flex: 1 для основного контента
4. Использовать flex-grow или flex-shrink для управления размерами
5. Выравнивание отдельных элементов

Вариант 1. Карточка книги.

Вариант 2. Карточка онлайн-курса.

Вариант 3. Карточка тура.

Вариант 4. Карточка рецепта.

Вариант 5. Карточка мероприятия.

Вариант 6. Карточка квартиры.

Вариант 7. Карточка фильма.

Вариант 8. Карточка услуги.

Вариант 9. Карточка игры.

Вариант 10. Карточка автомобиля.

Вариант 11. Карточка цветка.

Вариант 12. Карточка программы.

Вариант 13. Карточка мебели.

Вариант 14. Карточка подписки.

Вариант 15. Карточка спорта.

Лабораторная работа № 15

«Создание макета страницы с использованием CSS Grid (Часть 1)»

Цель работы: освоить принцип построения каркаса веб-страницы с помощью модуля CSS Grid Layout, научиться определять макет через именованные области.

Методические указания

CSS Grid Layout – это двумерная система компоновки, позволяющая создавать сложные макеты, управляя расположением элементов как по строкам, так и по столбцам. Контейнер, к которому применяется `display: grid`, становится сеточным контейнером, а его прямые дочерние элементы – сеточными элементами.

Свойство **grid-template-areas** позволяет задать именованные области сетки с помощью текстовых строк. Каждая строка в определении представляет собой строку сетки, а каждое имя (слово) в строке – ячейку столбца. Имена областей могут быть любыми. Повторение имени области в нескольких соседних ячейках объединяет их в одну прямоугольную область. Точка (.) обозначает пустую, незанятую ячейку.

После описания шаблона можно назначать элементы в эти области с помощью свойства **grid-area**, значением которого должно быть одно из заданных имен. Это позволяет управлять компоновкой на семантическом уровне, отделяя структуру макета от HTML-разметки.

Для управления размерами дорожек используются свойства **grid-template-rows** и **grid-template-columns**. В них можно задавать значения в пикселях, процентах и других единицах. Зазоры между элементами задаются свойствами `gap`.

Создадим веб-страницы с помощью модуля CSS Grid Layout.

Файл `index.html`:

```
<!DOCTYPE html>
<html lang="ru">
<head>
```

```

    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Лабораторная 15: CSS Grid макет</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <div class="container">
        <header>Шапка сайта</header>
        <aside>Боковая панель</aside>
        <main>Основной контент</main>
        <footer>Футер</footer>
    </div>
</body>
</html>

```

В файле css:

```

.container {
    display: grid; /* Включаем режим Grid */
    grid-template-areas:
        "header header" /* Первая строка: header занимает обе
колонки */
        "sidebar content" /* Вторая строка: sidebar слева, content
справа */
        "footer footer"; /* Третья строка: footer занимает обе
колонки */
    grid-template-columns: 200px 1fr; /* Левая колонка 200px,
правая - всё оставшееся пространство */
    grid-template-rows: auto 1fr auto; /* Первая и последняя строки
по высоте контента, средняя - всё доступное пространство */
    gap: 15px;
    min-height: 100vh;
    padding: 15px;
    background-color: #f5f5f5;
}

```

```

header {
    grid-area: header; /* Элемент займёт область с именем "header"
*/
    background-color: #3498db;
    color: white;
    padding: 20px;
    text-align: center;
}
aside {
    grid-area: sidebar; /* Элемент займёт область с именем
"sidebar" */
    background-color: #2ecc71;
    color: white;
    padding: 20px;
}
main {
    grid-area: content; /* Элемент займёт область с именем
"content" */
    background-color: #e74c3c;
    color: white;
    padding: 20px;
}
footer {
    grid-area: footer; /* Элемент займёт область с именем "footer"
*/
    background-color: #f39c12;
    color: white;
    padding: 20px;
    text-align: center;
}

```

Задания для выполнения лабораторной работы

Создать веб-сайт по вариантам.

Обязательные требования:

1. Использовать семантические HTML-теги (header, aside, main, footer и т.д.).
2. Применить к контейнеру display: grid.
3. Задать структуру макета с помощью grid-template-rows и grid-template-columns.
4. Обязательно использовать grid-template-areas для именования областей.
5. Привязать элементы к областям через grid-area.
6. Задать областям фоновые цвета и отступы для визуального различия.

Вариант 1. Блог путешественника.

Вариант 2. Сайт кинотеатра.

Вариант 3. Сайт пиццерии.

Вариант 4. Сайт-портфолио архитектора.

Вариант 5. Сайт учебного курса.

Вариант 6. Новостной портал.

Вариант 7. Сайт кофейни.

Вариант 8. Сайт юридической фирмы.

Вариант 9. Галерея современного искусства.

Вариант 10. Сайт музыкальной группы.

Вариант 11. Сайт библиотеки.

Вариант 12. Сайт цветочного магазина.

Вариант 13. Сайт футбольной команды.

Вариант 14. Сайт ресторана.

Вариант 15. Блог дизайнера интерьера.

Лабораторная работа № 16

«Создание макета страницы с использованием CSS Grid (Часть 2)»

Цель работы: освоить создание каскадной сетки, характерные для портфолио или блогов.

Методические указания

Для создания сложных, неравномерных сеток в CSS Grid используются свойства размещения элементов: **grid-column** и **grid-row**. Они определяют, где элемент начинается и заканчивается внутри сетки. Эти свойства могут задаваться в нескольких форматах.

Masonry-эффект - расположение элементов разной высоты в плотную каскадную сетку, как в кирпичной кладке. Наиболее полезным для создания masonry-эффекта является использование ключевого слова **span**. Запись **grid-column: span X** означает, что данный элемент должен растянуться на указанное количество колонок сетки, начиная с текущей позиции автоматического размещения.

Аналогично работает **grid-row: span Y**. Система автоматического размещения Grid будет заполнять пустые ячейки, что позволяет создавать плотные макеты без явного указания начальной линии для каждого элемента. В сочетании с **grid-auto-rows: minmax(100px, auto)**, которая задает минимальную высоту для автоматически создаваемых строк, можно добиться эффекта, когда блоки разной высоты занимают разное количество строк, формируя «рваный» правый край, характерный для портфолио или блога.

Создадим каскадную сетку. В области `.content` заменим содержимое на сетку карточек:

```
<main class="content">
  <div class="portfolio">
    <div class="card card--tall">Карточка 1 (высокая)</div>
    <div class="card">Карточка 2</div>
    <div class="card card--wide">Карточка 3 (широкая)</div>
    <!-- еще 7-10 карточек с разными классами -->
  </div>
```

</main>

В файле css зададим контейнеру .portfolio свойства Grid. Определите столбцы и автоматические строки:

```
.portfolio {  
    display: grid;  
    grid-template-columns: repeat(4, 1fr); /* 4 колонки */  
    grid-auto-rows: 120px; /* Базовая высота строки */  
    gap: 20px;  
}
```

Создадим стили для карточек, используя свойства span для изменения их размера в сетке

```
.card {  
    background: lightblue;  
    padding: 15px;  
    border-radius: 8px;  
}  
  
.card--tall {  
    grid-row: span 2; /* Займет 2 строки */  
    background: lightcoral;  
}  
  
.card--wide {  
    grid-column: span 2; /* Займет 2 колонки */  
    background: lightgreen;  
}  
  
.card--large {  
    grid-column: span 2;  
    grid-row: span 2;  
    background: gold;  
}
```

Задания для выполнения лабораторной работы

Создать веб-сайт, используя макет из лабораторной работы №15, сверстайте внутри области основного контента тематическую сетку,

подходящую для вашего сайта. Заполните карточки условным контентом (изображения-заглушки, заголовки)

Обязательные требования:

1. Макет должен быть выполнен как продолжение работы по вашему варианту из лабораторной работы №15.
2. В области, отведенной под основной контент (main или content), заменить простой блок на сетку портфолио/галереи/ленты.
3. Сетка должна иметь контейнер с `display: grid`.
4. Обязательно использовать `grid-template-columns` с `repeat()`.
5. Использовать `grid-auto-rows` для задания высоты строк.
6. Использовать `gap` для промежутков.
7. Создать не менее 4 элементов (карточек) внутри этой сетки.
8. Для управления размерами карточек обязательно применить `grid-column: span X` и/или `grid-row: span Y` минимум к 2 разным карточкам.

Вариант 1. Блог путешественника. Создайте ленту фотографий из поездок галерею.

Вариант 2. Сайт кинотеатра. Создайте сетку постеров фильмов, идущих в прокате.

Вариант 3. Сайт пиццерии. Создайте сетку блюд.

Вариант 4. Сайт-портфолио архитектора. Создайте портфолио проектов (masonry-галерею).

Вариант 5. Сайт учебного курса. Создайте сетку модулей или уроков курса.

Вариант 6. Новостной портал. Создайте ленту новостных карточек.

Вариант 7. Сайт кофейни. Создайте сетку напитков.

Вариант 8. Сайт юридической фирмы. Создайте сетку успешных кейсов или юристов компании.

Вариант 9. Галерея современного искусства. Создайте нерегулярную сетку экспонатов (masonry).

Вариант 10. Сайт музыкальной группы. Создайте сетку обложек музыкальных альбомов.

Вариант 11. Сайт библиотеки. Создайте сетку книг.

Вариант 12. Сайт цветочного магазина. Создайте сетку букетов.

Вариант 13. Сайт футбольной команды. Создайте сетку победных матчей.

Вариант 14. Сайт ресторана. Создайте сетку популярных блюд

Вариант 15. Блог дизайнера интерьера. Создайте портфолио проектов (masonry-галерею).

Лабораторная работа № 17

«Адаптивная верстка: применение медиа-запросов (@media)»

Цель работы: освоить практические навыки работы с CSS медиа-запросами (@media) для создания адаптивных веб-макетов. Научиться адаптировать существующие сеточные макеты под различные устройства, изменяя расположение блоков, скрывая элементы и корректируя размеры.

Методические указания

Медиа-запросы (Media Queries) – это модуль CSS, предназначенный для создания адаптивных макетов, которые могут динамически подстраиваться под характеристики устройства пользователя. Основная идея медиа-запросов заключается в предоставлении разработчику возможности применять различные CSS-правила в зависимости от условий, таких как ширина области просмотра, высота, ориентация или разрешение устройства. Это особенно полезно для создания интерфейсов, которые должны корректно отображаться на устройствах с разными размерами экрана – от десктопных мониторов до мобильных телефонов.

Область просмотра (Viewport) – это видимая пользователю область веб-страницы. Ширина области просмотра (viewport width) является ключевым параметром для создания адаптивных макетов. Медиа-запросы позволяют реагировать на изменения этой ширины и перестраивать компоновку страницы соответствующим образом.

Медиа-запрос (Media Query) – это CSS-правило, которое применяет стили только при выполнении определенных условий. Синтаксис медиа-запроса начинается с директивы @media, за которой следует условие в круглых скобках и блок CSS-правил в фигурных скобках. Наиболее часто используются условия, связанные с шириной области просмотра.

Точка останова (Breakpoint) – это конкретное значение ширины (или другого параметра), при котором происходит изменение макета. Точки останова определяют, когда и как должна изменяться верстка при разных размерах экрана.

Основные свойства и значения для адаптации макетов:

1. Создание медиа-запроса – используется правило **@media** с условиями:

- `max-width: 768px` – применяет стили при ширине окна 768px или меньше
- `min-width: 769px` – применяет стили при ширине окна 769px или больше
- `orientation: portrait` – применяет стили при портретной ориентации
- `orientation: landscape` – применяет стили при альбомной ориентации

2. Изменение структуры **Grid** – внутри медиа-запросов можно переопределять свойства Grid-контейнера:

- `grid-template-areas` – изменить схему расположения областей
- `grid-template-columns` – изменить количество и размеры колонок
- `grid-template-rows` – изменить количество и размеры строк
- `display` – изменить способ отображения контейнера (например, на `block`)

3. Скрытие элементов – для неважных на мобильных устройствах блоков:

- `display: none` – полностью скрывает элемент
- `visibility: hidden` – делает элемент невидимым, но он занимает место
- `opacity: 0` – делает элемент прозрачным, но он остается интерактивным

4. Корректировка типографики и размеров:

- `font-size` – уменьшение размеров шрифта для мобильных устройств
- `padding` и `margin` – корректировка внутренних и внешних отступов
- `width` и `height` – изменение размеров элементов

Для адаптации веб-страницы под мобильное устройство добавляется медиазапрос для планшетов и мобильных устройств в файл CSS:

```
@media (max-width: 768px) {  
    .container { /* Меняем структуру Grid на одноколоночную */  
        grid-template-columns: 1fr; /* Одна колонка */  
        grid-template-areas:  
            "header"  
            "sidebar"  
            "content"  
            "footer"; /* Все области в столбик */  
    }  
    .header { /* Уменьшаем размер шрифта в шапке для мобильных */  
        font-size: 18px;  
        padding: 15px;  
    }  
    .sidebar { /* Оптимизируем сайдбар для мобильных */  
        padding: 10px;  
        border-bottom: 2px solid #dee2e6;  
    }  
    .content { /* Корректируем контент для удобства чтения */  
        font-size: 14px;  
        line-height: 1.5;  
    }  
    .footer .secondary-info { /* Скрываем неважный элемент в футере */  
        display: none;  
    }  
}
```

Задания для выполнения лабораторной работы

Адаптируйте веб-сайт, используя макет из лабораторной работы №15, для мобильных устройств (ширина экрана $\leq 768\text{px}$).

Вариант 1. Блог путешественника.

Вариант 2. Сайт кинотеатра.

- Вариант 3.** Сайт пиццерии.
- Вариант 4.** Сайт-портфолио архитектора.
- Вариант 5.** Сайт учебного курса.
- Вариант 6.** Новостной портал.
- Вариант 7.** Сайт кофейни.
- Вариант 8.** Сайт юридической фирмы.
- Вариант 9.** Галерея современного искусства.
- Вариант 10.** Сайт музыкальной группы.
- Вариант 11.** Сайт библиотеки.
- Вариант 12.** Сайт цветочного магазина.
- Вариант 13.** Сайт футбольной команды.
- Вариант 14.** Сайт ресторана.
- Вариант 15.** Блог дизайнера интерьера

Лабораторная работа № 18

«Создание адаптивной навигации (бургер-меню)»

Цель работы: освоить практические навыки создания полностью адаптивного меню навигации с использованием чистого CSS.

Методические указания

Бургер-меню – это адаптивный паттерн навигации, представляющий собой скрытое меню, которое активируется при нажатии на иконку из трех горизонтальных линий. Основная идея бургер-меню заключается в экономии экранного пространства на мобильных устройствах путем скрывания элементов навигации до момента, когда пользователь явно запросит их отображение. Это особенно полезно для создания удобных интерфейсов, которые должны корректно работать на устройствах с ограниченным размером экрана – от десктопных мониторов до смартфонов.

Техника чекбокса для меню – это CSS-метод использования элемента `<input type="checkbox">` для хранения и изменения состояния меню. CSS позволяет стилизовать элементы на основе состояния чекбокса с помощью селектора `:checked`. Основной принцип: чекбокс скрыт с помощью `display: none` или других методов, а его состояние управляет отображением меню через CSS-правила.

Основные компоненты адаптивного меню:

1. HTML-структура – состоит из трех ключевых элементов:
 - `<input type="checkbox" id="menu-toggle">` – скрытый чекбокс для управления состоянием
 - `<label for="menu-toggle" class="burger-btn">` – видимая кнопка-триггер (иконка бургера)
 - `<nav class="menu">` – контейнер с элементами навигации
2. Принцип работы чекбокса – связь между чекбоксом и лейблом осуществляется через атрибут `for` у лейбла, который должен соответствовать `id` чекбокса. При клике на лейбл состояние чекбокса меняется, что позволяет применять CSS-правила для селектора `#menu-toggle:checked ~ .menu`.

3. CSS-селекторы для управления состоянием:

- `#menu-toggle:checked ~ .menu` – применяет стили к меню, когда чекбокс отмечен
- `#menu-toggle:not(:checked) ~ .menu` – применяет стили, когда чекбокс не отмечен
- `~` (тильда) – селектор общего соседнего элемента, выбирает все элементы `.menu`, которые находятся после `#menu-toggle` на одном уровне вложенности

4. Создание иконки бургера – реализуется с помощью псевдоэлементов `::before` и `::after` у лейбла. Три линии создаются: одна – самим элементом лейбла (через `border` или `background`), и две – через псевдоэлементы. Анимация превращения в крестик достигается поворотом и трансформацией этих линий при отмеченном состоянии чекбокса.

