МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ  
Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ   
ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий

Кафедра Информационных систем и технологий

Специальность 1-40 05 01 «Информационные системы и технологии»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА КУРСОВОГО ПРОЕКТА**

по дисциплине «Компьютерные языки разметки»

Тема: Web-сайт «Шаблоны веб-страниц»

**Исполнитель**

студент 1 курса 2 группы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. А. Багласов

подпись, дата

**Руководитель**

преподаватель-стажер \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Л. П. Степанова

должность, ученая степень, ученое звание подпись, дата

Допущен к защите \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

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

дата, подпись

Курсовой проект защищен с оценкой

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Л. П. Степанова

подпись дата инициалы и фамилия

**Содержание**

Введение 4

1 Постановка задачи 5

1.1 Обзор аналогичных решений 5

1.2 Техническое задание 8

1.3 Выбор средств реализации программного продукта 9

1.4 Выводы 10

2 Проектирование страниц веб-сайта 11

2.1 Выбор способа верстки 11

2.2 Выбор стилевого оформления 11

2.3 Выбор шрифтового оформления 11

2.4 Разработка логотипа 11

2.5 Разработка спецэффектов 11

2.6 Выводы 11

3 Реализация структуры веб-сайта 12

3.1 Описание верстки и её реализация 12

3.2 Использование XML и SVG 15

3.3 Выводы 16

4 Тестирование веб-сайта 17

4.1 Адаптивный дизайн веб-сайта 17

4.2 Кроссбраузерное тестирование 18

4.3 Руководство пользователя 20

4.4 Валидность 23

4.5 Выводы 24

Заключение 25

Список использованных источников 26

ПРИЛОЖЕНИЕ А Макеты страниц веб-сайта 27

ПРИЛОЖЕНИЕ Б Макет структуры веб-сайта 28

ПРИЛОЖЕНИЕ В Листинг кода для работы с xml 29

ПРИЛОЖЕНИЕ Г Листинг XML-файлов 30

**Введение**

В настоящее время достаточно тяжело найти простой, гибкий и быстрый шаблон, с которого можно было бы и начать свою работу. Испытывая трудности с поиском такого шаблона, что не имел бы дополнительных декораций и лишнего контента, в котором пришлось бы дополнительной разбираться и удалять большую часть имеющегося там кода и контента, чтобы перестроить шаблон под свои нужды, появилась идея данного проекта. Большая часть современных сайтов с шаблонами на сегодняшний день предоставляют шаблоны с дополнительным оформлением и огромным количеством ненужного для пользователя контентом.

Многие разработчики, особенно те, кто только начинают свой путь на этом поприще, чаще всего не испытывают энтузиазма при работе с шаблонами, перегруженными огромным количеством лишней информации, кода и контента. Они предпочли бы быстрый старт при помощи простого шаблона, желательно на одну страницу, с базовой гибкой версткой, с которой можно быстро разобраться и быстро начать работать, адаптируя её под свои нужды.

Цель курсовой работы: разработать сайт, с помощью которого пользователь мог бы получить доступ к простому шаблону веб-страницы или отправить свой шаблон, чтобы после модерации им могли пользоваться другие.

Задачи курсовой работы:

1. проанализировать сайты с похожей тематикой, выявив их положительные и отрицательные качества;
2. разработать веб-сайт для выбора базового шаблона с использованием HTML5, XML, JS, CSS3, а так же библиотеки PrismJS[1];
3. провести тестирование сайта, проверить его адаптивность и кроссбраузерность.
4. **Постановка задач и обзор аналогичных решений**

## 1.1 Обзор аналогичных решений

Прежде чем браться за разработку, следует узнать, а не реализован ли необходимый функционал сайта уже где-то ещё, какие сайты максимально близки к реализации поставленного функционала и какие у них сильные и слабые стороны. Данный шаг нам необходим для того, чтобы научиться на чужих ошибках, подсмотреть некоторые идеи или придумать что можно улучшить или полностью убрать.

Для смотра были подобраны сайты: nicepage.com [2], freehtmlthemes.ru [3], html-templates.info [4], psd-html-css.ru [5].

Первым рассмотрим сайт nicepage.com, представленный ниже на рисунке 1.1.

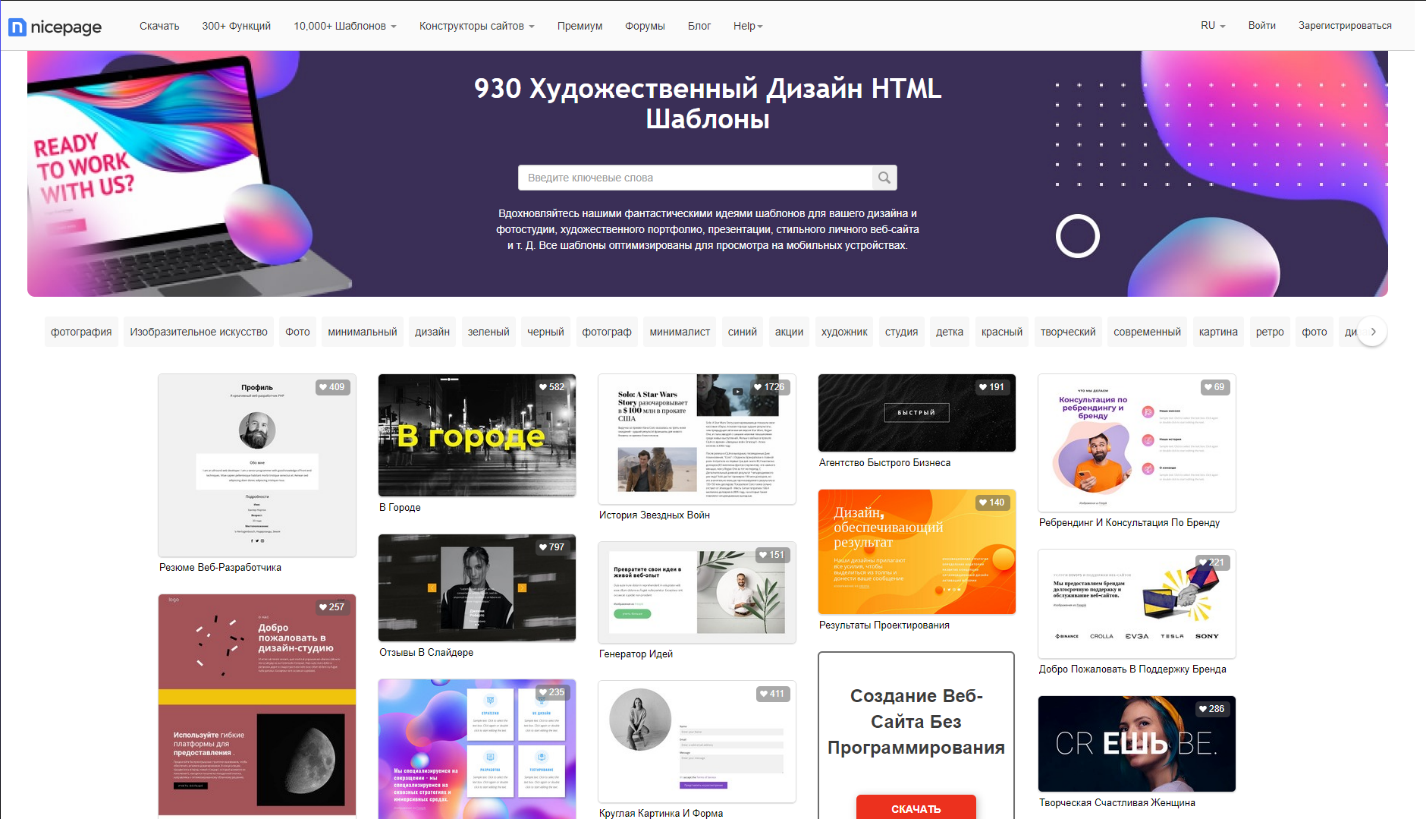


Рисунок 1.1 – Оформление сайта «nicepage.com»

Данный сайт имеет простую цветовую гамму, гибкую верстку и приятный список, однако такая структура гибкая списка приводит к тому, что отмеченный вами элемент быстро теряется и его сложно повторно открыть. Шаблоны, предоставляемые сайтом, являются сравнительно простыми в плане своей структуры и контента, однако количество дополнительных декораций делает их очень неудобным для восприятия и редактирования.