5. Адаптивная логика – с помощью медиа-запросов реализуется переключение между двумя состояниями:

- На десктопах (широких экранах) – горизонтальное меню всегда видимо, бургер-кнопка скрыта
- На мобильных устройствах (узких экранах) – горизонтальное меню скрыто, отображается бургер-кнопка, меню появляется при клике

Ключевые CSS-свойства для реализации:

1. Управление видимостью меню:

- `display: none/block/flex` – полное скрытие/отображение
- `transform: translateX(-100%)` – выезд сбоку
- `opacity: 0/1` – плавное появление
- `visibility: hidden/visible` – управление видимостью с сохранением layout

2. Стилизация бургер-кнопки:

- `cursor: pointer` – указание на интерактивность
- `position: relative/absolute` – позиционирование для псевдоэлементов

- transition – плавные анимации преобразования
- transform: rotate() – поворот линий в крестик

Создадим полностью адаптивное меню навигации

в файле html:

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Бургер-меню</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <header class="header">
    <div class="logo">Логотип</div>
    <input type="checkbox" id="menu-toggle"><!-- Скрытый
чекбокс для управления меню -->
    <label for="menu-toggle" class="burger"><!-- Кнопка-бургер
-->

      <span></span>
      <span></span>
      <span></span>
    </label>
    <nav class="menu"><!-- Основное меню -->
      <a href="#">Главная</a>
      <a href="#">О нас</a>
      <a href="#">Услуги</a>
      <a href="#">Контакты</a>
    </nav>
  </header>
  <main class="content">
    <h1>Контент страницы</h1>
```

<p>При уменьшении ширины окна до 768px горизонтальное меню скроется и появится бургер-иконка.</p>

```
</main>
```

```
</body>
```

```
</html>
```

в файле css:

```
* {  
    margin: 0;  
    padding: 0;  
    box-sizing: border-box;  
}  
body {  
    font-family: Arial, sans-serif;  
}  
.header { /* Шапка и десктопное меню */  
    display: flex;  
    justify-content: space-between;  
    align-items: center;  
    padding: 20px;  
    background: #2c3e50;  
    color: white;  
}  
.logo {  
    font-size: 24px;  
    font-weight: bold;  
}  
.menu {  
    display: flex;  
    gap: 20px;  
}  
.menu a {  
    color: white;  
    text-decoration: none;  
    padding: 10px;
```

```

}
.menu a:hover {
    background: #34495e;
    border-radius: 4px;
}
#menu-toggle,/* Скрываем чекбокс и бургер на десктопе */
.burger {
    display: none;
}
.content {/* Контент */
    padding: 40px 20px;
    max-width: 800px;
    margin: 0 auto;
}
.content h1 {
    margin-bottom: 20px;
}
@media (max-width: 768px) {/* Мобильная версия */
    .burger {/* Показываем бургер */
        display: block;
        width: 30px;
        height: 21px;
        position: relative;
        cursor: pointer;
        z-index: 10;
    }
    .burger span {/* Три линии бургера */
        display: block;
        height: 3px;
        width: 100%;
        background: white;
        position: absolute;
        transition: 0.3s;
    }
}

```

```

.burger span:nth-child(1) {
    top: 0;
}
.burger span:nth-child(2) {
    top: 9px;
}
.burger span:nth-child(3) {
    top: 18px;
}
#menu-toggle:checked ~ .burger span:nth-child(1) { /*
Превращение в крестик */
    transform: rotate(45deg);
    top: 9px;
}
#menu-toggle:checked ~ .burger span:nth-child(2) {
    opacity: 0;
}
#menu-toggle:checked ~ .burger span:nth-child(3) {
    transform: rotate(-45deg);
    top: 9px;
}
.menu { /* Мобильное меню */
    position: fixed;
    top: 0;
    right: -100%;
    width: 250px;
    height: 100vh;
    background: #2c3e50;
    flex-direction: column;
    justify-content: center;
    align-items: center;
    gap: 30px;
    transition: 0.3s;
    z-index: 9;
}

```

```

    }
    #menu-toggle:checked ~ .menu {/* Показываем меню при
активированном чекбоксе */
        right: 0;
    }
    .menu a {
        font-size: 18px;
        width: 80%;
        text-align: center;
        padding: 15px;
    }
}

```

Задания для выполнения лабораторной работы

Создайте адаптивное бургер-меню для сайта вашего варианта из лабораторной работы №15, для мобильных устройств.

Вариант 1. Блог путешественника.

Вариант 2. Сайт кинотеатра.

Вариант 3. Сайт пиццерии.

Вариант 4. Сайт-портфолио архитектора.

Вариант 5. Сайт учебного курса.

Вариант 6. Новостной портал.

Вариант 7. Сайт кофейни.

Вариант 8. Сайт юридической фирмы.

Вариант 9. Галерея современного искусства.

Вариант 10. Сайт музыкальной группы.

Вариант 11. Сайт библиотеки.

Вариант 12. Сайт цветочного магазина.

Вариант 13. Сайт футбольной команды.

Вариант 14. Сайт ресторана.

Вариант 15. Блог дизайнера интерьер

Лабораторная работа № 19

«Создание адаптивного макета с использованием CSS-фреймворка (на примере локального Bootstrap)»

Цель работы: освоить практические навыки работы с CSS-фреймворком Bootstrap для быстрого создания адаптивных макетов.

Методические указания

Bootstrap – это популярный CSS-фреймворк с открытым исходным кодом, предназначенный для разработки адаптивных и мобильных веб-проектов. Основная идея Bootstrap заключается в предоставлении готовых инструментов и компонентов, которые позволяют быстро создавать современные интерфейсы без необходимости писать CSS с нуля. Фреймворк включает в себя мощную сеточную систему, готовые компоненты (кнопки, формы, навигацию) и JavaScript-плагины.

Сеточная система Bootstrap – это основа фреймворка, представляющая собой 12-колоночную адаптивную сетку. Система использует контейнеры, строки и колонки для создания макетов любой сложности. Ключевой принцип: контейнер оборачивает все содержимое, строки группируют колонки горизонтально, а колонки содержат непосредственно контент и определяют его ширину на разных устройствах (Рис 19.1).

Span 1	Span 1	Span 1	Span 1	Span 1	Span 1	Span 1	Span 1	Span 1	Span 1	Span 1	Span 1
Span 12											
Span 2	Span 2		Span 2		Span 2		Span 2		Span 2		
Span 3			Span 3			Span 3			Span 3		
Span 4				Span 4				Span 4			
Span 6						Span 6					
Span 4				Span 8							
Span 7							Span 5				
Span 10										Span 2	
Span 11											Span 1
Span 4				Span 3			Span 2		Span 3		

Рисунок 19.1 – Сеточная система Bootstrap.

Адаптивные брейкпоинты Bootstrap – это заранее определенные точки останова, которые определяют, как макет будет вести себя на разных устройствах:

- .col- – экстремальные устройства (<576px)
- .col-sm- – маленькие устройства (≥576px)
- .col-md- – средние устройства (≥768px)
- .col-lg- – большие устройства (≥992px)
- .col-xl- – экстрабольшие устройства (≥1200px)

Контейнеры (Containers) – это основные строительные блоки Bootstrap, которые выравнивают и ограничивают ширину содержимого. Существует два типа контейнеров:

- .container – контейнер с фиксированной максимальной шириной, которая изменяется на каждом брейкпоинте
- .container-fluid – контейнер на всю ширину окна браузера

Строки (Rows) – это горизонтальные группы колонок, которые должны быть размещены внутри контейнера. Строки используют технологию flexbox для выравнивания колонок и гарантируют, что колонки будут правильно расположены. Класс `.row` создает горизонтальную группу, которая содержит колонки и обеспечивает отрицательные отступы для компенсации отступов колонок.

Колонки (Columns) – это непосредственные дочерние элементы строк, которые определяют ширину контента. Каждая колонка имеет горизонтальные отступы (`padding`) для создания промежутков между контентными блоками. Классы колонок указывают количество из 12 доступных колонок, которые должен занимать элемент на разных брейкпоинтах. Например, `.col-md-4` означает "на устройствах среднего размера и больше займи 4 из 12 колонок".

Для работы с Bootstrap необходимо скачать с официального сайта getbootstrap.com версию Bootstrap 5 и разархивировать файлы. В папке проекта создать папку `css`. Файл `bootstrap.min.css` переместить в папку `css/`.

Создадим веб-сайт с помощью фреймворка Bootstrap.

в файле `html`:

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Лендинг на Bootstrap</title>
  <link rel="stylesheet" href="css/bootstrap.min.css"><!--
Подключение локального Bootstrap CSS -->
</head>
<body class="d-flex flex-column min-vh-100">
<main class="flex-grow-1">
  <section class="section-1 py-5"><!-- секция с сеткой 1:1 -->
    <div class="container">
      <div class="row align-items-center">
```

```

        <div class="col-md-6">
            <h1 class="mb-4">Основной заголовок</h1>
            <p class="lead">Текст первого блока, который
будет адаптироваться под разные размеры экрана.</p>
        </div>
        <div class="col-md-6">
            <div class="placeholder-box">
                <p>Второй блок контента</p>
            </div>
        </div>
    </div>
</div>
</section>
<section class="section-2 py-5 bg-light">
    <div class="container">
        <h2 class="text-center mb-5">Секция с тремя
колонками</h2>
        <div class="row">
            <div class="col-lg-3 col-md-6 mb-4">
                <div class="placeholder-box">
                    <p>Колонка 1 (3 из 12 на больших экранах,
6 из 12 на средних)</p>
                </div>
            </div>
            <div class="col-lg-6 col-md-12 mb-4">
                <div class="placeholder-box">
                    <p>Центральная колонка (6 из 12 на больших
экранах, 12 из 12 на средних)</p>
                </div>
            </div>
            <div class="col-lg-3 col-md-6 mb-4">
                <div class="placeholder-box">
                    <p>Колонка 3 (3 из 12 на больших экранах,
6 из 12 на средних)</p>
                </div>
            </div>
        </div>
    </div>

```

```

        </div>
    </div>
</div>
</div>
</section>
<section class="section-4 py-5 bg-light">
    <div class="container">
        <h2 class="text-center mb-5">Секция с вложенной
сеткой</h2>
        <div class="row">
            <div class="col-lg-8">
                <div class="placeholder-box p-4 mb-4">
                    <h3>Основной контент (8 колонок)</h3>
                    <div class="row mt-3">
                        <div class="col-md-6 mb-3">
                            <div class="nested-box">
                                <p>Вложенный блок 1</p>
                            </div>
                        </div>
                        <div class="col-md-6 mb-3">
                            <div class="nested-box">
                                <p>Вложенный блок 2</p>
                            </div>
                        </div>
                    </div>
                </div>
            </div>
            <div class="col-lg-4">
                <div class="placeholder-box p-4">
                    <h3>Сайдбар (4 колонки)</h3>
                    <p>На мобильных устройствах будет под
основным контентом</p>
                </div>
            </div>
        </div>
    </div>

```

```

        </div>
    </div>
</section>
</main>
<footer class="footer bg-dark text-white py-4 mt-auto">
    <div class="container">
        <div class="row align-items-center">
            <div class="col-md-6 mb-3 mb-md-0">
                <p class="mb-0">&copy; 2024 МойЛендинг. Все права
защищены.</p>
            </div>
            <div class="col-md-6 text-md-end">
                <span class="d-block d-md-inline me-md-3 mb-2 mb-
md-0">Контакты: info@example.com</span>
                <span class="d-block d-md-inline">Телефон: +7
(123) 456-78-90</span>
            </div>
        </div>
    </div>
</footer>
</body>
</html>

```

Задания для выполнения лабораторной работы

Создайте лендинг с помощью Bootstrap, подключенный локально, для сайта вашего варианта из предыдущих работ. Используйте контейнер Bootstrap классы для сетки, минимум 3 строки с колонками, применить разные классы колонок. Макет должен быть адаптивным и корректно отображаться на мобильных устройствах.

Вариант 1. Блог путешественника.

Вариант 2. Сайт кинотеатра.

Вариант 3. Сайт пиццерии.

Вариант 4. Сайт-портфолио архитектора.

Вариант 5. Сайт учебного курса.

- Вариант 6.** Новостной портал.
- Вариант 7.** Сайт кофейни.
- Вариант 8.** Сайт юридической фирмы.
- Вариант 9.** Галерея современного искусства.
- Вариант 10.** Сайт музыкальной группы.
- Вариант 11.** Сайт библиотеки.
- Вариант 12.** Сайт цветочного магазина.
- Вариант 13.** Сайт футбольной команды.
- Вариант 14.** Сайт ресторана.
- Вариант 15.** Блог дизайнера интерьер

Лабораторная работа № 20

«Разработка пользовательского интерфейса с использованием готовых CSS-компонентов»

Цель работы: освоить практические навыки работы с готовыми CSS-компонентами Bootstrap для быстрого создания интерфейсов. Научиться кастомизировать стандартные компоненты через CSS-переменные и дополнительные классы.

Методические указания

Готовые CSS-компоненты – это предварительно стилизованные элементы интерфейса, которые можно быстро интегрировать в веб-проект. В Bootstrap эти компоненты построены на основе семантической HTML-разметки с соответствующими CSS-классами. Основная идея использования готовых компонентов заключается в значительном ускорении процесса разработки за счет повторного использования проверенных решений, которые уже включают адаптивность, доступность и кроссбраузерную совместимость. Это особенно полезно для создания профессиональных интерфейсов без необходимости написания сложного CSS с нуля для каждого элемента.

CSS-переменные в Bootstrap – это динамические значения, которые хранятся в CSS и могут быть использованы многократно по всему документу. Bootstrap 5 использует CSS-переменные для управления темами, цветами, отступами, размерами шрифтов и другими параметрами. Основное преимущество CSS-переменных – возможность их переопределения в любом месте каскада, что позволяет легко кастомизировать внешний вид компонентов без изменения исходного кода Bootstrap. Например, переменная **--bs-primary** определяет основной цвет темы, и изменив ее значение, можно поменять цвет всех элементов, которые используют этот цвет.

Кастомизация компонентов Bootstrap может выполняться двумя основными способами. Первый способ – переопределение CSS-переменных Bootstrap в корневом селекторе (:root) или в любом другом селекторе. Это позволяет глобально изменить тему приложения, поменяв значения

переменных, которые затем автоматически применяются ко всем компонентам. Второй способ – добавление собственных CSS-классов к HTML-элементам Bootstrap и написание дополнительных стилей для них. Этот подход позволяет создавать модификации стандартных компонентов, не затрагивая их базовую функциональность, и применяется когда нужно изменить внешний вид только конкретных экземпляров компонентов.

Создадим полностью адаптивное меню навигации с готовыми компонентами.

в файле html:

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Кастомизация Bootstrap</title>
  <!-- Подключение Bootstrap локально -->
  <link rel="stylesheet" href="css/bootstrap.min.css">
  <link rel="stylesheet" href="css/style.css">
</head>
<body>
<nav class="navbar navbar-expand-lg navbar-custom">
  <div class="container">
    <a class="navbar-brand" href="#">
      <span class="brand-icon">
```

```

        <a                class="nav-link                active"
href="#">Главная</a>
    </li>
    <li class="nav-item">
        <a class="nav-link" href="#">О нас</a>
    </li>
    <li class="nav-item">
        <a class="nav-link" href="#">Услуги</a>
    </li>
    <li class="nav-item">
        <a class="nav-link" href="#">Контакты</a>
    </li>
</ul>
</div>
</div>
</nav>
<main class="container my-5">
    <div class="row g-4">
        <div class="col-md-4">
            <div class="card custom-card">
                <div class="card-header">
                    <h5                class="card-title                mb-0">Стандартная
карточка</h5>
                </div>
                <div class="card-body">
                    <p class="card-text">Эта карточка использует
кастомизированные CSS-переменные Bootstrap.</p>
                    <button                class="btn                btn-primary">Основная
кнопка</button>
                </div>
            </div>
        </div>
    </div>
</div>
<div class="row mt-5"><!-- Разные типы кнопок -->

```

```

    <div class="col-12">
      <h3 class="mb-4">Кастомизированные кнопки</h3>
      <div class="d-flex flex-wrap gap-3">
        <button class="btn btn-primary">Основная</button>
        <button class="btn btn-secondary">Вторичная</button>
        <button class="btn btn-success">Успех</button>
        <button class="btn btn-danger">Опасность</button>
        <button class="btn btn-warning">Предупреждение</button>
        <button class="btn btn-info">Информация</button>
        <button class="btn btn-custom">Своя
кнопка</button>
        <button class="btn btn-outline-
primary">Контурная</button>
      </div>
    </div>
  </div>
</main>
</body>
</html>

```

в файле CSS для кастомизации:

```

:root { /* Переопределение CSS-переменных Bootstrap */
  --bs-primary: #6f42c1;
  --bs-font-sans-serif: 'Segoe UI', system-ui, -apple-system,
sans-serif;
  --bs-body-font-size: 1rem;
  --bs-body-font-weight: 400;
  --bs-body-line-height: 1.6;
  --bs-border-radius: 12px;
  --bs-border-radius-lg: 16px;
  --bs-border-radius-sm: 8px;
  --bs-spacer: 1.5rem;
}

```

```

.navbar-custom { /* Кастомизация навигационной панели */
    background: linear-gradient(135deg, var(--bs-primary) 0%,
var(--bs-secondary) 100%);
    box-shadow: 0 2px 10px rgba(0,0,0,0.1);
}
.navbar-custom .navbar-brand {
    font-weight: 700;
    font-size: 1.5rem;
    color: white !important;
}
.navbar-custom .navbar-brand .brand-icon {
    margin-right: 8px;
}
.navbar-custom .nav-link {
    color: rgba(255,255,255,0.9) !important;
    font-weight: 500;
    padding: 0.5rem 1rem !important;
    border-radius: var(--bs-border-radius-sm);
    transition: all 0.3s ease;
}
.navbar-custom .nav-link:hover,
.navbar-custom .nav-link.active {
    color: white !important;
    background-color: rgba(255,255,255,0.1);
}
.btn-custom { /* Собственный класс для кнопки */
    --bs-btn-color: #fff;
    --bs-btn-bg: #6c757d;
    --bs-btn-border-color: #6c757d;
    --bs-btn-hover-color: #fff;
    --bs-btn-hover-bg: #5c636a;
    --bs-btn-hover-border-color: #565e64;
    --bs-btn-focus-shadow-rgb: 130, 138, 145;
    --bs-btn-active-color: #fff;

```