По итогу были выявлены следующие преимущества:

1. цветовая гамма;
2. гибкая верстка;
3. приятный вид списка.

К недостаткам в функционале веб-сайта можно отнести неудобный поиск по списку.

Вторым рассмотрим сайт freehtmlthemes.ru, представленный ниже на рисунке 1.2.

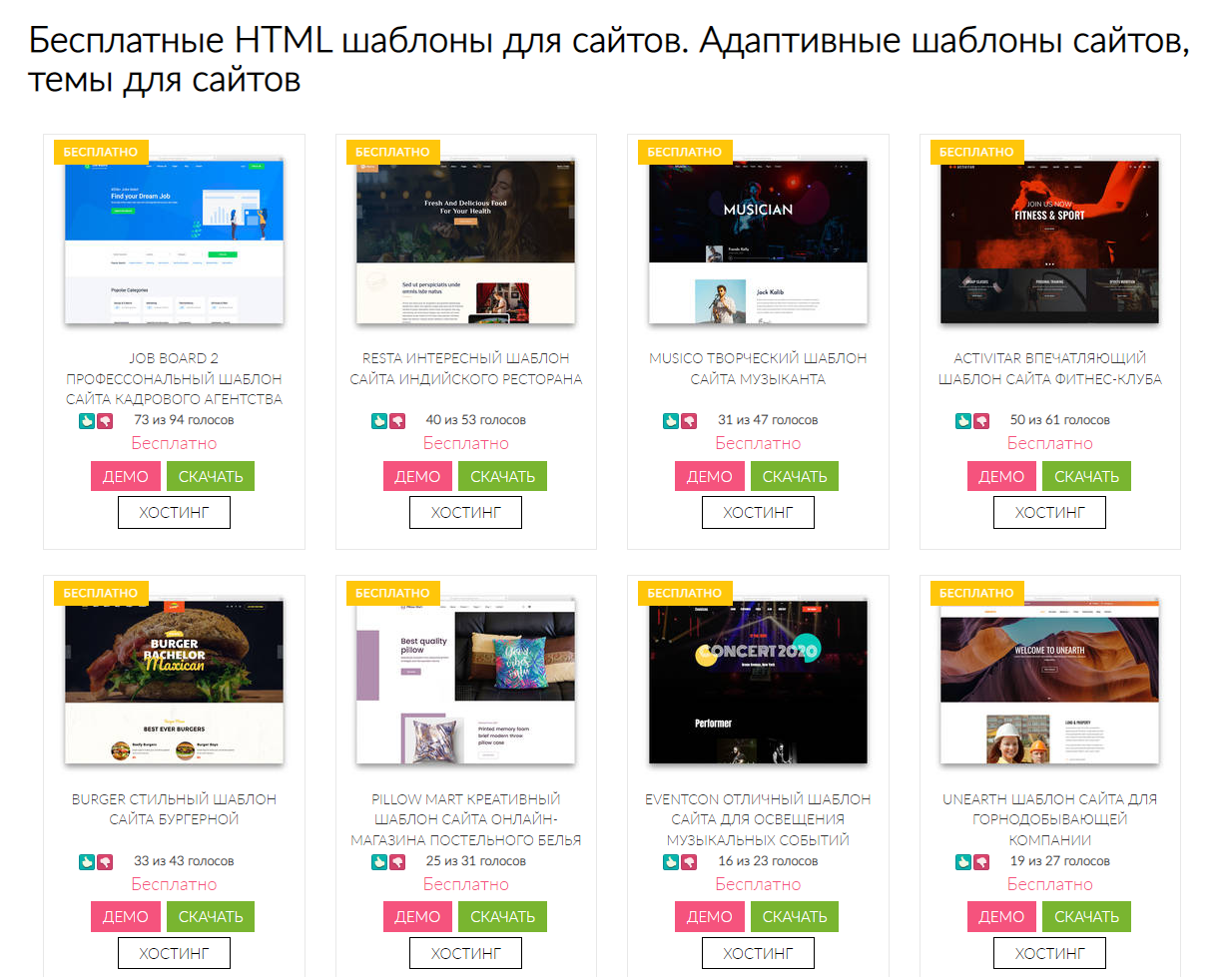


Рисунок 1.2 – Оформление сайта «freehtmlthemes.ru»

Данный сайт так же имеет простую цветовую гамму, а список реализован намного удобней, хотя и имеет недостаток в виде тонкого и потому невнятного текста и нечеткого предпросмотра шаблона.

По итогу были выявлены следующие преимущества:

1. цветовая гамма;
2. удобный поиск по списку.

К недостаткам в функционале веб-сайта можно отнести:

1. невнятный текст;
2. нечеткий предпросмотр шаблона.

Третьим рассмотрим сайт html-templates.info, представленный ниже на рисунке 1.3.

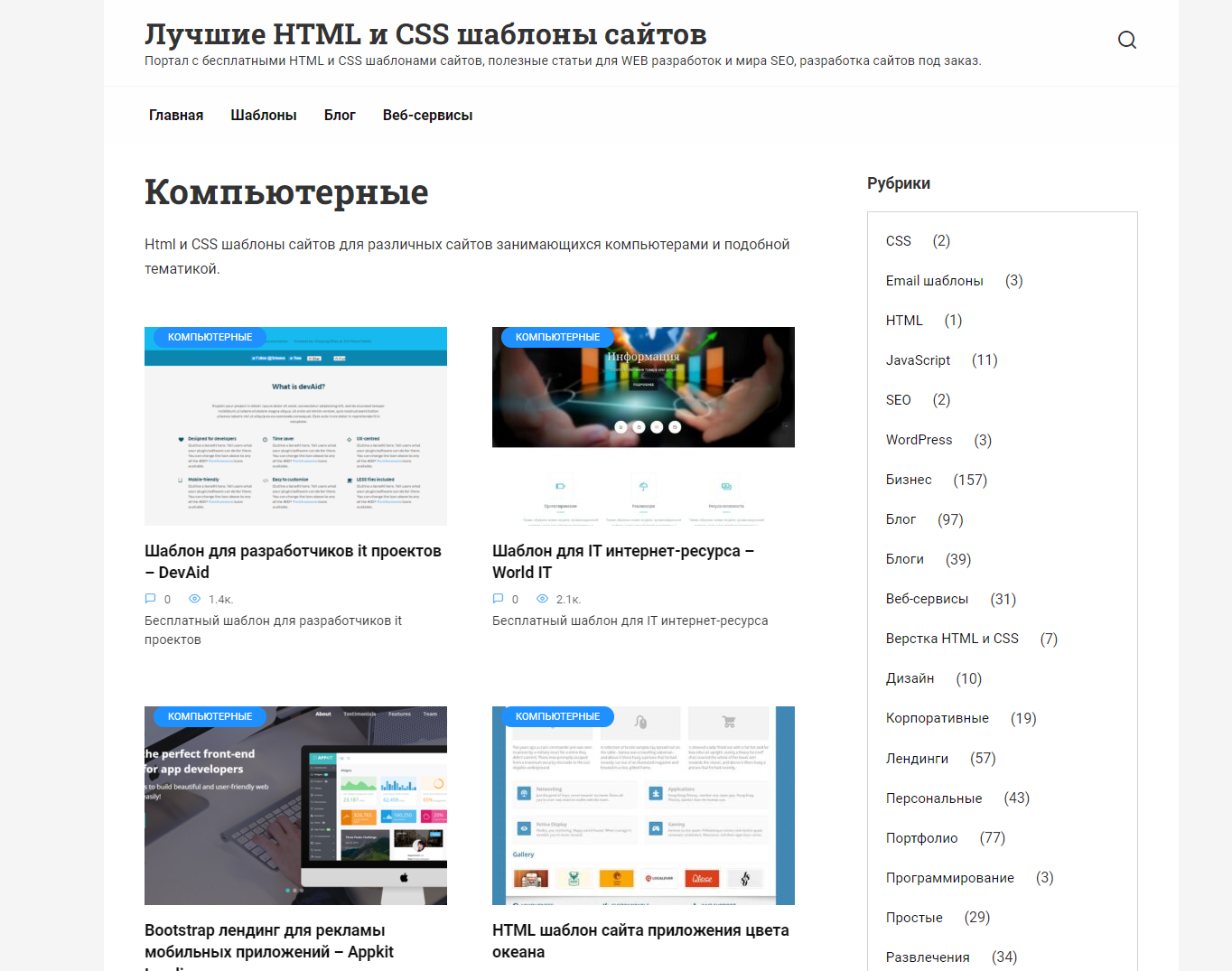


Рисунок 1.3 – Оформление сайта «html-templates.info»

На данном сайте виден четкий и понятный текст, информативные изображения для предпросмотра, однако 1/3 веб-страницы занимает область “Рубрики”, в которой осуществляется упорядочивание, что слишком много.