```

--bs-btn-active-bg: #565e64;
--bs-btn-active-border-color: #51585e;
--bs-btn-active-shadow: inset 0 3px 5px rgba(0, 0, 0, 0.125);
--bs-btn-disabled-color: #fff;
--bs-btn-disabled-bg: #6c757d;
--bs-btn-disabled-border-color: #6c757d;
}
@media (max-width: 768px) {
  .navbar-custom .navbar-nav {
    margin-top: 1rem;
    padding: 1rem;
    background-color: rgba(255,255,255,0.1);
    border-radius: var(--bs-border-radius);
  }
  .custom-card {
    margin-bottom: 1.5rem;
  }
}

```

Задания для выполнения лабораторной работы

Создайте страницу с кастомизированными компонентами Bootstrap для вашего варианта. Измените CSS-переменные и добавьте свои классы для стилизации.

Вариант 1. Блог путешественника.

Вариант 2. Сайт кинотеатра.

Вариант 3. Сайт пиццерии.

Вариант 4. Сайт-портфолио архитектора.

Вариант 5. Сайт учебного курса.

Вариант 6. Новостной портал.

Вариант 7. Сайт кофейни.

Вариант 8. Сайт юридической фирмы.

Вариант 9. Галерея современного искусства.

Вариант 10. Сайт музыкальной группы.

Вариант 11. Сайт библиотеки.

Вариант 12. Сайт цветочного магазина.

Вариант 13. Сайт футбольной команды.

Вариант 14. Сайт ресторана.

Вариант 15. Блог дизайнера интерьер.

Лабораторная работа № 21

«Практическая работа: верстка макета главной страницы по готовому шаблону (этап 1 - HTML)»

Цель работы: освоить практические навыки преобразования графического макета в семантическую HTML-структуру.

Методические указания

Семантическая разметка – использование HTML-тегов, которые несут смысловую нагрузку и описывают содержание контейнера. Это улучшает читаемость кода, SEO и доступность.

Основные семантические теги HTML5:

- <header> – шапка сайта или раздела.
- <nav> – навигационное меню.
- <main> – основное содержание страницы.
- <section> – логический раздел документа.
- <article> – самостоятельный, независимый контент.
- <aside> – боковая панель, дополнительный контент.
- <footer> – подвал сайта или раздела.
- <figure> и <figcaption> – иллюстрации с подписями.

Методология именования классов БЭМ (Блок-Элемент-Модификатор):

- **Блок** – независимый компонент (header, menu).
- **Элемент** – часть блока (header__logo, menu__item).
- **Модификатор** – свойство блока или элемента (menu__item--active).

Подготовка рабочей среды

1. Создайте папку для проекта (например, lab21-variant15).
2. Внутри создайте файлы index.html и style.css (последний понадобится на следующем этапе).
3. Создайте каркас документа.

Анализ макета

Внимательно изучите предоставленный макет. Определите:

1. Основные блоки страницы (шапка, основное содержание, подвал).

2. Состав каждого блока (логотип, меню, баннеры, карточки товаров/услуг, формы и т.д.).

3. Иерархию заголовков (h1-h6).

4. Типы контента (текст, изображения, кнопки, иконки)

Разметка структуры страницы

Внимательно изучите предоставленный макет. Определите:

На основе анализа создайте семантическую структуру. Пример:

```
<body>
  <header class="header">
    <!-- Шапка: лого, навигация, контакты -->
  </header>
  <main class="main">
    <section class="hero">
      <!-- Блок-герой (баннер) -->
    </section>