По итогу были выявлены следующие преимущества:

1. читаемость элементов;
2. удобство при использовании.

К недостаткам в функционале веб-сайта можно отнести то, что часть страницы для упорядочивания занимает слишком много.

Последним рассмотрим сайт psd-html-css.ru, представленный ниже на рисунке 1.4.

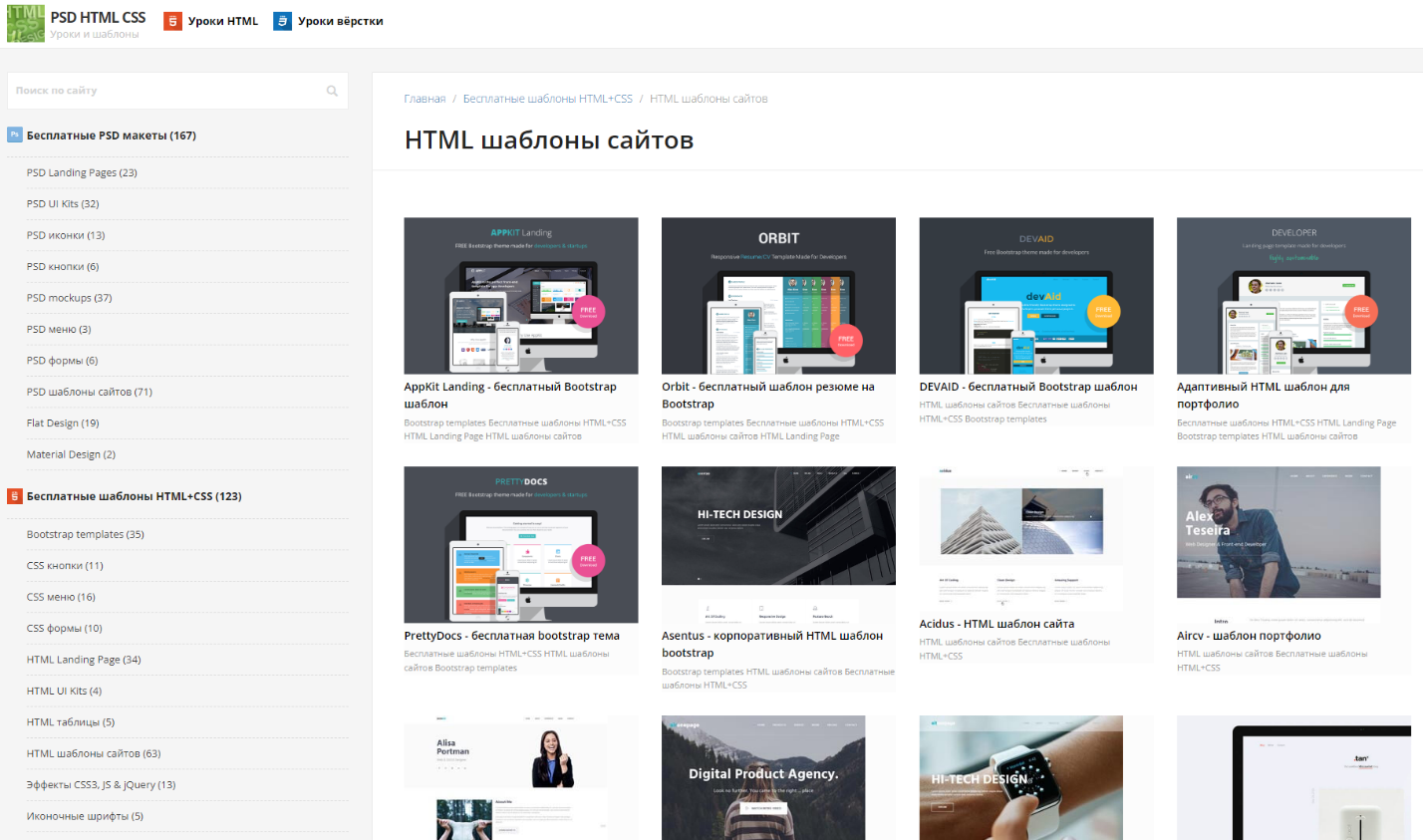


Рисунок 1.4 – Оформление сайта «psd-html-css.ru»

Здесь наблюдается простая, удобная и легко читаемая разметка, где тяжело потеряться и с которой приятно и удобно работать.

По итогу были выявлены следующие преимущества:

1. читаемость элементов;
2. удобство при использовании;
3. удобный поиск по списку.

Однако, все эти сайты объединяет то, что шаблоны на этих сайтах заполнены большим количеством лишнего контента и адаптации под него, из-за чего при работе будут возникать трудности в виде нужды разбираться в большом количестве кода с лишним контентом.

* 1. **Техническое задание**

В данной работе будет создан веб-сайт, состоящий из четырех страниц.

Первая страница является главной, она предоставляет базовую информацию о сайте, рассказывает о том, куда переходить и что делать, а также содержит информацию о принципах сайта.

Вторая страница представляет собой список шаблонов. Содержит ячейки, имеющие название шаблона, её автора, путь к изображению с превью и краткое описание. Создание ячеек происходит в соответствии с XML документом, который считывает JavaScript, что создает ячейки и помещает их под flex-контейнер. При нажатии на ячейку открывается страница просмотра шаблона, при этом страница открывается с пометкой индекса для шаблона, через которое и загружается нужный шаблон из XML документа.

Третья страница представляет собой просмотр шаблона. На ней выводится информация из XML документа о конкретном шаблоне, включая: название, автора, краткое описание, длинное описание, код HTML, код CSS, код JS, а также функция предпросмотра шаблона в отдельном окне, при помощи функций window.open() и document.write(), перед которыми компилируем код HTML, CSS и JS в один HTML код.

Четвертая страница содержит форму для отправки данных с пользовательским шаблоном, который в будущем могут добавить в список. Содержит ввод: названия шаблона, именование автора, контактную информацию (почту), файлы HTML, CSS, JS, изображение превью, краткое описание шаблона (с ограничением в 200 символов) и длинное описание.

В качестве стиля будет использован минималистичный стиль с деревянными тонами. Основные цвета: различные оттенки древесины, белый и черный. Для оформления кусков кода будет использована библиотека PrismJS.

* 1. **Выбор средств реализации программного продукта**

Для реализации проекта были выбраны языки: HTML, CSS, JavaScript. Структура сайта создана с помощью языка разметки HTML, дизайн сайта оформлен с помощью CSS. Для чтения XML, предпросмотра шаблона, заполнения контентом страниц со списком шаблоном и просмотром шаблона, использовался JavaScript, а для разметки синтаксиса кода на странице с просмотром шаблона использовалась библиотека PrismJS.

HTML (HyperText Markup Language) — стандартизированный язык гипертекстовой разметки документов для просмотра веб-страниц в браузере. Используется для размещения элементов по веб-странице, путем написания HTML-кода пользователем, а затем считыванием этого кода браузером, что согласно ему и размещает элементы по веб-странице.

CSS (или Cascading Style Sheets) — формальный язык описания внешнего вида документа, написанного с использованием языка разметки.

JavaScript — мультипарадигменный язык программирования. Поддерживает объектно-ориентированный, императивный и функциональный стили. Является реализацией спецификации ECMAScript. JavaScript обычно используется как встраиваемый язык для программного доступа к объектам приложений.

PrismJS - это легкий, расширяемый синтаксический инструмент для подсветки синтаксиса кода, созданный с учетом современных веб-стандартов. Имеет поддержку множества языков.

Код проекта был написан в редакторе кода Visual Studio Community 2019. Данный редактор предоставляет удобные инструменты для работы с HTML, CSS и JS, осуществляет автоматические переносы строк и расстановку табуляции, ускоряет написание кода при помощи всплывающих подсказок и навигации по коду, и так далее. Visual Studio так же имеет приятный и интуитивный интерфейс, благодаря чему в нем приятно работать.

Для размещения ячеек в списке шаблонов используется новая технология Flexbox, имеющая широкую поддержку современных браузеров.

В качестве основного языка для сайта используется русский.

Для доступа к файлу XML как и эмуляция локального сервера через npm, так и используется запуск браузера в небезопасном режиме, позволяя избежать нужды с запуском эмуляции сервера для доступа к файлу XML.

* 1. **Выводы**

В ходе написания данного пункта были рассмотрены альтернативы данному проекту. Было выявлено, что популярной альтернативы, что предоставляла бы доступ к простым шаблонам, нет, подтверждая актуальность темы данного сайта. Также была осознана важность четкого и простого дизайна для обеспечения лучшей читаемости пользователем.

Было принято решение использовать язык разметки HTML для создания структуры веб-сайта, CSS для его внешнего оформления, создать логотип в формате SVG, для чтения сайта и заполнения контентом определенный областей использовать JavaScript, а также веб-сайт будет взаимодействовать с XML документом при помощи JavaScript получая из него данные, а код будет помечаться при помощи PrismJS.

**2 Проектирование страниц веб-сайта**

**2.1 Выбор способа верстки**

Для того, чтобы сайт был удобным для пользователей различных устройств необходима адаптивность. Для этого будут использованы “@media screen and (min/max-width: 1200px)” для обеспечения более приятного и удобного внешнего вида для людей, как и с широкими, так и малыми экранами, а также проценты для указания размеров некоторых компонентов и Flexbox вёрстка для расстановки ячеек таблиц. С её помощью можно очень легко расставить элементы в контейнере, распределить доступное пространство между ними и выровнять.

**2.2 Выбор стилевого оформления**

Как уже было сказано, в качестве стиля будет использоваться минималистичный стиль с деревянными тонами. Основными цветами станут различные оттенки древесины, белый и черный. Для оформления кусков кода будем использовать библиотеку PrismJS. Так же добавим подсветку кнопок при наведении в качестве простого, но приятного эффекта.

**2.3 Выбор шрифтового оформления**

Выбора шрифта пал на стандартный шрифт Times New Roman, являющимся шрифтом по умолчанию, что вкупе с контрастным цветом шрифта и фона, будет выделять текст, обеспечивая хорошую читаемость.

**2.4 Разработка логотипа**

В качестве логотипа был выбран узор из трех документов идущих снизу вверх, справа налево, где первый видно полностью, второй скрыт на 25 %, третий скрыт на 50 %. Для создания логотипа будет использован SVG.

**2.5 Разработка спецэффектов**

Спецэффекты будет представлены плавным изменением цвета при наведении на кнопки, для чего используется атрибут “transition-duration”. Для кнопок эффект представляет собой инверсию цветов: с черного на белый, с белого на черный. Для ячейки списка шаблонов эффект представляет собой изменение цвета фона с белого на серый.

**2.6 Выводы**

На основе данной главы были разработаны макет страниц сайта в Figma, представленный в приложении А на рисунке А.1, и создан макет структуры веб-сайта, представленный в приложении Б на рисунке Б.1.

**3 Реализация структуры веб-сайта**

## 3.1 Описание верстки

Создание сайта – долгая и требующая правильного разделения на этапы работа. Разделение на этапы позволяет работать избежать лишней работы и поддерживать организацию. И первым этапом будет создания сайта будет создание верстки сайта. Верстка — это описание визуальной части сайта с помощью гипертекстового документа на основе HTML-разметки.

Этот этап разделим на несколько этапов:

1. разработка страниц: HTML-верстка с подключением CSS;
2. создание и подключение скриптов;
3. разработка шаблона для демонстрации.

Код JS, XML, CSS и HTML будет храниться в корневой директории, так как самих файлов будет немного, всего 13 штук, и, следовательно, отдельную папку под них выделять необязательно. Основной графический контент, как и дополнительная библиотека PrismJS, будет расположен в папке assets, в то время как графический контент для шаблонов, а точнее изображения превью для них, будут находится в папке preview**\_**img. В качестве основного формата изображений используется формат png.

Всего у нас четыре веб-страницы: главная, шаблоны, просмотр шаблона, отправка шаблона. Есть ещё и предпросмотр шаблона, однако он реализован через JavaScript, потому пока что его не затронем. Под каждую из этих страниц будет выделен HTML документ с CSS. Ввиду наличия общего стиля и общей верстки, было принято решение сделать пятый CSS файл, в который будет загружена основная верстка, общая для каждой страницы. Базовая разметка представлена ниже на рисунках 3.1 и 3.2.