    <section class="features">
      <!-- Блок с преимуществами -->
    </section>

    <article class="content">
      <!-- Основной контент -->
    </article>

    <aside class="sidebar">
      <!-- Боковая панель (если есть) -->
    </aside>
  </main>
  <footer class="footer">
    <!-- Подвал: ссылки, соцсети, копирайт -->
  </footer>
</body>
```

Детальная разметка блоков

Заполняйте каждый блок содержимым согласно макету:

- Используйте теги <h1>-<h6> для заголовков.
- Для навигации применяйте <nav> и списки /.
- Текстовые блоки размещайте в <p>.
- Кнопки – <button> или <a> с соответствующими классами.
- Изображения – с обязательными атрибутами.
- Формы– <form>, <input>, <label>.

Добавление классов

Каждому значимому элементу добавьте классы по методологии БЭМ:

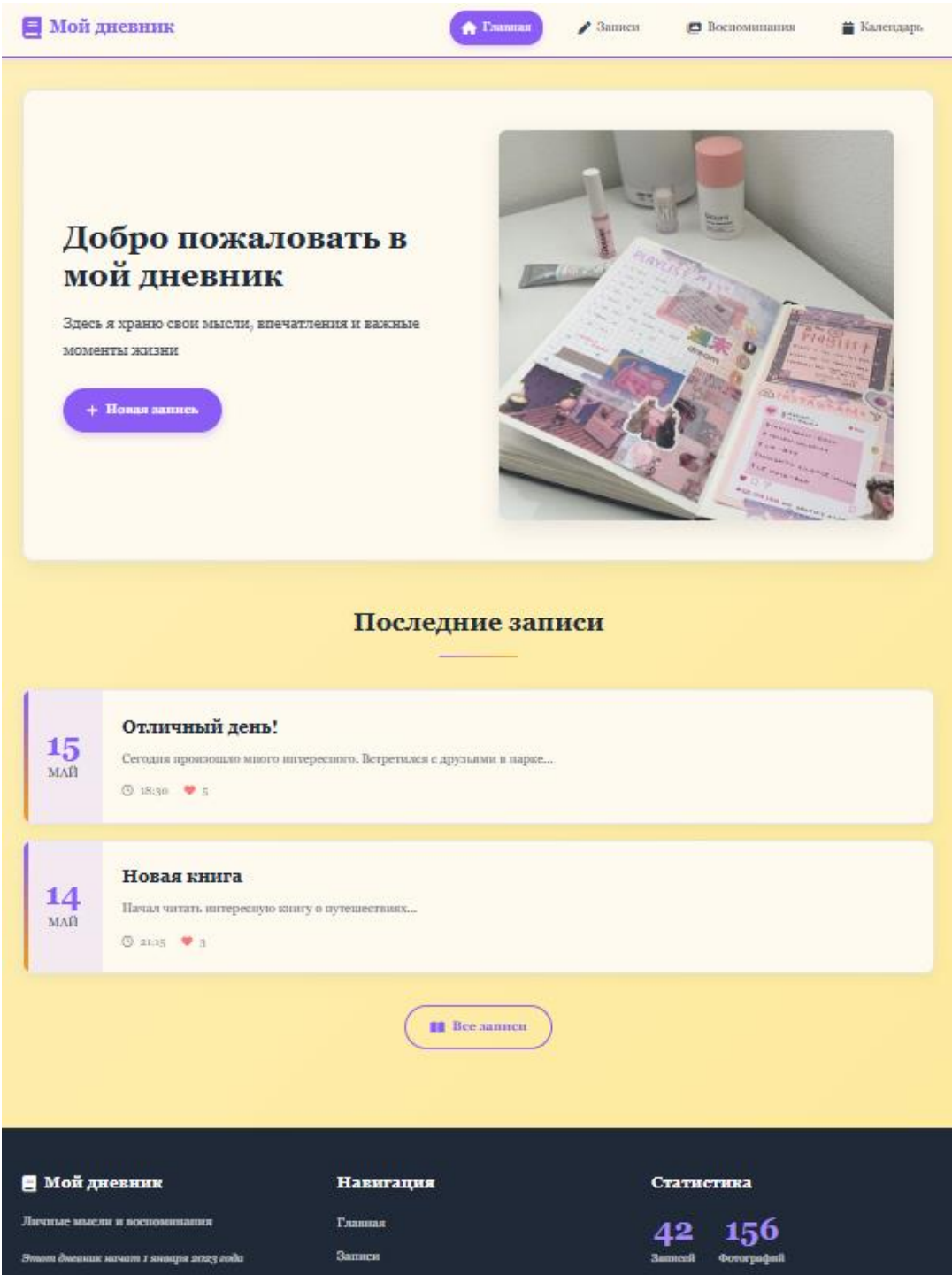
- Имя класса должно отражать функцию, а не внешний вид.
- Используйте двойное подчеркивание для элементов (block__element).
- Используйте двойной дефис для модификаторов (block--modifier).

Убедитесь, что все элементы макета присутствуют в разметке.

Задания для выполнения лабораторной работы

Создайте разметку всей структуры страницы с использованием семантических тегов и классов используя полученный макет по вариантам.


Вариант 1. Личный дневник.



Вариант 2. Сайт городского мероприятия.

Городской фестиваль

[Главная](#)[Программа](#)[Карта](#)[Информация](#)




Городской фестиваль 2023

Парк Горького, 10:00-22:00


[Смотреть программу](#)

О фестивале


Ежегодный праздник для всей семьи. Концерты, мастер-классы, еда и развлечения.



Музыка
Живые выступления



Творчество
Мастер-классы

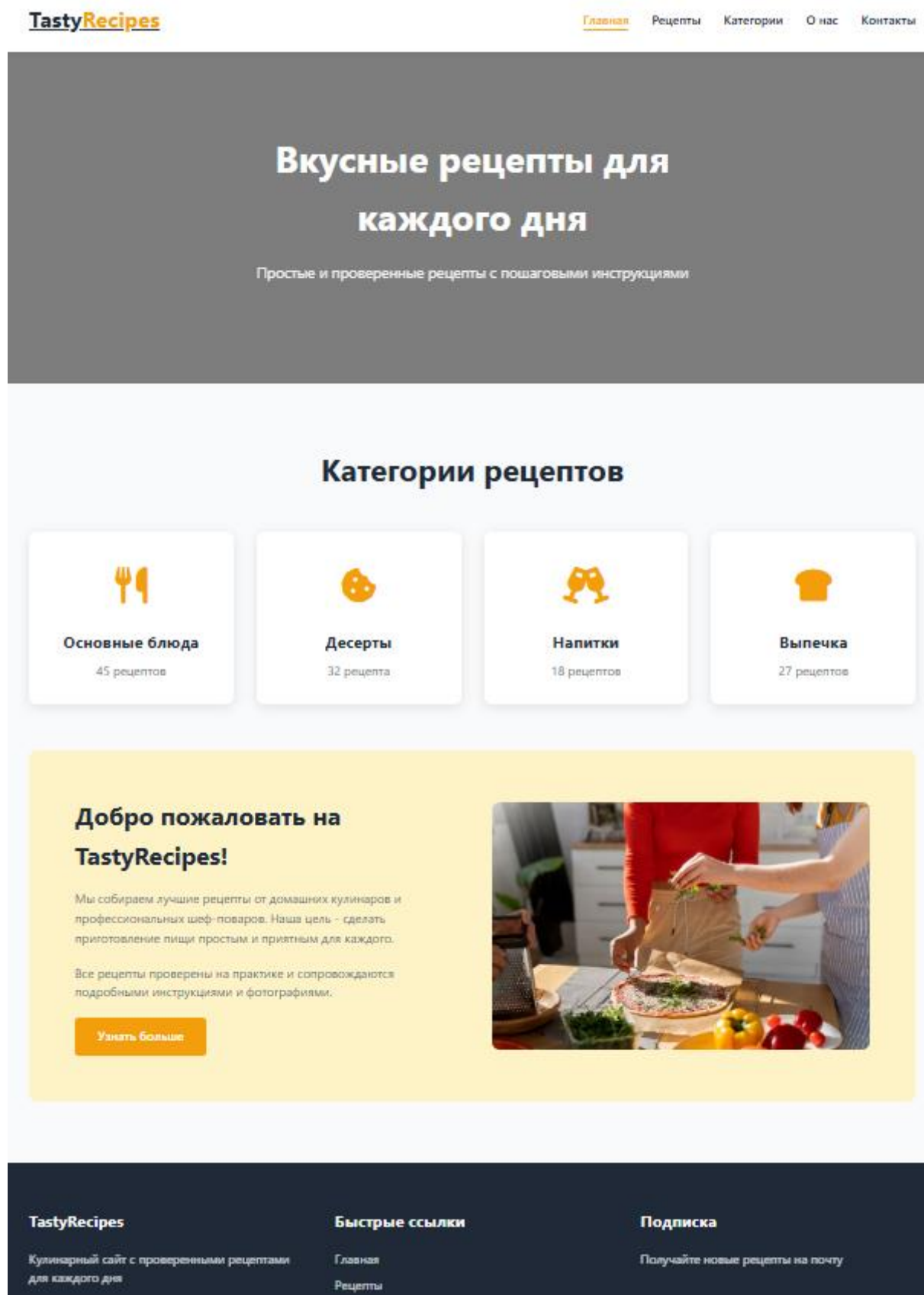


Еда
Фудкорты

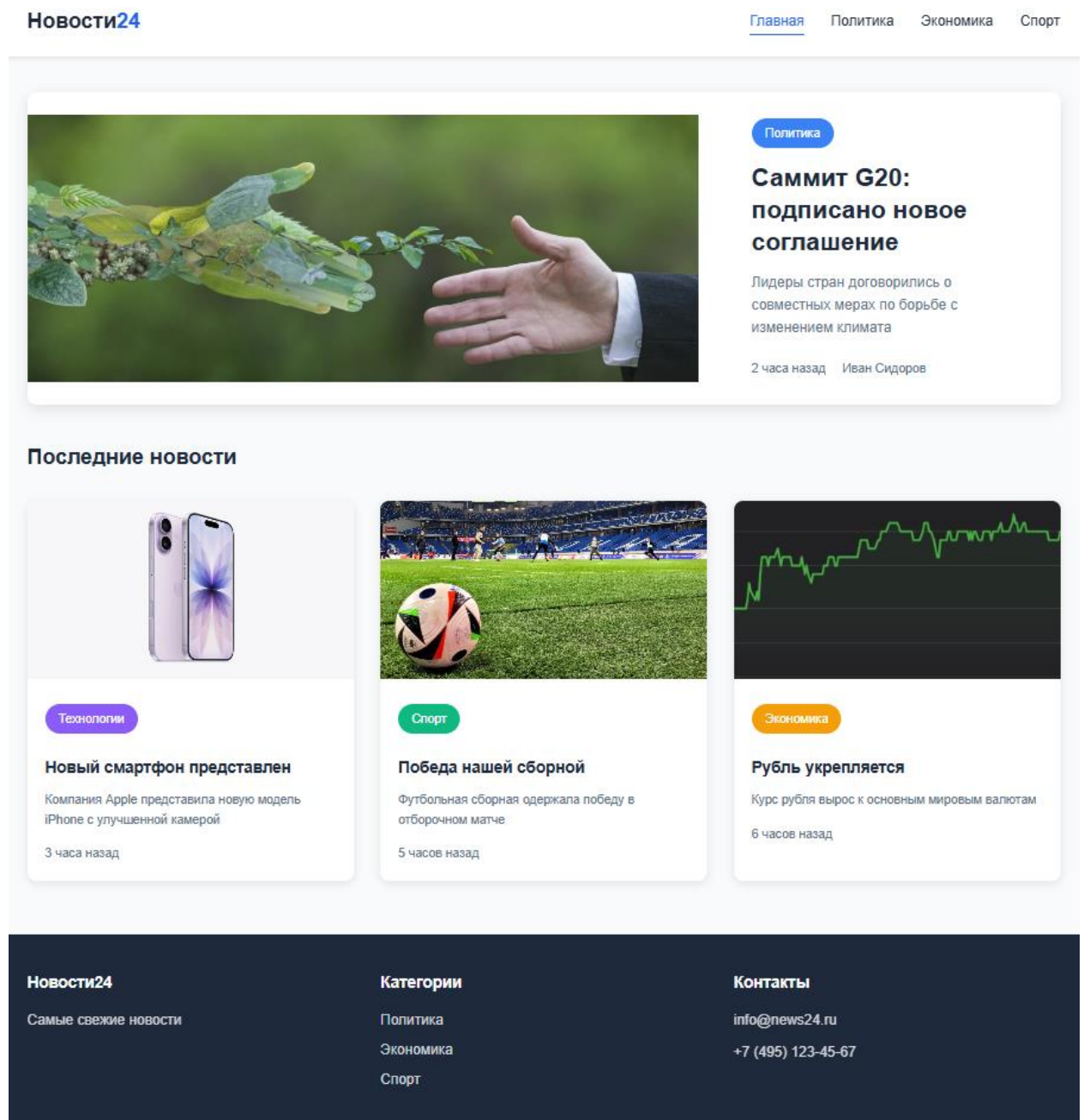
Городской фестиваль © 2023

fest@gorod.ru

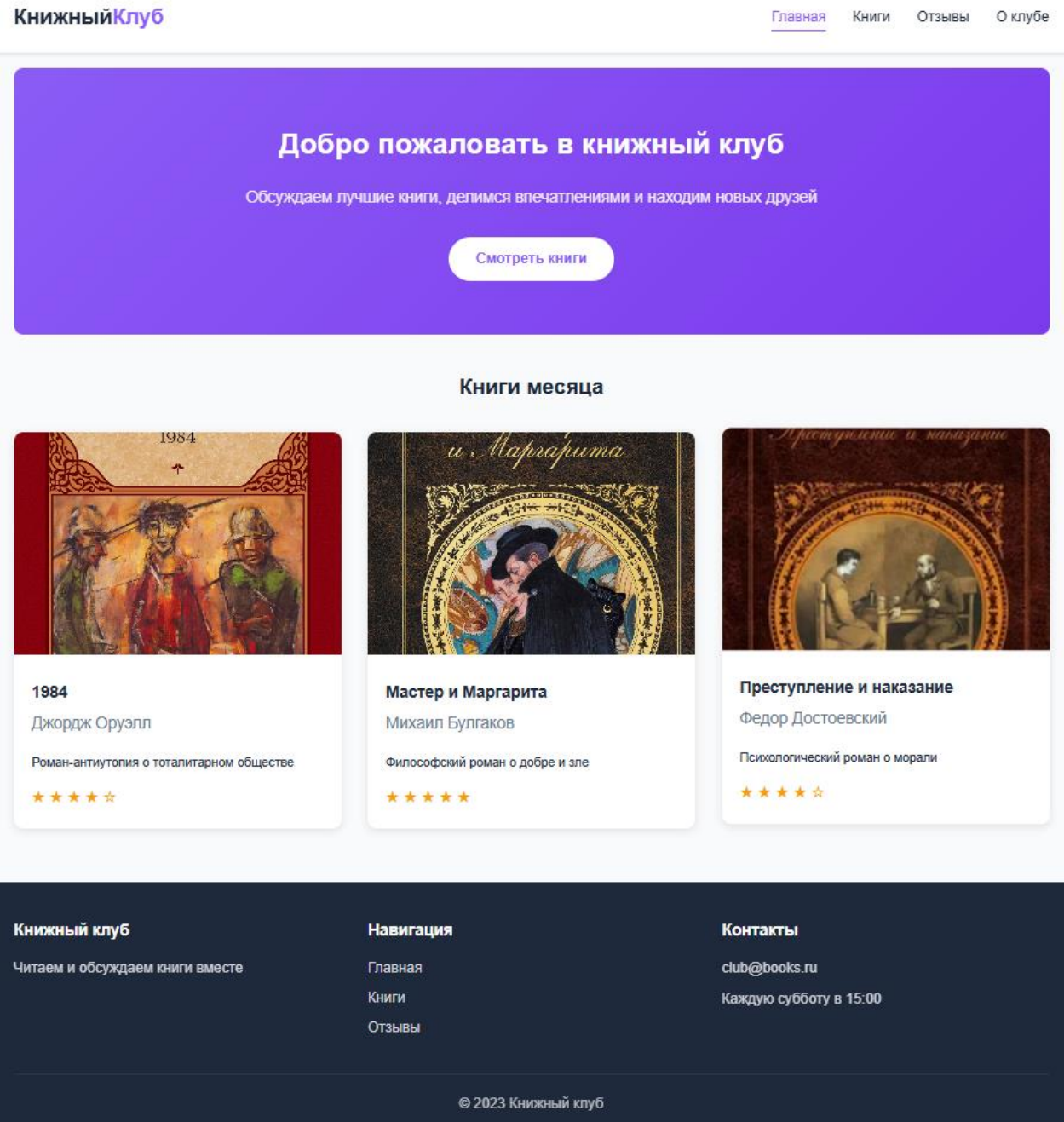
Вариант 3. Сайт кулинарных рецептов.



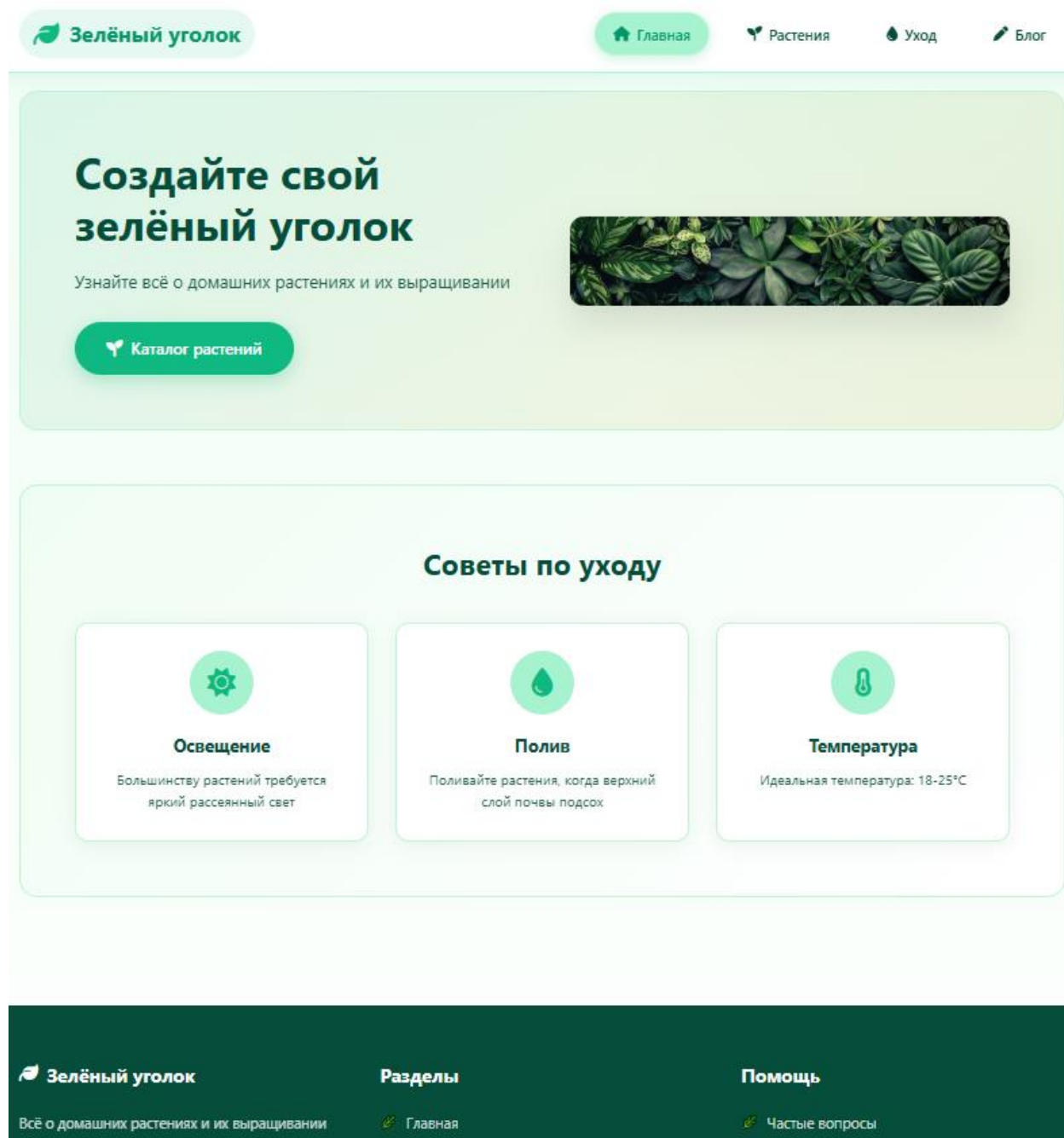
Вариант 4. Новостной портал.



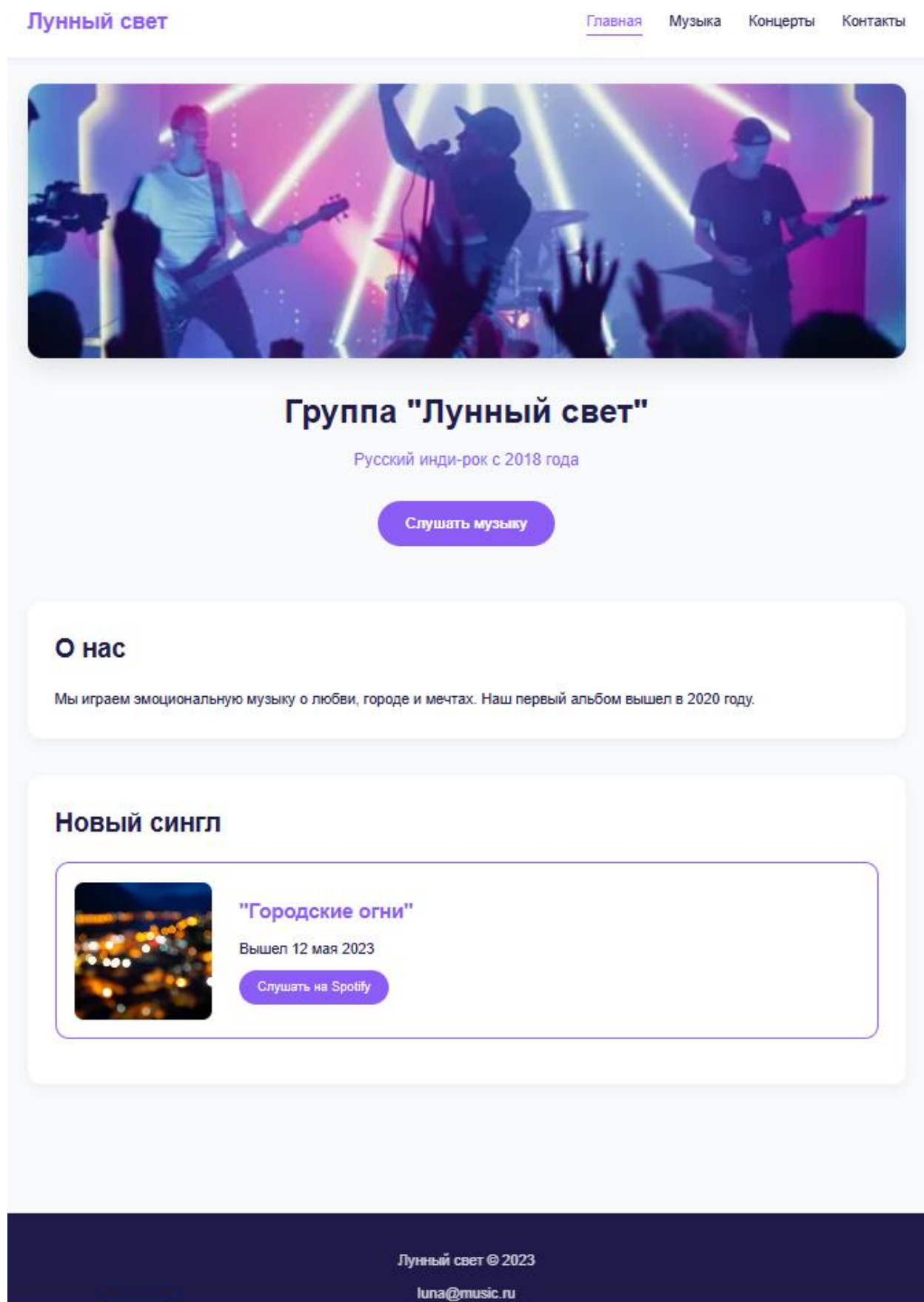
Вариант 5. Сайт книжного клуба.



Вариант 6. Сайт о домашних растениях.



Вариант 7. Сайт музыкальной группы.



Вариант 8. Сайт образовательного центра.

ОбразовательныйЦентр


[Главная](#)[Курсы](#)[Преподаватели](#)[Контакты](#)

Добро пожаловать в образовательный центр

Развиваем навыки для успешного будущего

Смотреть курсы


Популярные курсы



Программирование

Изучение Python и веб-разработки


Подробнее →



Аналитика данных

Работа с Excel и Power BI

Подробнее →



Английский язык

Разговорный курс для начинающих

Подробнее →

Образовательный центр

Знания для успешного будущего

Контакты

+7 (495) 123-45-67

Москва, ул. Образовательная, 5

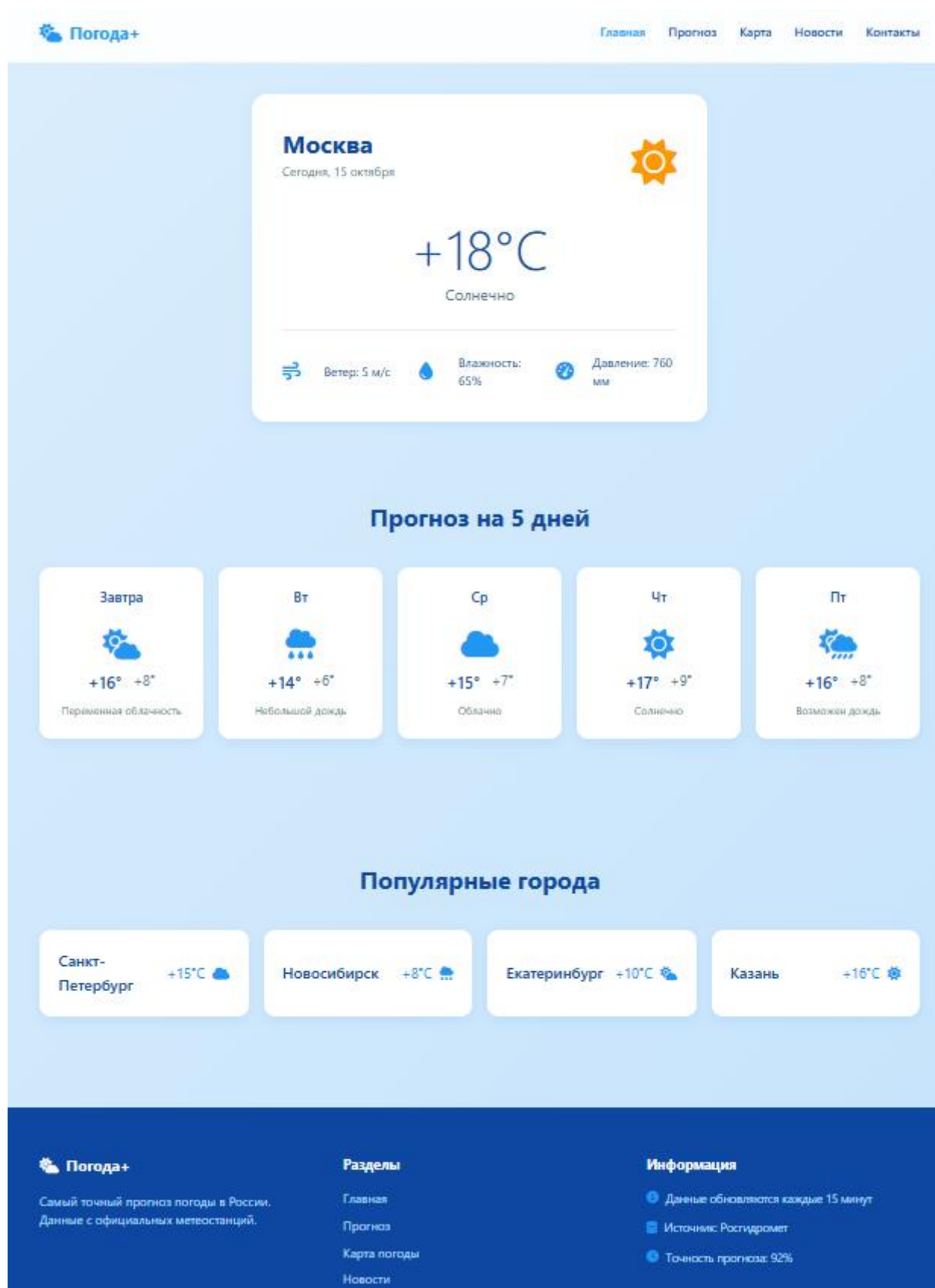
Часы работы

Пн-Пт: 9:00-20:00

Сб-Вс: 10:00-18:00

© 2023 Образовательный центр

Вариант 9. Сайт погодного сервиса.



Вариант 10. Сайт кинотеатра.

Кинотеатр "Экран"

Главная

Фильмы

Расписание

Цены


Контакты

Лучшие фильмы на
большом экране

Премьеры, блокбастеры и мировые хиты в лучшем качестве.
Современные залы, комфортные кресла и отличный звук.


Купить билет

Что идет сегодня



Сейчас в кино

Выберите фильм для просмотра



16+


Грань будущего

Фантастика, боевик

2ч 15м

8.2

Подробнее



12+


Смех до слёз

Комедия, приключения

1ч 45м

7.5

Подробнее



18+

Мгновения жизни

Драма, мелодрама

2ч 30м

8.7

Подробнее

Кинотеатр "Экран"

ул. Кинотеатральная, 15
Ежедневно с 9:00 до 23:00

Быстрые ссылки

Главная

Фильмы

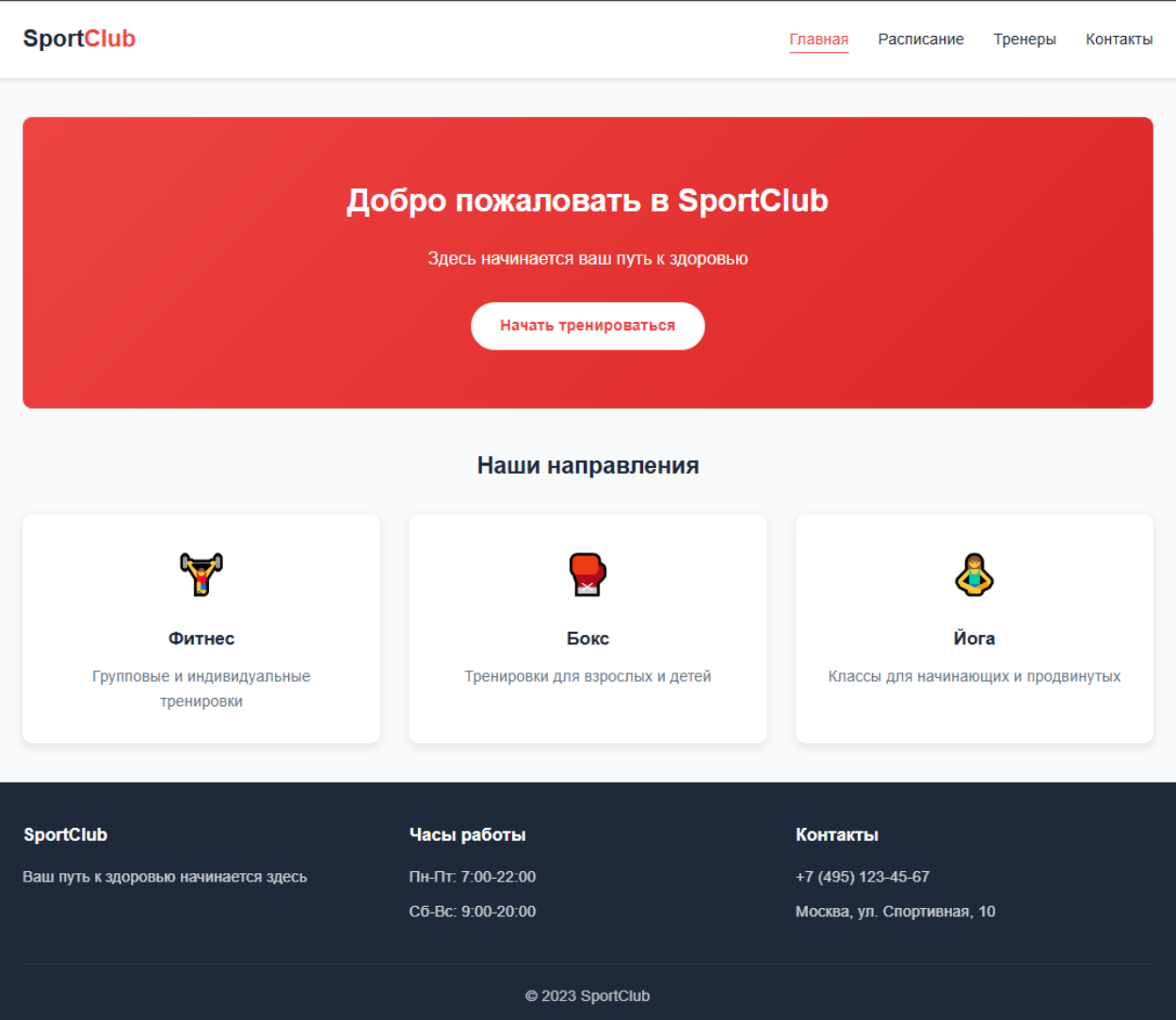
Контакты

+7 (495) 123-45-67

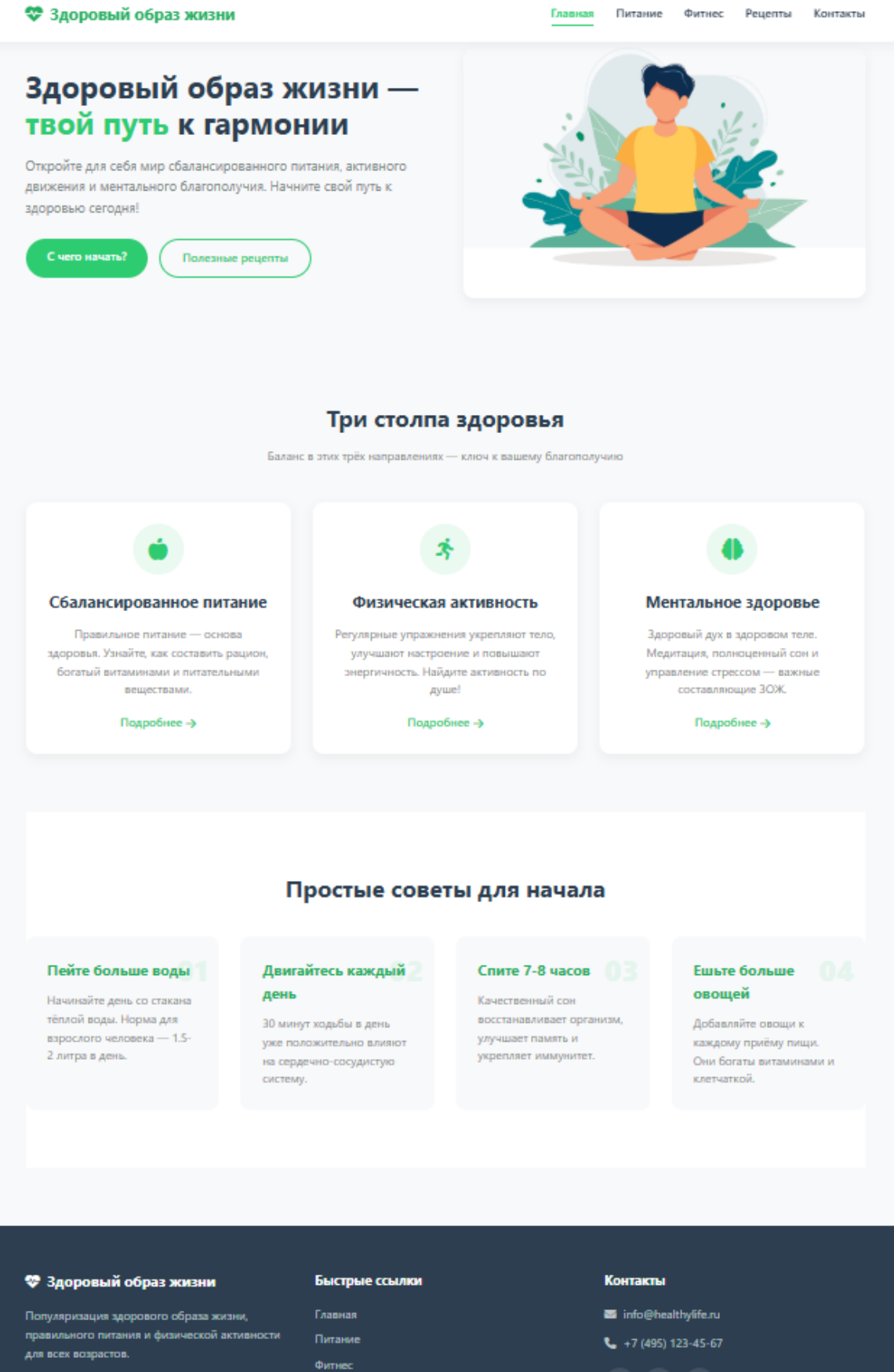
info@ekran-cinema.ru

163

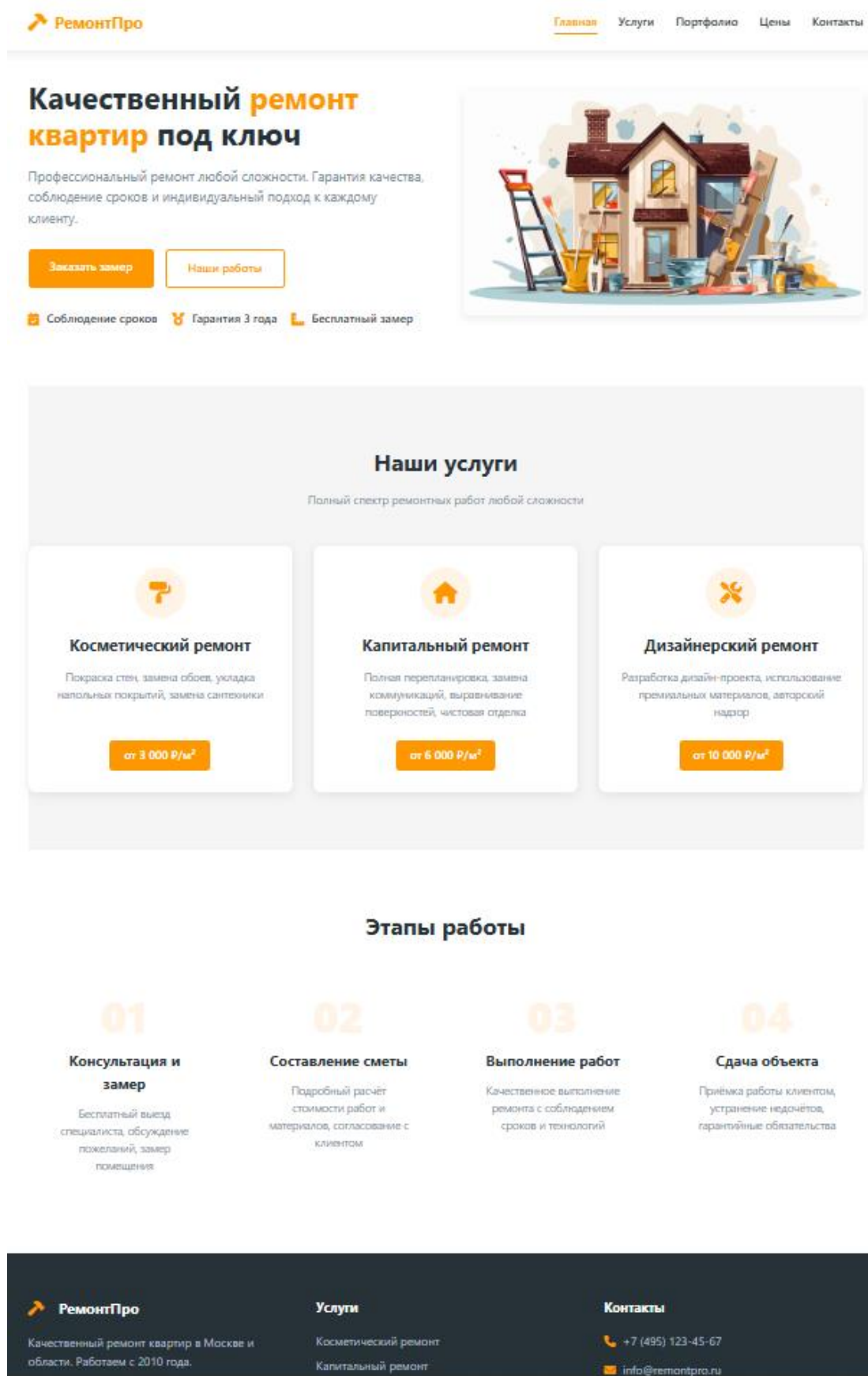
Вариант 11. Спортивный сайт.



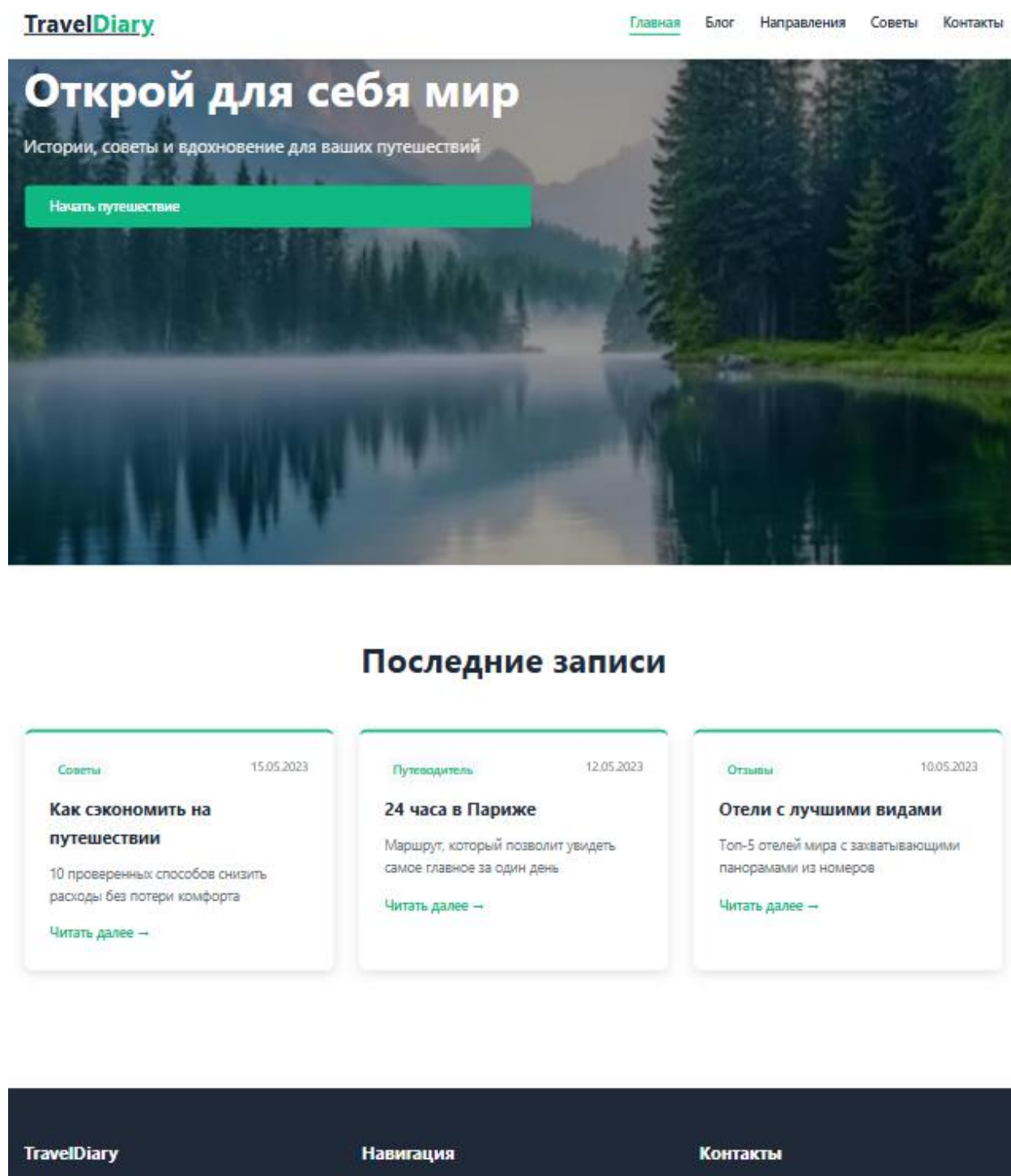
Вариант 12. Сайт про здоровый образ жизни



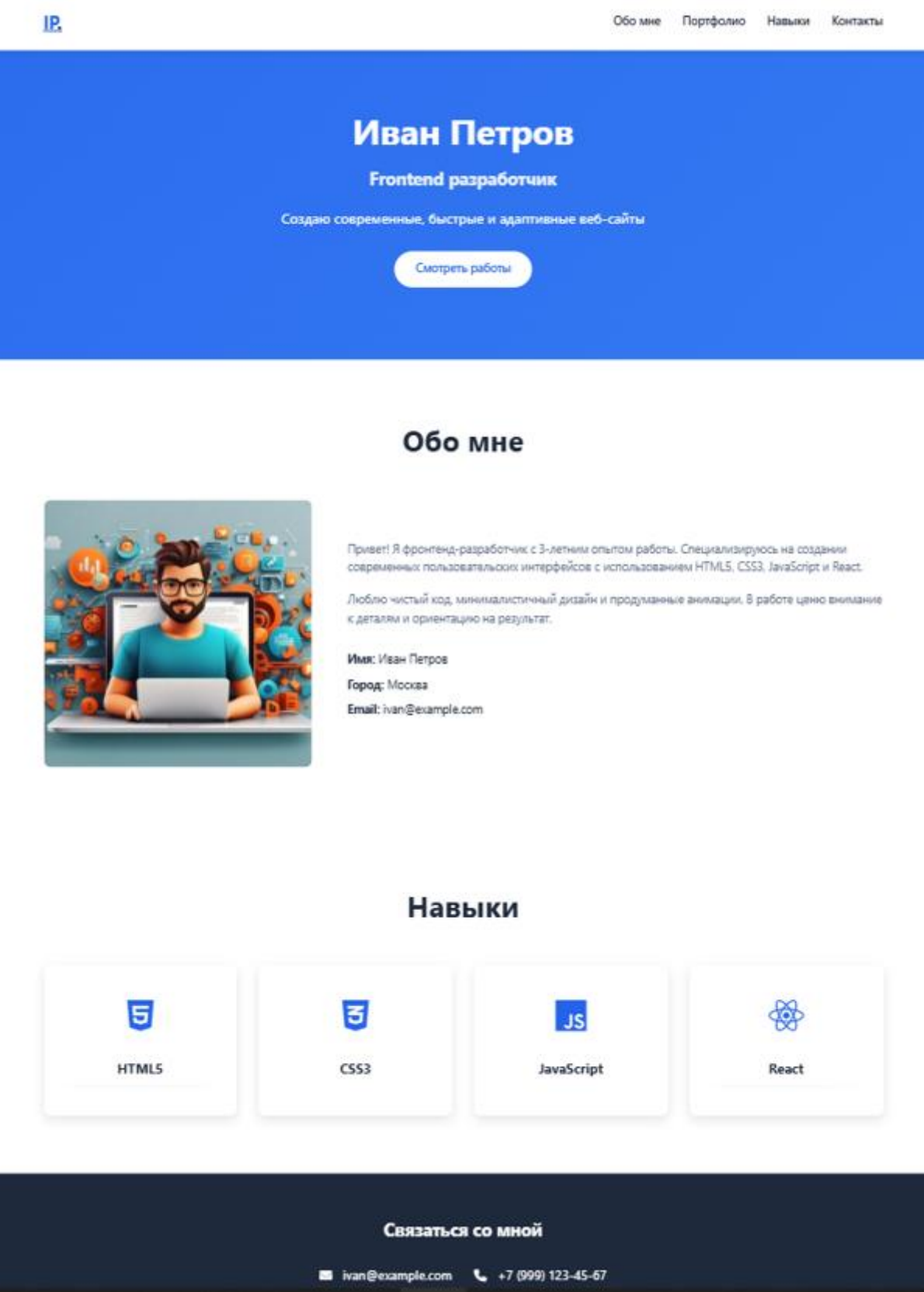
Вариант 13. Сайт ремонта квартир.



Вариант 14. Блог о путешествиях.



Вариант 15. Сайт-портфолио веб-разработчика.



Лабораторная работа № 22

«Практическая работа: верстка макета главной страницы по готовому шаблону (этап 2 - Базовая стилизация)»

Цель работы: освоить практические навыки подключения и написания CSS-стилей для HTML-структуры для готового макета сайта.

Методические указания

Селекторы CSS определяют, к каким элементам применяются стили:

1. Базовые селекторы:
 - * – универсальный селектор
 - div – селектор по тегу
 - .class – селектор по классу
 - #id – селектор по идентификатору
2. Комбинаторы:
 - div p – потомки
 - div > p – прямые потомки
 - .class1.class2 – множественные классы
 - div, p – группировка
3. Псевдоклассы и псевдоэлементы:
 - :hover, :focus, :nth-child()
 - ::before, ::after

Относительные единицы:

- % – процент от размера родителя
- em – относительно размера шрифта элемента
- rem – относительно размера шрифта корневого элемента
- vw / vh – процент от ширины/высоты окна просмотра

Абсолютные единицы:

- px – пиксели
- pt – пункты (1pt = 1/72 дюйма)

Flexbox – одномерная система компоновки для расположения элементов в строке или столбце.

Основные свойства контейнера:

- display: flex – активация flex-контейнера
- flex-direction – направление основной оси
- justify-content – выравнивание по основной оси
- align-items – выравнивание по поперечной оси
- flex-wrap – перенос элементов

Свойства flex-элементов:

- flex-grow – коэффициент роста
- flex-shrink – коэффициент сжатия
- flex-basis – базовый размер
- align-self – индивидуальное выравнивание

Подготовка рабочей среды

1. Создайте файл style.css.
2. Убедитесь, что в html файле в <head> подключен CSS-файл.
`<link rel="stylesheet" href="style.css">`
3. Создайте каркас CSS-файла с базовыми блоками.

Проанализируйте макет №15 и выпишите:

1. Основные цвета (фона, текста, акцентные)
2. Шрифтовые пары (заголовки, основной текст)
3. Размеры контейнеров, отступы
4. Тени, скругления углов

Работайте от общего к частному. Не забудьте проверить работу стилей в браузере.

Задания для выполнения лабораторной работы

Оформите структуру страницы, созданный в лабораторной работе № 21, стилями CSS.

Лабораторная работа № 23

«Практическая работа: верстка макета главной страницы по готовому шаблону (этап 3 - Детальная стилизация)»

Цель работы: освоить навыки детальной стилизации интерфейсных компонентов.

Методические указания

Иерархия текста:

- H1 – главный заголовок (обычно 1-2 на странице)
- H2-H3 – заголовки разделов
- H4-H6 – подзаголовки и акценты
- Body – основной текст (16px+ для читаемости)
- Small/Caption – вспомогательный текст

Метрики типографики:

- Line-height (межстрочный интервал): 1.4-1.6 для body текста
- Letter-spacing (межбуквенный интервал): -0.5px до 1px
- Font-weight: 400 (regular), 500 (medium), 600 (semibold), 700 (bold)

Компонентный подход

Atomic Design принципы:

- Атомы: кнопки, инпуты, иконки
- Молекулы: формы, карточки
- Организмы: навигация, секции
- Шаблоны: страницы

Состояния элементов

Каждый интерактивный элемент должен иметь:

- Normal – обычное состояние
- Hover – при наведении курсора
- Focus – при фокусе (клавиатурная навигация)
- Active – при нажатии
- Disabled – неактивное состояние

Проанализируйте макет веб-сайта и определите:

- Шрифты
- Размеры шрифтов
- Межстрочные интервалы
- Глобальные стили типографики
- Заголовки с кастомными отступами
- Специальные текстовые классы

Также нужно детально стилизовать систему кнопок, карточек и иконок.

Задания для выполнения лабораторной работы

Продолжите детальную стилизацию страницы, созданный в лабораторной работе № 21, веб-сайта стилями CSS.

Лабораторная работа № 24

«Практическая работа: верстка макета главной страницы по готовому шаблону (этап 4 - Адаптация)»

Цель работы: освоить практические навыки создания адаптивных веб-страниц с использованием медиа-запросов. Научиться адаптировать макет под различные разрешения экранов: планшетные и мобильные устройства.

Методические указания

Адаптивный веб-дизайн (Responsive Web Design) – подход к веб-дизайну, который обеспечивает оптимальное отображение и удобство использования сайта на различных устройствах с разными размерами экранов.

Основные принципы адаптивного дизайна:

1. Гибкая сетка (Flexible Grid)

- Использование относительных единиц измерения (%)
- Отказ от фиксированных пиксельных размеров

2. Гибкие изображения (Flexible Images)

- Изображения масштабируются относительно родительского контейнера
- Использование max-width: 100% для изображений

3. Медиа-запросы (Media Queries)

- Позволяют применять CSS-правила в зависимости от характеристик устройства

Типичные брейкпоинты (breakpoints):

```
@media (max-width: 767px) {  
    /* Стили для мобильных */  
}  
@media (min-width: 768px) and (max-width: 1023px) {  
    /* Стили для планшетов */  
}  
@media (min-width: 1024px) {  
    /* Стили для десктопов */  
}
```

Адаптивные стратегии:

1. **Mobile First** – проектирование сначала для мобильных устройств

2. **Desktop First** – проектирование сначала для десктопов (более традиционный подход)

Подготовка рабочей среды

1. Откройте проект из предыдущих лабораторных работ
2. Убедитесь, что у вас есть:
 - Завершенный HTML-файл с семантической разметкой
 - CSS-файл с базовой стилизацией
 - Скриншот макета (вариант №15)

Анализ поведения элементов на разных разрешениях

Внимательно изучите макет и определите:

1. Элементы, требующие реорганизации:
 - Навигационное меню (преобразование в бургер-меню)
 - Многоуровневые сетки (grid/flex)
 - Боковые панели
 - Таблицы данных
2. Элементы, требующие изменения размеров:
 - Заголовки и текст
 - Изображения и медиа
 - Отступы и промежутки
 - Кнопки и формы
3. Элементы, требующие скрытия/отображения:
 - Второстепенные элементы на мобильных
 - Декоративные элементы
 - Дублирующий контент

Задания для выполнения лабораторной работы

Проанализируйте десктопную версию вашего макета по вариантам и определите элементы, требующие адаптации. Создайте планшетную версию (768px-1024px) и мобильную версию (до 767px).

Лабораторная работа № 25

«Практическая работа: верстка внутренней страницы по готовому шаблону»

Цель работы: освоить практические навыки создания внутренних страниц сайта на основе готовых компонентов главной страницы. Научиться повторно использовать стили и компоненты для обеспечения единообразия дизайна.

Методические указания

Компонентный подход в веб-разработке – методология, при которой интерфейс разбивается на независимые переиспользуемые компоненты. Это позволяет:

- Сократить время разработки
- Обеспечить единообразие дизайна
- Упростить поддержку и обновление
- Повысить переиспользуемость кода

Типы внутренних страниц:

1. Статьи/Блог – текстовый контент с изображениями
2. Каталог товаров – сетка товаров с фильтрами
3. Контакты – контактная информация, карта, форма обратной связи
4. О компании – информация о компании, команда
5. Услуги – детальное описание услуг

Общие элементы внутренних страниц:

- Шапка (header) с навигацией
- Хлебные крошки (breadcrumbs)
- Основной контент
- Боковая панель (sidebar) – опционально
- Подвал (footer)

Задания для выполнения лабораторной работы

1. Создайте новую папку для внутренней страницы в вашем проекте:

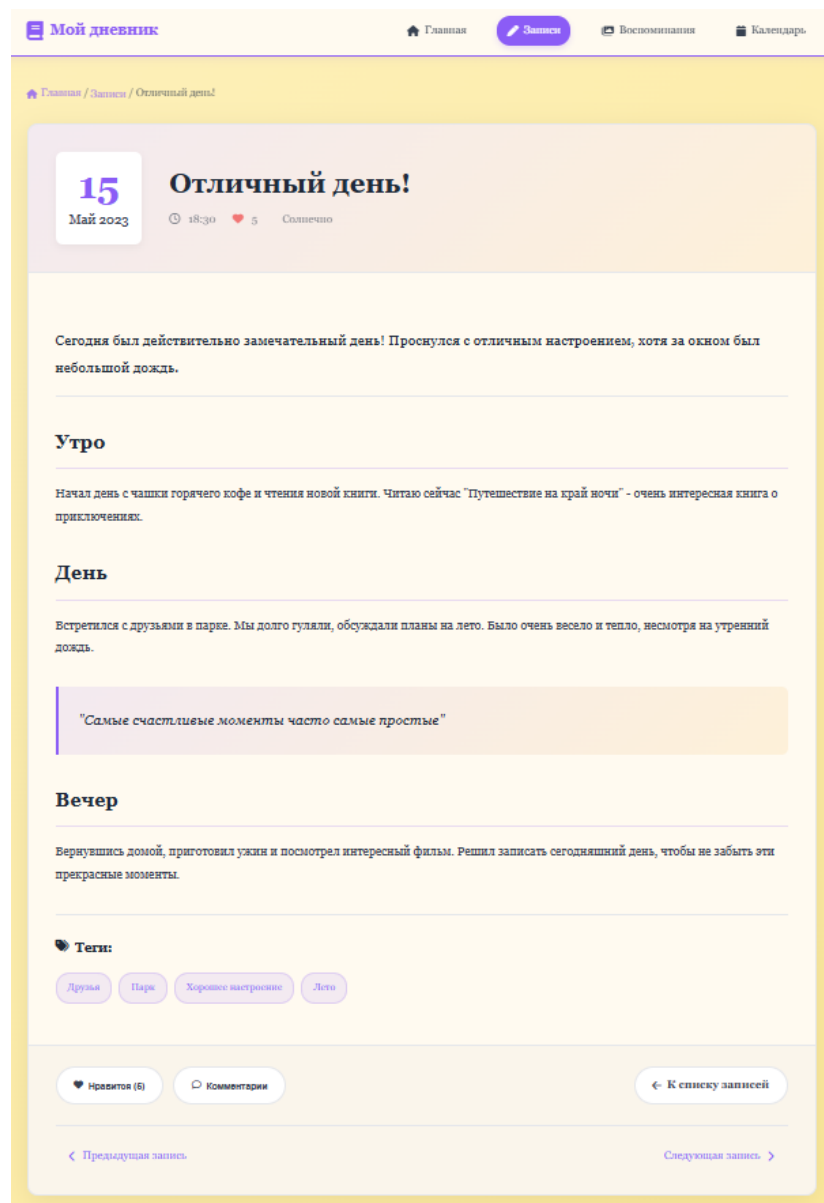
- project-folder/
- index.html (главная страница)
- style.css (общие стили)
- article.html (страница статьи)

2. Скопируйте общие компоненты из главной страницы:

- Шапку (<header>)
- Подвал (<footer>)
- Навигацию (<nav>)
- Общие CSS-стили

3. Откройте скриншот внутренней страницы для анализа по вариантам.

Вариант 1. Личный дневник.



Вариант 2. Сайт городского мероприятия.

Программа фестиваля

15 июня, Парк Горького



12:00-14:00

Открытие фестиваля

Главная сцена



14:00-16:00

Концерт групп

Музыкальная площадка



16:00-18:00

Мастер-классы

Творческая зона



18:00-20:00

Вечерний концерт

Главная сцена

Вход свободный. Все мероприятия бесплатные.


[← На главную](#)


Вариант 3. Сайт кулинарных рецептов.


TastyRecipes


ГлавнаяРецептыКатегорииО насКонтакты

Спагетти Карбонара

Время
30 минут

Порции
4 персоны

Сложность
Легко

Лайки
245

Ингредиенты

- Спагетти - 400 г
- Бекон - 200 г
- Яйца - 4 шт
- Сыр Пармезан - 100 г
- Чеснок - 2 зубчика
- Оливковое масло - 2 ст.л.
- Соль - по вкусу
- Черный перец - по вкусу

Способ приготовления

- Подготовка ингредиентов**

Нарежьте бекон небольшими кусочками. Натрите сыр на мелкой терке. Отделите желтки от белков (для карбонары нужны только желтки).
- Приготовление бекона**

Разогрейте сковороду с оливковым маслом, добавьте измельченный чеснок и обжарьте 1 минуту. Добавьте бекон и жарьте до золотистого цвета.
- Варка пасты**

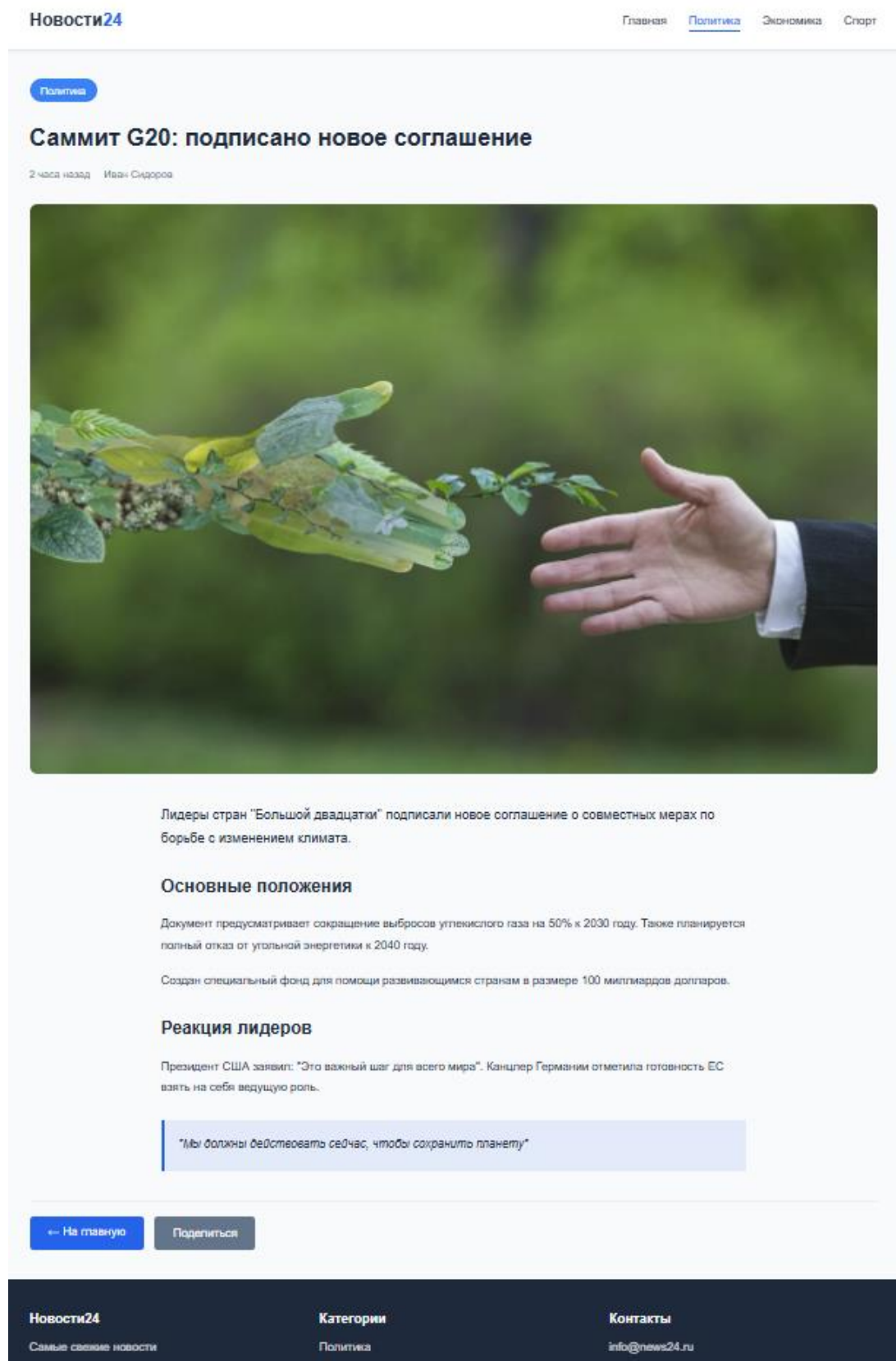
В большой кастрюле вскипятите воду, добавьте соль и варите спагетти согласно инструкции на упаковке (обычно 8-10 минут).
- Приготовление соуса**

В миске смешайте яичные желтки, тертый пармезан и черный перец. Добавьте 2-3 ложки воды от варки пасты, чтобы соус стал кремообразным.
- Соединение ингредиентов**

Слейте воду с пасты, сохранив немного воды. Добавьте пасту в сковороду с беконом, снимите с огня и быстро смешайте с яично-сырной смесью.
- Подача**

Немедленно подавайте блюдо, посыпав дополнительным сыром пармезан и черным перцем.

Вариант 4. Новостной портал.

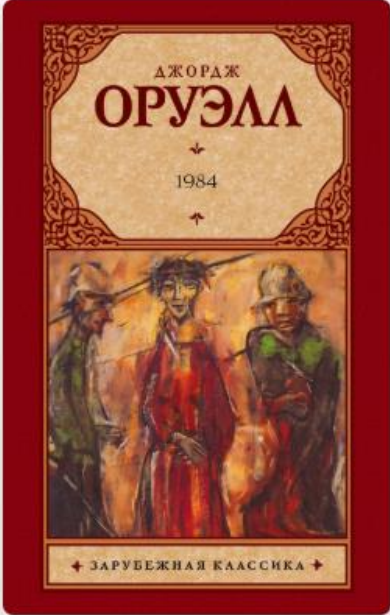


Вариант 5. Сайт книжного клуба.

КнижныйКлуб

ГлавнаяКнигиОтзывыО клубе

Главная / Книги / 1984



1984

Джордж Оруэлл

1949 Антиутопия ★★★★★

О книге

Роман-антиутопия, действие которого происходит в тоталитарном обществе под постоянным контролем "Большого Брата".

Главный герой, Уинстон Смит, работает в Министерстве правды и начинает сомневаться в системе.

Детали

Страниц: 328

Язык: Английский

Издательство: Secker & Warburg

Добавить в списокОбсудить в клубе

Отзывы

Анна16.05.2023★★★★★

Потрясающая книга! Заставляет задуматься о свободе и контроле.

Сергей10.05.2023★★★★☆

Актуально даже сегодня. Рекомендую к прочтению.

Добавить отзыв


Ваш отзыв...

Отправить

Вариант 6. Сайт о домашних растениях.

Зелёный уголок

ГлавнаяРастенияУходБлог



Монстера


Monstera deliciosa

ВЫСОТА
до 2 м

ШИРИНА
до 1.5 м

ТИП
Лиственное


Руководство по уходу



Освещение

Яркий рассеянный свет. Прямое солнце может обжечь листья.


Средний свет



Полив

Поливайте, когда верхние 2-3 см почвы подсохнут. Летом - 1 раз в неделю.

Умеренный полив



Температура

18-27°C. Не ниже 15°C зимой.

Теплолюбивая

Описание

Монстера - одно из самых популярных комнатных растений благодаря своим крупным перфорированным листьям. Родом из тропических лесов Центральной Америки.

Растение известно своими воздушными корнями, которые помогают ему получать дополнительную влагу из воздуха. Монстера может вырастать до 2 метров в высоту при правильном уходе.

Особенности

✓ Быстрый рост

✓ Очищает воздух

✓ Не токсично для животных

✓ Легко размножается

Частые проблемы

Желтеющие листья

Причина: перелив. Дайте почве просохнуть между поливами.

Коричневые кончики

Причина: низкая влажность. Опрыскивайте растение.

Мелкие листья


Причина: недостаток света. Переставьте в более светлое место.

Вариант 7. Сайт музыкальной группы.

Лунный свет