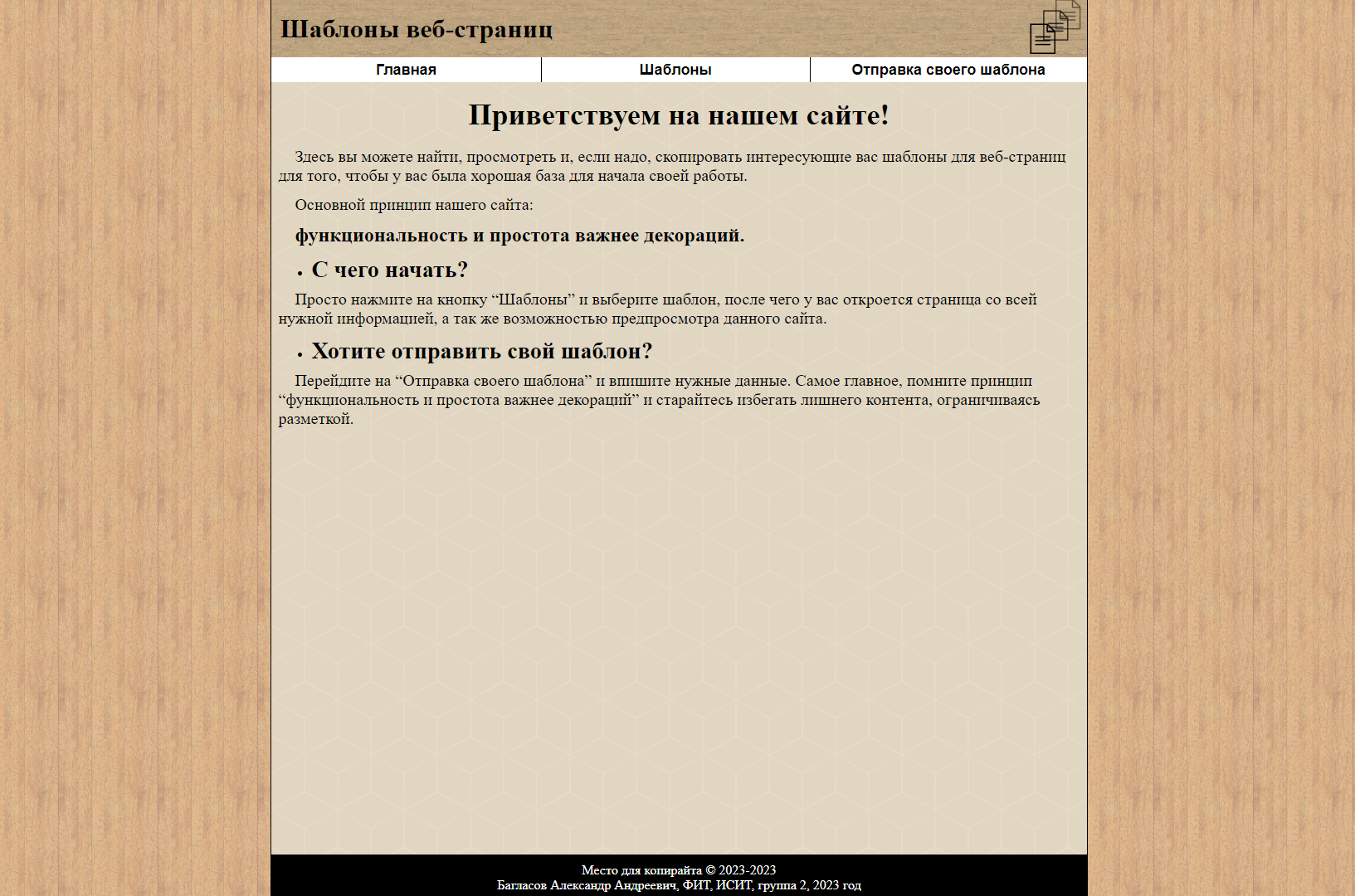


Рисунок 3.1 – Базовая верстка под экраны с шириной более 1200 пикселей

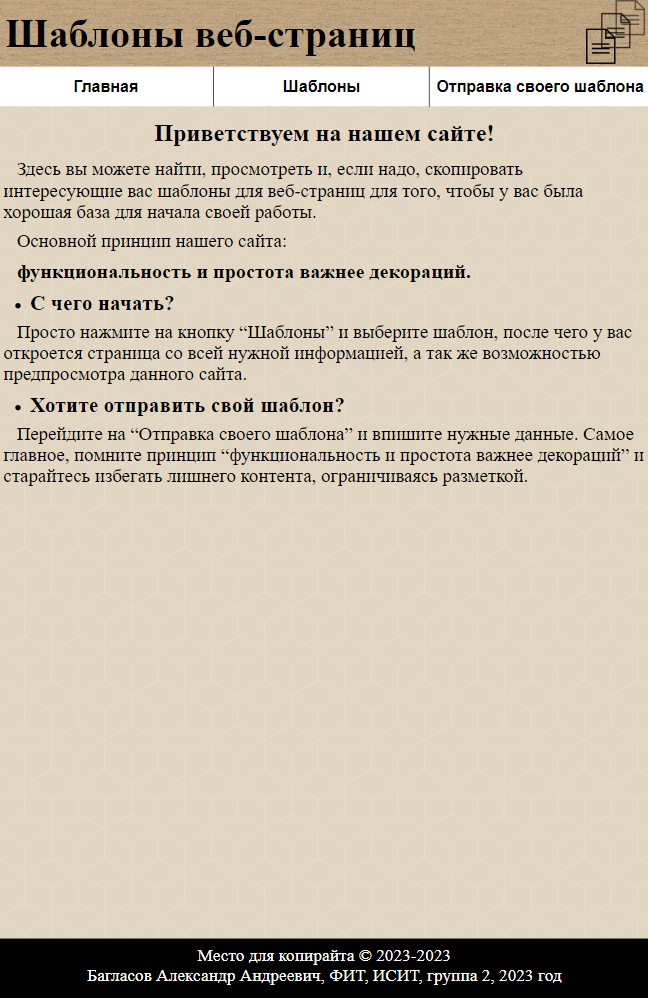


Рисунок 3.2 – Базовая верстка под экраны с шириной меньше 1200 пикселей

Страница поделена на три части: шапка (header), располагающаяся сверху, в которой располагается название сайта, его логотип и кнопки перехода на другие страницы; подвал (footer), располагающийся снизу, в котором располагается место для контактной почты и копирайта, ведь если посмотреть на другие сайты, то такие места есть на всех сайтах и они почти всегда располагаются внизу; контент, располагающийся между ними – основное наполнение страницы. Так же по бокам для широкоформатный экранов можно заметить наличие пустых полос. Данные полосы служат для удобства пользователям с широкоформатными экранами, позволяя контенту не расплываться слишком сильно, поддерживая сайт в компактном состоянии.

Для достижения данного эффекта было использовано at-правило “@media screen and (min-width: 1200px)” и “@media screen and (max-width: 1200px)”.

Дальше рассмотрим просмотр списка шаблонов, представленный ниже на рисунке 3.3.

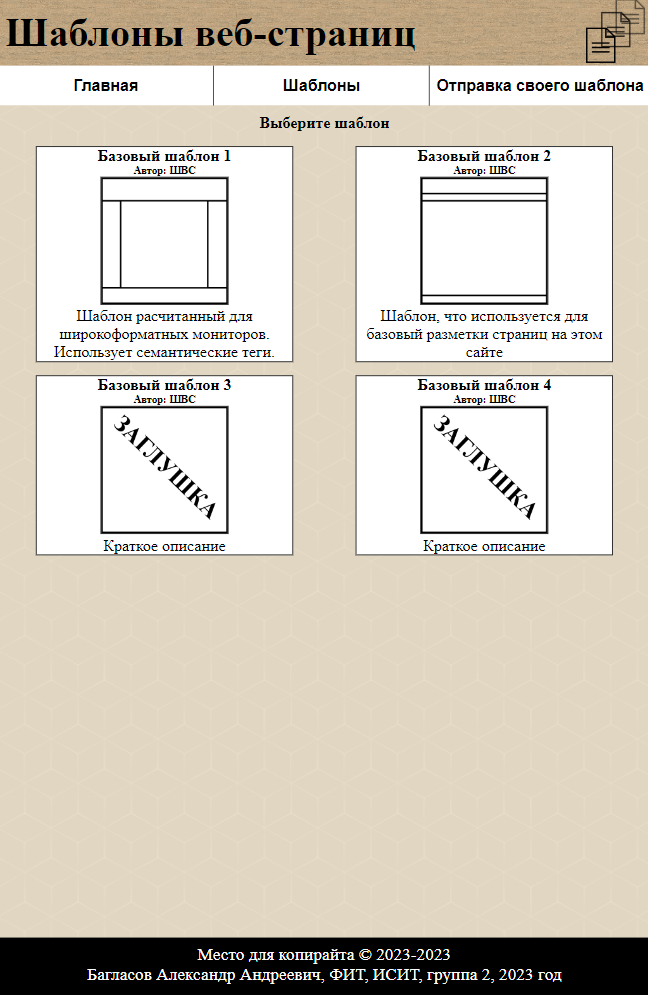


Рисунок 3.3 – Верстка страницы списка шаблонов, представленная в вертикальном варианте.

Для реализации данной страницы был использован Flexbox со следующими свойствами: расположение в несколько линий; равномерное распределение всех элементы по главной оси, в данном случае – по горизонтальной оси.

Заполнение происходит путем чтения XML документа при помощи JavaScript, который создает ячейки, заполняет их контентом и добавляет в качестве дочернего объекта к flex-контейнеру. Заполнение данной таблицы данными будет приведено в следующей главе.

Следующей страницей идет страница просмотра шаблона. На данной странице выводятся данные выбранного шаблона. Указание на нужный элемент в XML документе происходит через параметр в URL ссылке, указывающий на номер элемента. Код для заполнения представлен в следующей главе. Для оформления кода используется библиотека PrismJS. Пример внешнего вида страницы просмотра шаблона представлен на рисунке 3.4.