ГлавнаяМузыкаКонцертыКонтакты


Наша музыка



Альбом "Ночной город" (2022)


Лунная дорога	4:12
Дождь в октябре	3:45
Метро	5:01

Синглы



Городские огни (2023)

Слушать



Белая ночь (2021)

Слушать

← На главную

Лунный свет © 2023

luna@music.ru

182

Вариант 8. Сайт образовательного центра.

ОбразовательныйЦентр

ГлавнаяКурсыПреподавателиКонтакты

Главная / Курсы / Программирование

Курс программирования на Python

Длительность: 3 месяца

О курсе

Изучите основы программирования на Python и создадите свои первые проекты. Курс подходит для начинающих.

Что вы узнаете:

- ✓ Основы синтаксиса Python
- ✓ Работа с данными и файлами
- ✓ Создание веб-приложений
- ✓ Работа с базами данных

Детали курса

Формат:	Занятия:	Стоимость:
Онлайн	2 раза в неделю	25 000 Р

[Записаться на курс](#)[← Все курсы](#)

Образовательный центр

Знания для успешного будущего

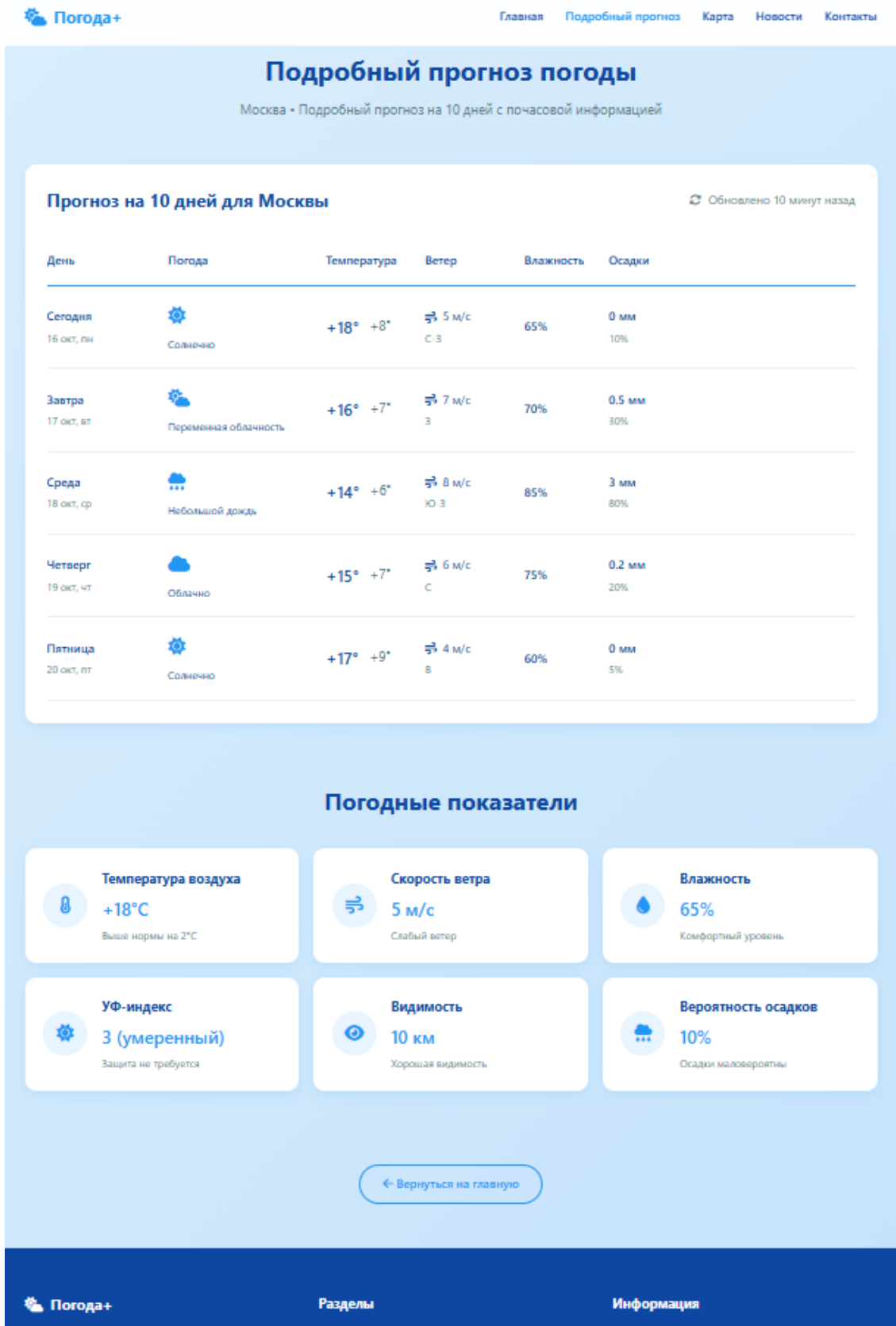
Контакты

+7 (495) 123-45-67
Москва, ул. Образовательная, 5

Часы работы

Пн-Пт: 9:00-20:00
Сб-Вс: 10:00-18:00

Вариант 9. Сайт погодного сервиса.



Вариант 10. Сайт кинотеатра.

Кинотеатр "Экран"

ГлавнаяФильмыРасписаниеЦеныКонтакты

Расписание сеансов

С 15 по 21 октября 2023 года

Все залы

Зал 1 (Большой)

Зал 2 (3D)

Зал 3 (VIP)

Сегодня

Завтра

Понедельник, 16 октября

10:00

Грань будущего
Зал 1 • 2ч 15м • 16+

450 Р

Купить

13:30

Смех до слёз
Зал 2 • 1ч 45м • 12+

400 Р

Купить

17:00

Тихие мгновения
Зал 3 • 2ч 30м • 18+

500 Р

Купить

20:30

Грань будущего
Зал 1 • 2ч 15м • 16+

450 Р

Купить

Вторник, 17 октября

11:00

Смех до слёз
Зал 2 • 1ч 45м • 12+

400 Р

Купить

14:30

Грань будущего
Зал 1 • 2ч 15м • 16+

450 Р

Купить

18:00

Тихие мгновения
Зал 3 • 2ч 30м • 18+

500 Р

Купить

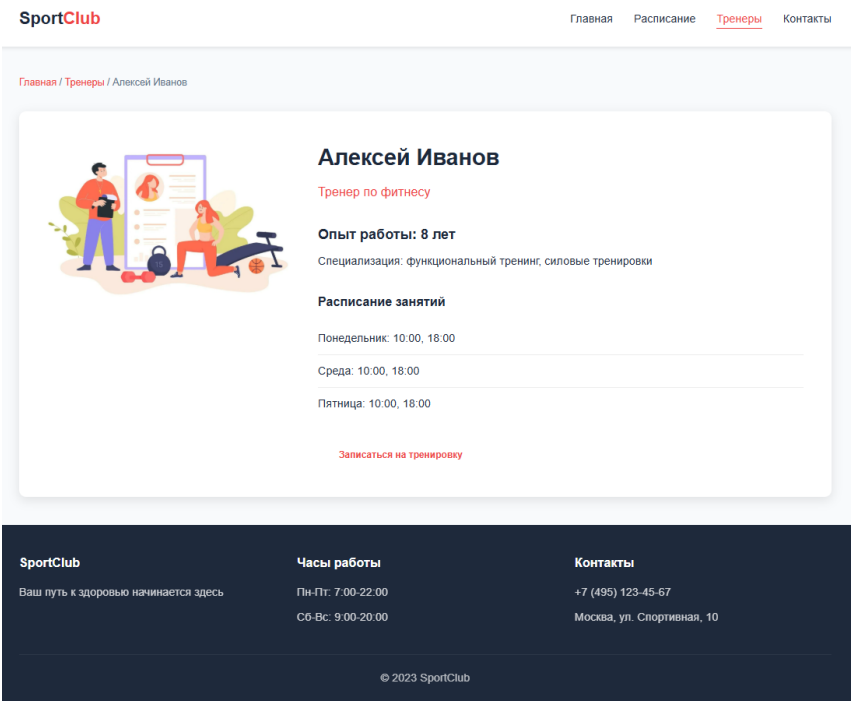
21:15

Смех до слёз
Зал 2 • 1ч 45м • 12+

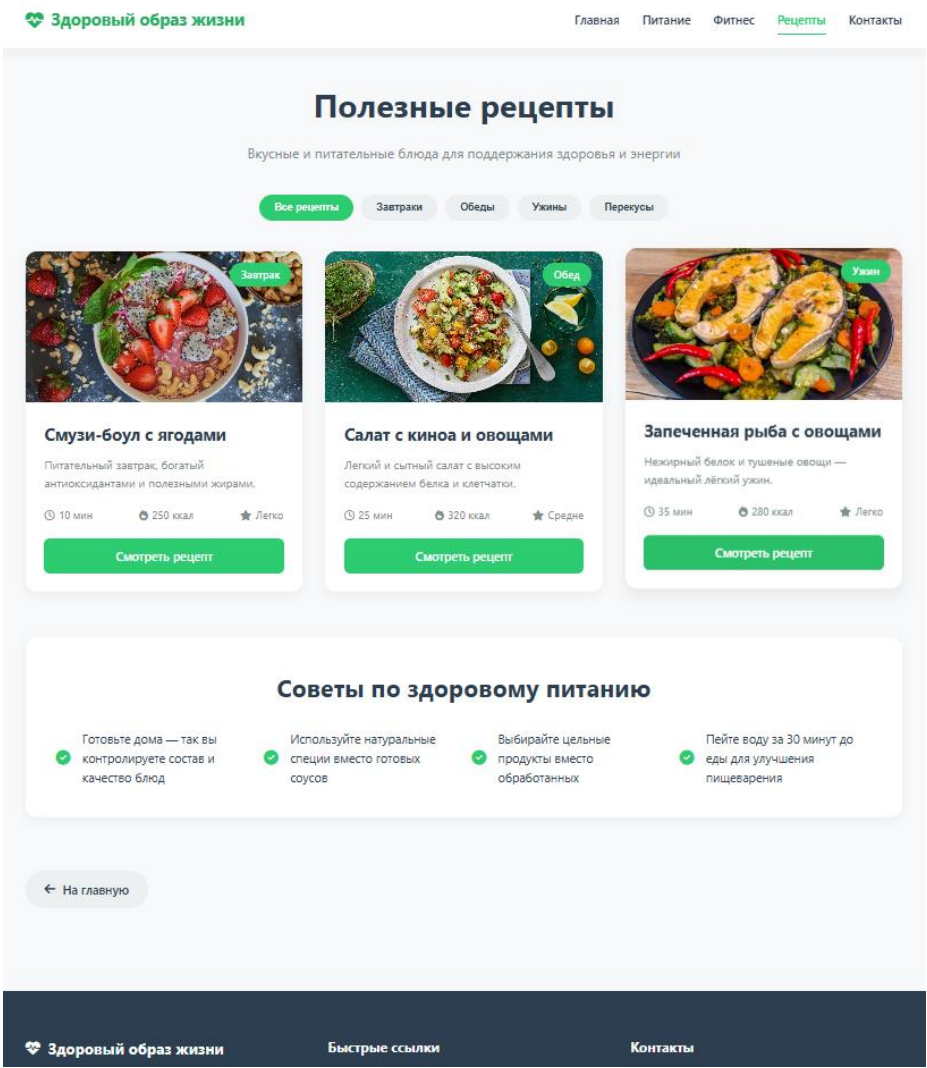
400 Р

Купить

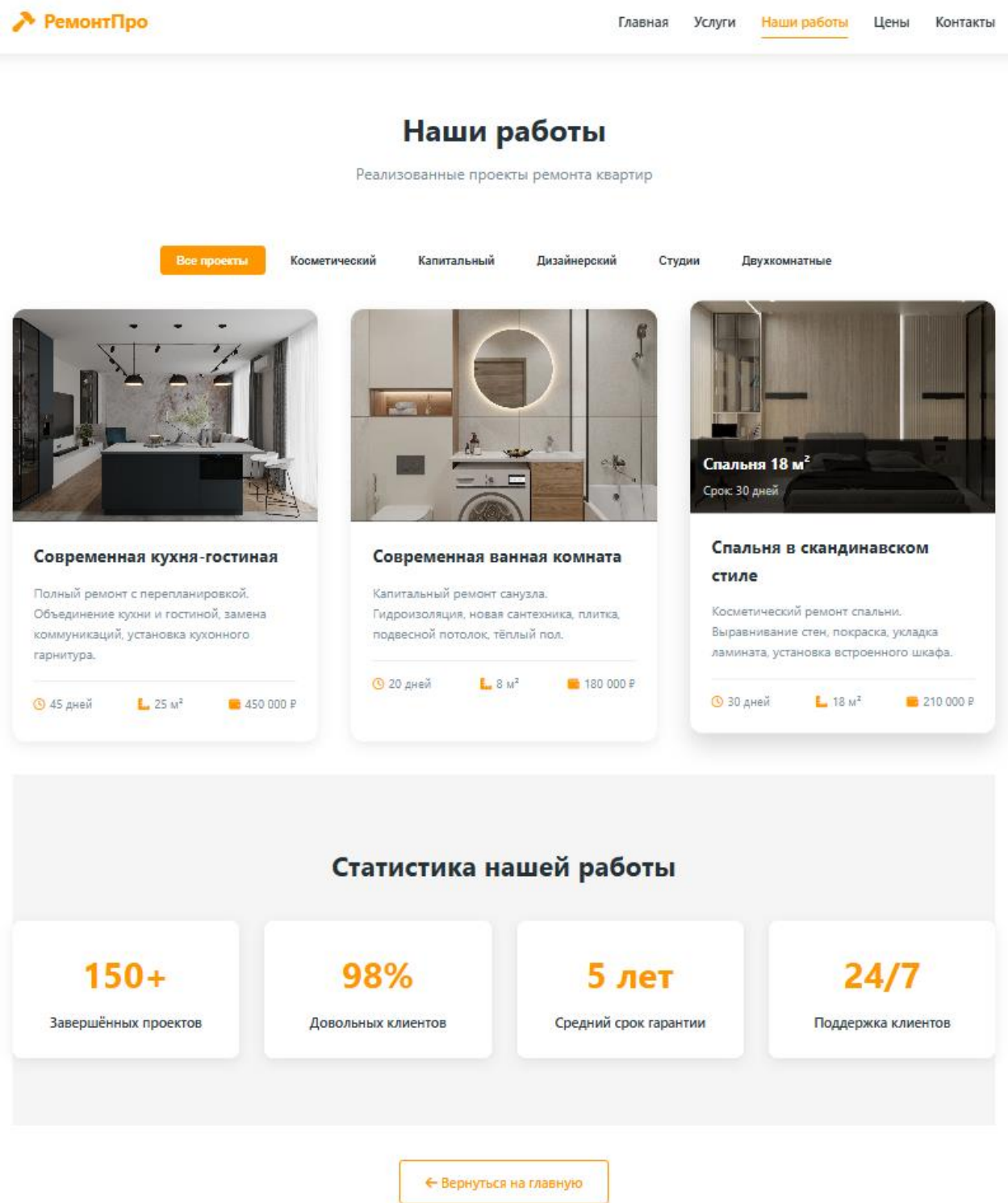
Вариант 11. Спортивный сайт.



Вариант 12. Сайт про здоровый образ жизни



Вариант 13. Сайт ремонта квартир.



Вариант 14. Блог о путешествиях.

TravelDiary

Главная

Блог

Направления

Советы


Контакты

Блог о путешествиях

Истории, советы и вдохновение из наших поездок по всему миру

Путеводитель

18.05.2023



Токио: Гид по самому технологичному городу

Токио - это город, где будущее уже наступило. От неоновых улиц Сибуя до спокойных садов Императорского дворца, этот город удивляет на каждом шагу. В этой статье я расскажу о must-visit местах, местной кухне и лайфаках для путешественников.

Япония


Токио

Азия

Читать полностью →

Истории

15.05.2023



Неделя в Тоскане: Вино, паста и бескрайние холмы

Тоскана - это не просто регион Италии, это состояние души. Проведя здесь неделю, я понял, почему люди возвращаются сюда снова и снова. Виноградники Кьянти, средневековые города и самая вкусная паста в мире - всё это ждёт вас в Тоскане.

Италия

Тоскана

Европа

Читать полностью →

Категории

Путеводители

12

Советы путешественникам

8

Истории из поездок

15

Бюджетные путешествия

6

Еда в путешествиях

9

Приключения

11

Популярные теги

Азия

Европа

Пляжный отдых

Горные походы

Городские путешествия

Еда

Культура

Природа

История

Фотография

Назад

1

2

3

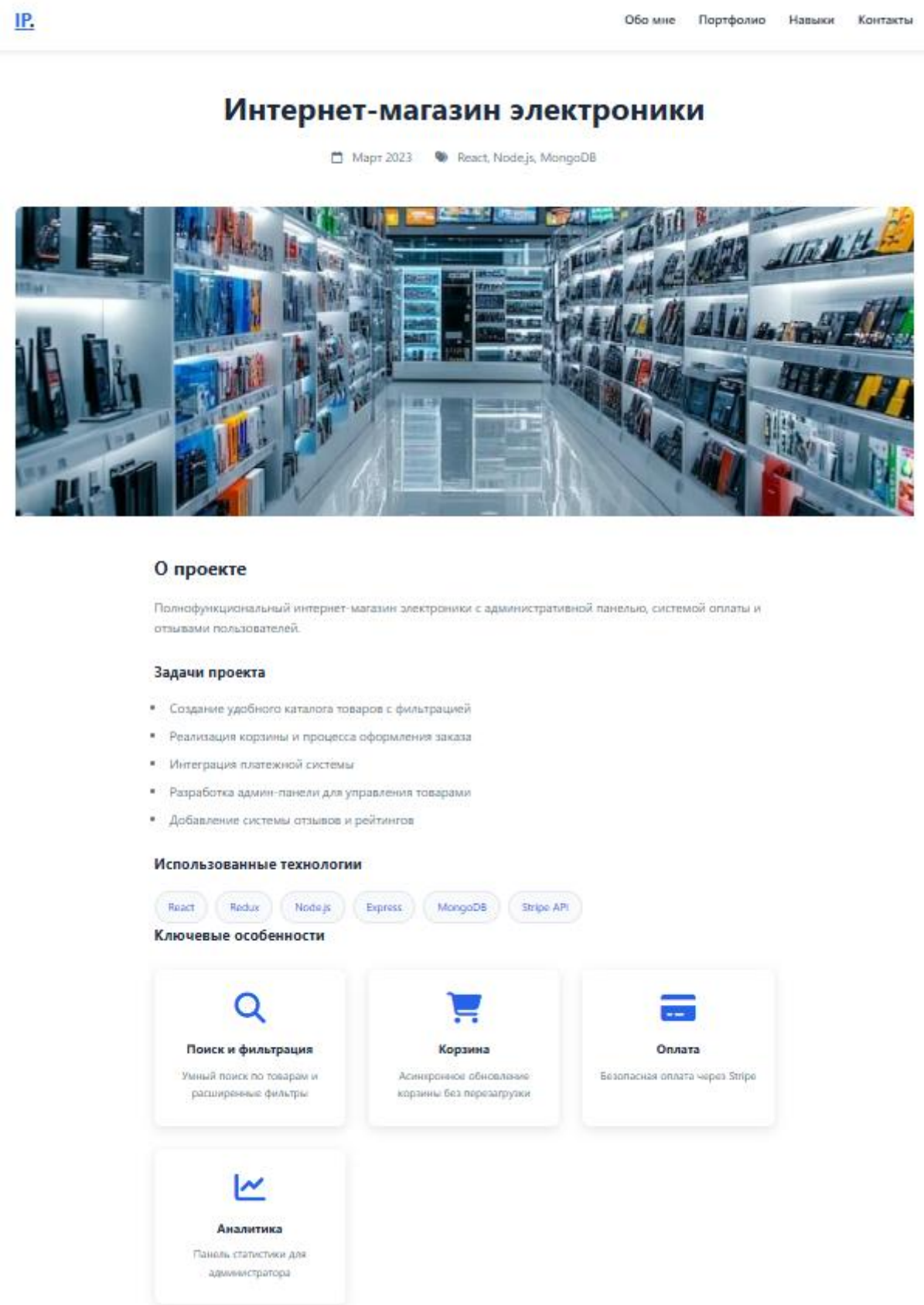
4

5

Вперёд

188

Вариант 15. Сайт-портфолио веб-разработчика.



Лабораторная работа № 26

«Индивидуальный проект: выбор темы и проектирование структуры сайта»

Цель работы: определить тему проекта, создать прототип и описание функциональных блоков.

Методические указания

Проектирование веб-сайта – первый и самый важный этап разработки, который определяет успех всего проекта. Этот этап включает анализ требований, определение целей сайта и создание прототипа.

Этапы проектирования:

1. Анализ цели сайта – для чего нужен сайт?
2. Определение целевой аудитории – кто будет пользоваться сайтом?
3. Создание информационной архитектуры – структура контента
4. Разработка прототипа – схематическое представление страниц
5. Определение функциональных требований – что должен уметь сайт?

Типы сайтов для выбора:

- Персональный сайт/портфолио – демонстрация навыков и работ
- Блог – публикация статей, записей
- Сайт-визитка компании – информация о компании, услугах
- Лендинг (посадочная страница) – продвижение одного продукта/услуги
- Интернет-магазин (упрощенный) – каталог товаров с описанием

Задания для выполнения лабораторной работы

1. Выберите тему проекта из предложенных типов сайтов
2. Определите основные цели вашего сайта:
 - Кто ваша целевая аудитория?
 - Какую проблему решает сайт?
 - Какие действия должен совершить пользователь?
3. Создайте прототип (wireframe) для 3-4 страниц:

- Главная страница
 - 2-3 внутренние страницы (например: "Обо мне/О компании", "Услуги/Портфолио", "Контакты")
 - Используйте инструменты: Figma, Adobe XD, или просто бумагу и карандаш.
4. Опишите функциональные блоки для каждой страницы:
- Навигационное меню
 - Шапка с логотипом
 - Основной контент (разбить на секции)
 - Подвал сайта
 - Контактная форма (если требуется)
5. Создайте карту сайта (sitemap) – визуальное представление структуры

Лабораторная работа № 27

«Индивидуальный проект: семантическая HTML-разметка всех страниц»

Цель работы: создать HTML-файлы для всех страниц проекта.

Методические указания

Семантическая HTML-разметка – использование HTML-тегов, которые несут смысловую нагрузку и описывают содержание контейнера. Это критически важно для:

- SEO – поисковые системы лучше понимают структуру сайта
 - Доступности (Accessibility) – скринридеры правильно интерпретируют контент
 - Поддержки кода – понятная структура облегчает работу с кодом
- Основные принципы:

1. Использование семантических тегов – <header>, <nav>, <main>, <section>, <article>, <aside>, <footer>
2. Правильная иерархия заголовков – последовательность <h1>-<h6>
3. Альтернативный текст для изображений – атрибут alt
4. ARIA-атрибуты для улучшения доступности
5. Логическая структура – контент должен быть понятен без CSS

Задания для выполнения лабораторной работы

1. Создайте HTML-файлы для всех страниц проекта.
2. Для каждой страницы создайте семантическую структуру:

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Название страницы | Название сайта</title>
  <link rel="stylesheet" href="css/style.css">
</head>
<body>
  <header class="header">
    <!-- Шапка сайта -->
  </header>
```

```
<main class="main">
  <!-- Основной контент с семантическими секциями -->
</main>

<footer class="footer">
  <!-- Подвал сайта -->
</footer>
</body>
</html>
```