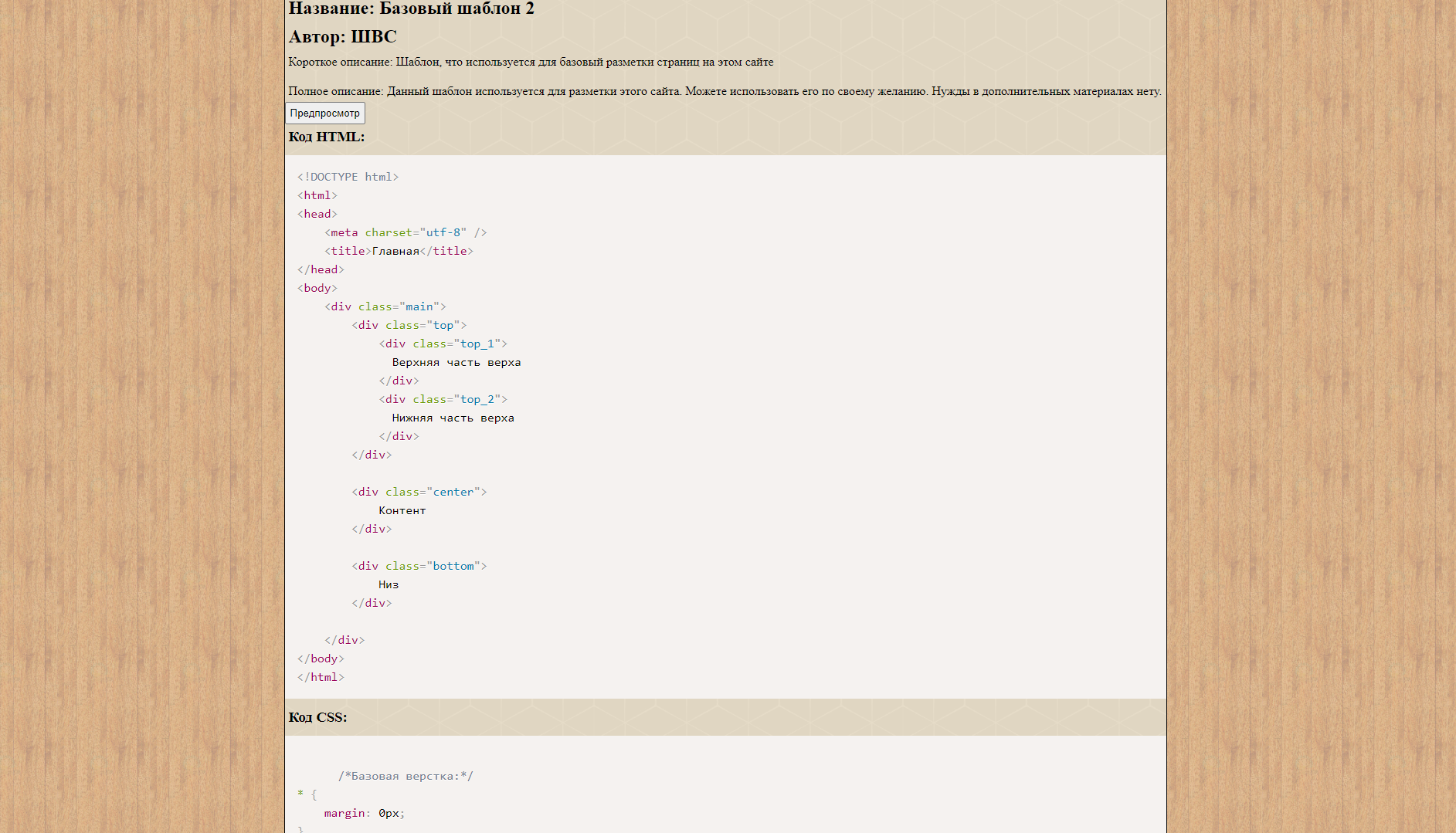


Рисунок 3.4 – Верстка страницы просмотра шаблонов.

## Использование XML и SVG

Для хранения данных о шаблоне, используется XML документ, каждый элемент в котором включает в себя: название, автора, короткое описание, длинное описание, ссылку на файл изображения превью, код HTML, код CSS, код JS. Код для размещения данных на странице со списком приведен на листинге 3.1.

|  |
| --- |
| let xmlContent = ‘’;  let putElementsHere = document.getElementById(‘putElementsHere’);  fetch(‘xml.xml’).then((response) => {  response.text().then((xml) => {  xmlContent = xml;  let parser = new DOMParser();  let xmlDOM = parser.parseFromString(xmlContent, ‘application/xml’);  let cells = xmlDOM.querySelectorAll(‘element’);  let I = 0;  cells.forEach(xmlElement => {  let div = document.createElement(‘a’);  div.href = “view.html?id=” + I;  i++;  let element = document.createElement(‘h2’);  element.innerText = xmlElement.children[0].innerHTML;  div.appendChild(element);  element = document.createElement(‘h4’);  element.innerText = “Автор: “ + xmlElement.children[1].innerHTML;  div.appendChild(element);  element = document.createElement(‘img’);  element.className = “preview”;  element.src = “preview\_img/” + xmlElement.children[3].innerHTML;  div.appendChild(element);  element = document.createElement(‘p’);  element.innerText = xmlElement.children[2].innerHTML;  div.appendChild(element);  putElementsHere.appendChild(div);  });  });  }); |

Листинг 3.1 – Чтение XML файла и заполнение списка данными

Здесь получен доступ к файлу, получен набор элементов, из каждого элемента берутся данные и размещаются в будущую ячейку, а затем ячейка размещается на странице в качестве дочернего объекта к flex-контейнеру. Ввиду использования Flexbox вёрстки, данные ячейки будут динамически переставляться, в зависимости от размеров контейнера. Так же к URL ссылке ячейки ставится указание на HTML документ с параметром id=X, где X – номер элемента в XML документе). Параметры в URL — это последовательность символов, расположенных после адреса ресурса в интернете. Для их отделения от основного URL используется знак вопроса. Каждый параметр представляет собой пару символов, разделенную знаком “=”.

Код для заполнения данными страницы с просмотром шаблона представлен в приложении Г.

Получаем доступ к файлу, из URL адреса извлекается параметр с номером нужного адреса, после чего начинаем обращаться к элементу из списка с номером (индексом), взятым из параметра.

SVG был использован для создания логотипа, состоящего из трех документов идущих друг за другом по диагонали, с видимостью 100, 75 и 50 процентов соответственно.

**3.3 Выводы**

В ходе данной главы была разработана верстка сайта и стилевые решения сайта. Верстака сайта выполнялась с учетом необходимости наличия адаптивности, была разработана структура xml файлов, код для загрузки данных из xml файлов.

**4 Тестирование веб-сайта**

# **4.1 Адаптивный дизайн веб-сайта**

Для обеспечения работы сайта на мобильных устройствах использовались медиа-запросы (@media), что меняют верстку сайта в зависимости от ширины экрана. Само тестирование было проведено при помощи инструментов браузеров Google Chrome, Firefox и Opera GX.

Пример оформления для ПК версии и для телефонной версии представлены на рисунках 4.1 и 4.2.