3. Добавьте контент согласно вашему прототипу:

- Текстовый контент (заголовки, параграфы, списки)
- Изображения с атрибутами alt
- Ссылки навигации
- Формы (если есть в прототипе)

4. Проверьте доступность:

- У всех изображений есть alt
- Все интерактивные элементы доступны с клавиатуры

Лабораторная работа № 28

«Индивидуальный проект: базовая CSS-стилизация и создание общей сетки»

Цель работы: написать основной файл стилей. Определить общую сетку макета и стилей повторяющихся компонентов.

Методические указания

CSS (Cascading Style Sheets) – язык стилей, определяющий внешний вид HTML-документов. На этом этапе мы создаем основу для визуального представления сайта.

Ключевые концепции:

1. **Сетка макета (Layout Grid)** – организация элементов на странице
 - **Flexbox** – для одномерных макетов (строка или столбец)
 - **CSS Grid** – для двумерных макетов (и строки, и столбцы)
2. **CSS Custom Properties**– для хранения повторяющихся значений
3. **Методология БЭМ** – для именования классов
4. **Нормализация стилей** – сброс браузерных стилей

Задания для выполнения лабораторной работы

1. Создайте структуру CSS-файлов:

css/:

- base.css // Базовые стили, сброс, переменные
- layout.css // Сетка макета, общая структура
- components.css // Стили компонентов (кнопки, карточки)
- style.css // Главный файл, импортирующий все остальные

2. В base.css определите:

- CSS-переменные для цветов, шрифтов, отступов
- Сброс стилей (reset или normalize)
- Базовые стили для типографики

3. В layout.css создайте общую сетку:

- Стили для контейнера (.container)
- Общую структуру с помощью Flexbox или Grid

- Сетки для основных секций
4. Стилизируйте повторяющиеся компоненты:
- Шапку сайта (.header)
 - Навигационное меню (.nav)
 - Подвал сайта (.footer)
 - Общие кнопки (.btn)
5. Подключите все CSS-файлы в HTML:
- ```
<link rel="stylesheet" href="css/style.css">
```

## Лабораторная работа № 29

### «Индивидуальный проект: детальная стилизация и адаптивность»

**Цель работы:** завершить стилизацию всех страниц. Реализовать адаптивную верстку для мобильных устройств. Исправить ошибки и несоответствия.

### Методические указания

**Адаптивный веб-дизайн (Responsive Web Design)** – подход, при котором сайт корректно отображается на устройствах с разными размерами экрана.

Основные принципы:

1. **Mobile First** – проектирование сначала для мобильных устройств
2. **Гибкая сетка** – использование относительных единиц (%)
3. **Гибкие изображения** – max-width: 100%
4. **Медиа-запросы** – применение стилей в зависимости от характеристик устройства

Брейкпоинты (breakpoints):

```
/* Мобильные устройства */
@media (max-width: 767px) { /* ... */ }
/* Планшеты */
@media (min-width: 768px) and (max-width: 1023px) { /* ... */ }
/* Десктопы */
@media (min-width: 1024px) { /* ... */ }
```

### Задания для выполнения лабораторной работы

1. Завершите стилизацию всех страниц:
  - Стилизируйте все компоненты согласно дизайну
  - Добавьте состояния элементов (hover, focus, active)
  - Работайте с типографикой (шрифты, размеры, межстрочные интервалы)
2. Реализуйте адаптивность:
  - Создайте мобильную версию (до 767px)
  - Проверьте корректность отображения на всех разрешениях
3. Адаптируйте навигацию:

- Реализуйте бургер-меню для мобильных устройств
- Убедитесь, что меню доступно с клавиатуры

4. Оптимизируйте изображения:

- Используйте `max-width: 100%` для гибкости
- Добавьте `loading="lazy"` для отложенной загрузки

5. Исправьте найденные ошибки:

- Проверьте выравнивание элементов
- Устраните горизонтальную прокрутку
- Исправьте проблемы с переполнением контента

## Лабораторная работа № 30

### «Индивидуальный проект: финальная отладка, тестирование и презентация»

**Цель работы:** написать основной файл стилей. Определить общую сетку макета и стилей повторяющихся компонентов.

#### Методические указания

**Тестирование веб-сайта** – процесс проверки корректности работы сайта. Включает несколько типов тестирования:

1. **Функциональное тестирование** – проверка работы всех функций
2. **Кроссбраузерное тестирование** – проверка в разных браузерах
3. **Тестирование на разных устройствах** – проверка адаптивности
4. **Тестирование производительности** – скорость загрузки
5. **Тестирование доступности** – соответствие стандартам WCAG

Инструменты для тестирования:

- **Валидатор W3C** – проверка HTML и CSS
- **Lighthouse (Chrome DevTools)** – проверка производительности, доступности, SEO
- **BrowserStack** – тестирование в разных браузерах и на разных устройствах
- **Screen Readers** – тестирование доступности (NVDA, VoiceOver)

#### Задания для выполнения лабораторной работы

1. Проверьте код на валидность:
  - HTML-валидатор
  - CSS-валидатор
  - Исправьте все найденные ошибки
2. Протестируйте в разных браузерах.
3. Протестируйте на разных устройствах:
  - Мобильные телефоны (разные размеры экрана)
  - Десктопы (разные разрешения)
4. Проверьте производительность:

- Оптимизируйте изображения
- Минимизируйте CSS

5. Подготовьте презентацию проекта:

- Создайте краткую презентацию (3-4 слайда)
- Расскажите о сайте и для кого он будет полезен
- Продемонстрируйте ключевые особенности сайта
- Расскажите о технических решениях и проблемах, с которыми столкнулись

## БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. **Никсон, Р.** Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и HTML5 / Р. Никсон. – 7-е изд. – Санкт-Петербург : Питер, 2025. – 832 с. – (Серия «Бестселлеры O'Reilly»).
2. **Роббинс, Дж.** HTML5, CSS3 и JavaScript. Исчерпывающее руководство / Дженнифер Роббинс; [пер. с англ. М. А. Райтман]. – 4-е изд. – Москва : Эксмо, 2014. – 528 с.
3. **Мейер, Эрик А.** CSS. Карманный справочник / Эрик А. Мейер ; [пер. с англ.]. – 5-е изд. – Москва : ООО «И.Д. Вильямс», 2019. – 208 с.
4. **Фримен, Э.** Изучаем HTML, XHTML и CSS / Эрик Фримен, Элизабет Фримен. – 2-е изд. – Санкт-Петербург : Питер, 2024. – 720 с.
5. **Макфарланд, Д.** Новая большая книга CSS / Д. Макфарланд. – СанктПетербург : Питер, 2025. – 720 с. – (Серия «Бестселлеры O'Reilly»).
6. **Ломаш, Д. А.** Интернет-технологии и мультимедиа : учебное пособие / Д. А. Ломаш, О. Г. Ведерникова ; ФГБОУ ВО РГУПС. – Ростов-на-Дону, 2017. – 119 с.

*Учебное издание*

**Муконина Мария Ильинична**

**Основы web-разработки и верстки**

Печатается в авторской редакции  
Технический редактор Т. И. Исаева

Подписано в печать . Формат 60×84/16.  
Усл. печ. л. 2,32. Тираж 100 экз. Изд. №. Заказ .

Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Ростовский государственный университет путей сообщения»  
(ФГБОУ ВО РГУПС)

---

Адрес университета: 344038, Ростовская область, г. о. город Ростов-на-Дону,  
г. Ростов-на-Дону, пл. Ростовского Стрелкового Полка  
Народного Ополчения, зд. 2, [www.rgups.ru](http://www.rgups.ru)