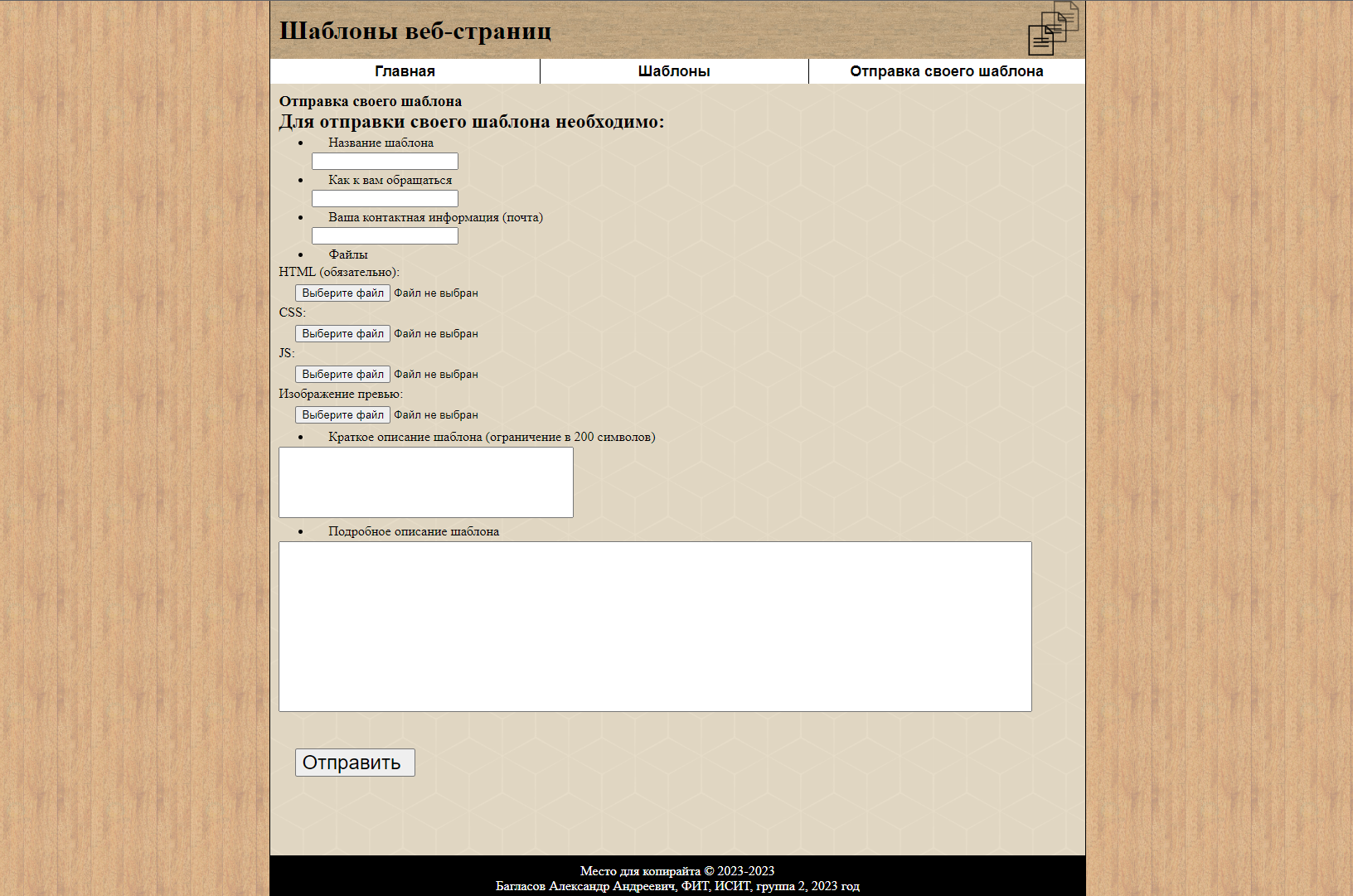


Рисунок 4.1 – Страница отправки шаблона для ПК версии

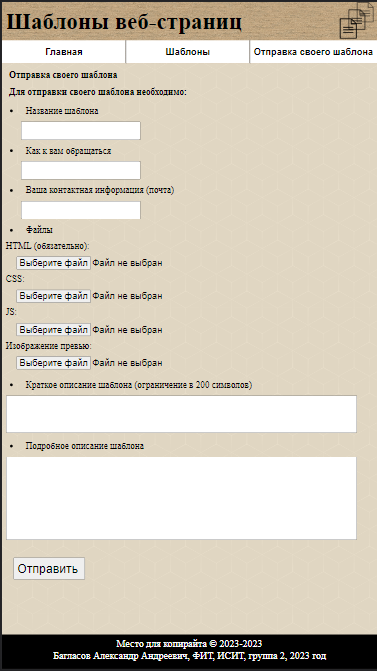


Рисунок 4.2 – Страница отправки шаблона для телефонной версии

# **4.2 Кроссбраузерное тестирование**

Кросс-браузерное тестирование — это практика обеспечения уверенности в том, что веб-сайты и веб-приложения, которые вы создаёте, работают в приемлемом количестве браузеров. Обязанность веб-разработчика быть уверенным, что проект работает не только у вас, а у всех ваших пользователей, независимо от браузера, устройства, или других вспомогательных инструментов, которые они используют.

Кроссбраузерное тестирование было проведено на основе браузеров Google Chrome и Mozilla Firefox. В ходе тестирования было выявлено лишь одно отличие, в виде разного рендеринга компонента “<input type=’file’>”, что можно увидеть на рисунках 4.3 и 4.4 соответственно.

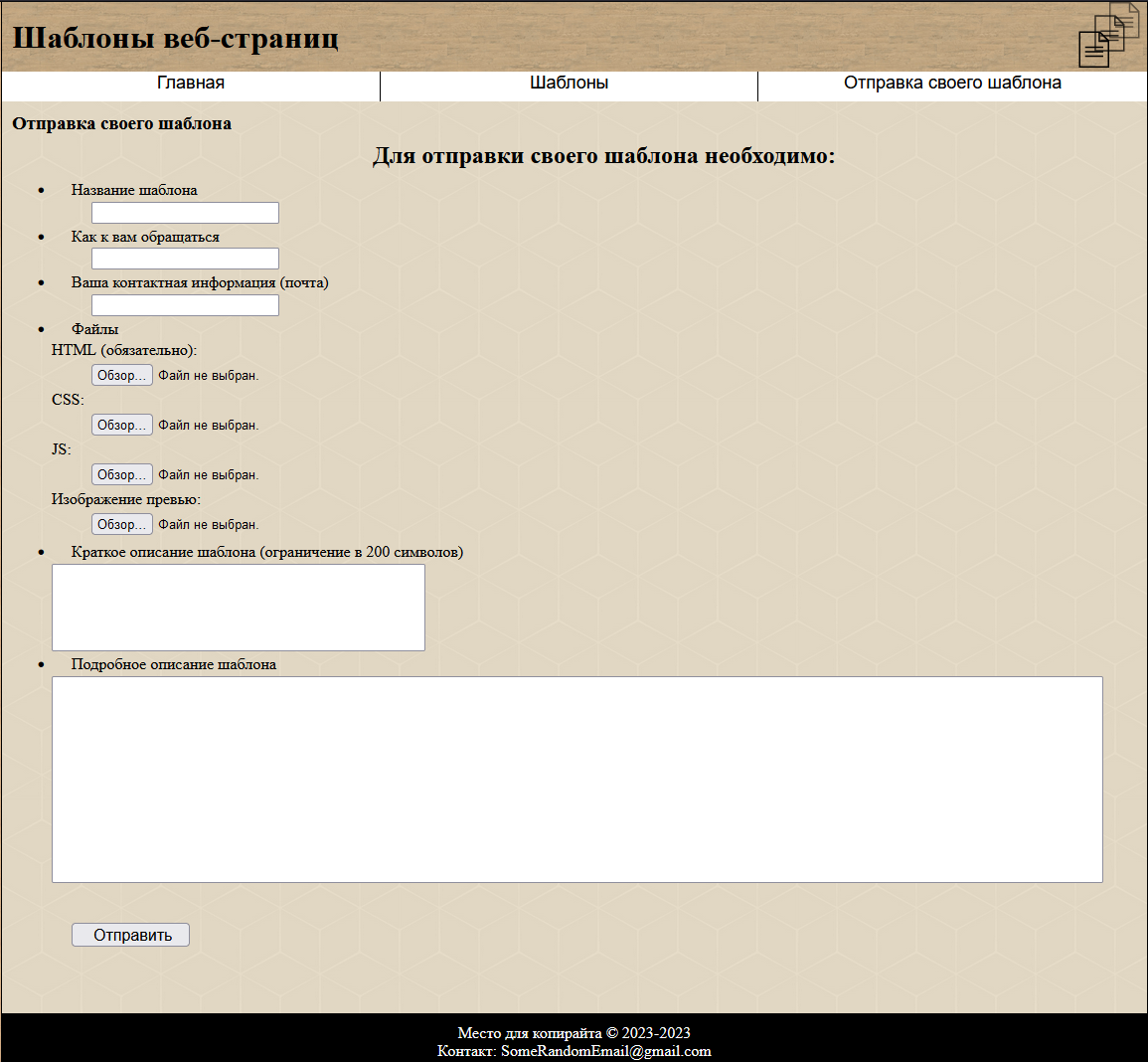


Рисунок 4.3 – Страница отправки шаблона в Mozilla Firefox

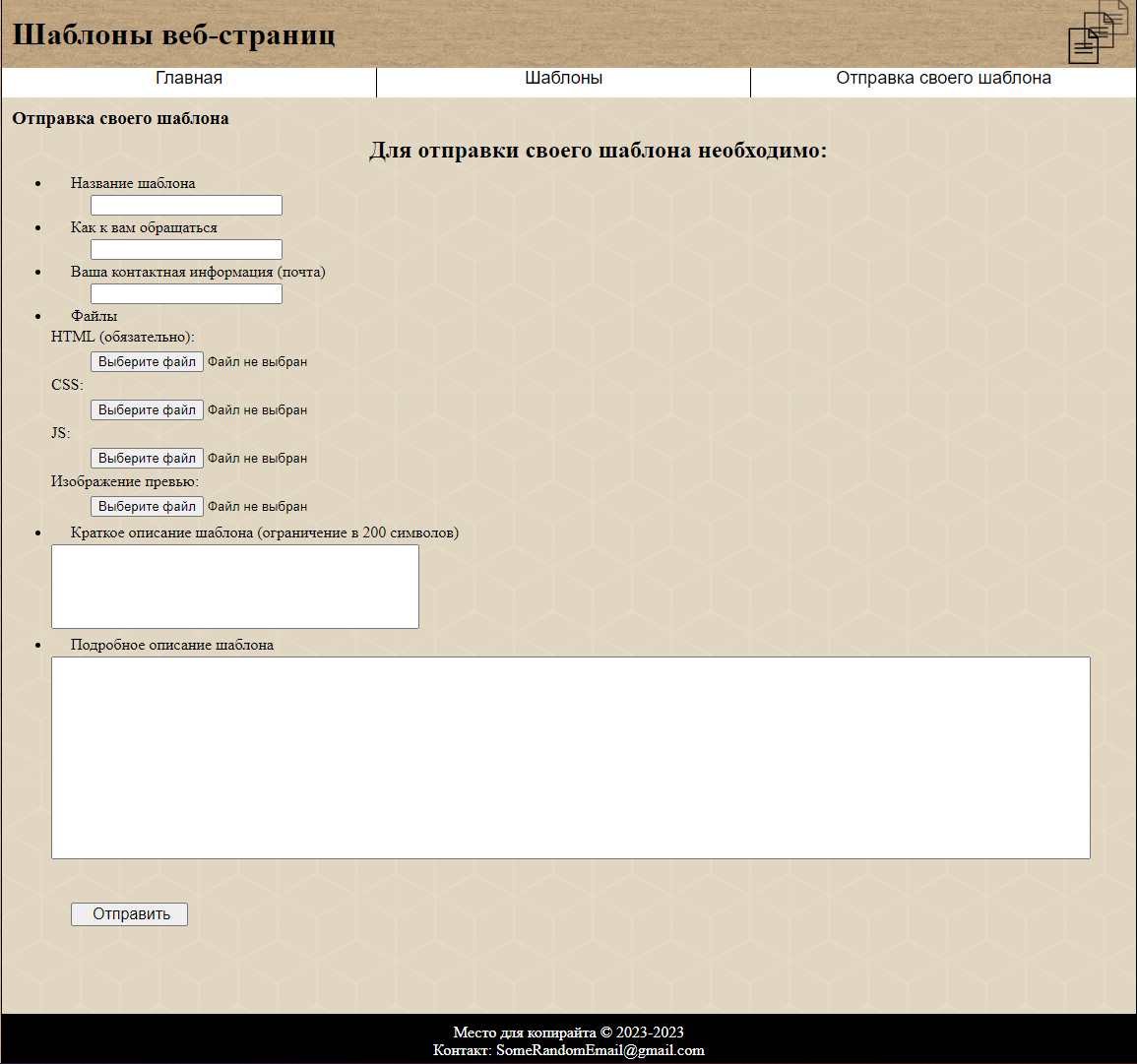
****

Рисунок 4.4 – страница отправки шаблона в Google Chrome

**4.3 Руководство пользователя**

Первой страницей является главная представленная на рисунке 4.5. На ней можно узнать базовую информацию о сайте, а также перейти на другие страницы при помощи панели навигации.

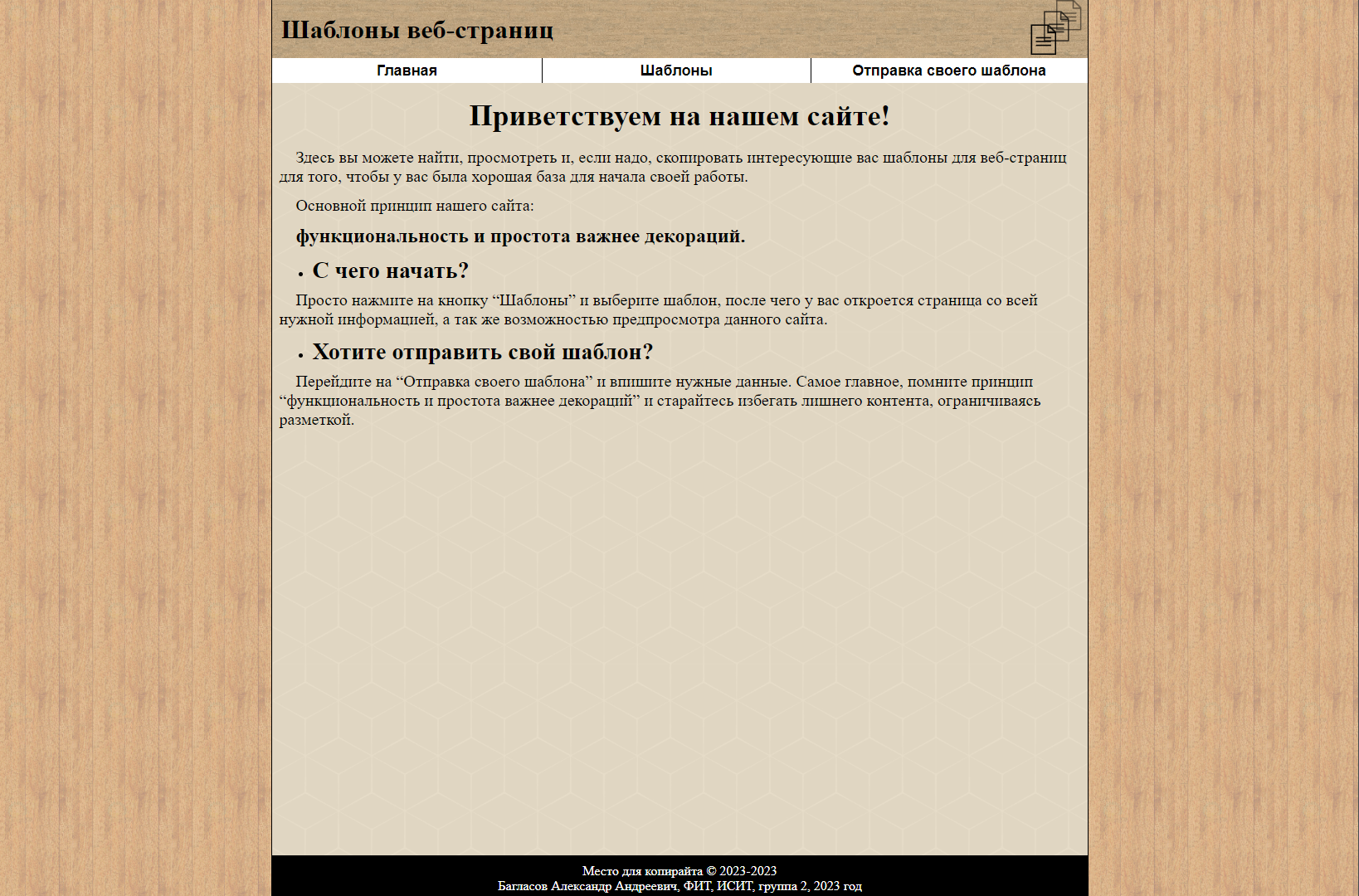


Рисунок 4.5 – Главная страница

На странице просмотра списка шаблонов, представленной на рисунке 4.6, пользователь может выбрать шаблон, что откроется на веб-странице просмотра шаблона, позволяя просмотреть более детальную информацию о нем, включая его описание и код.

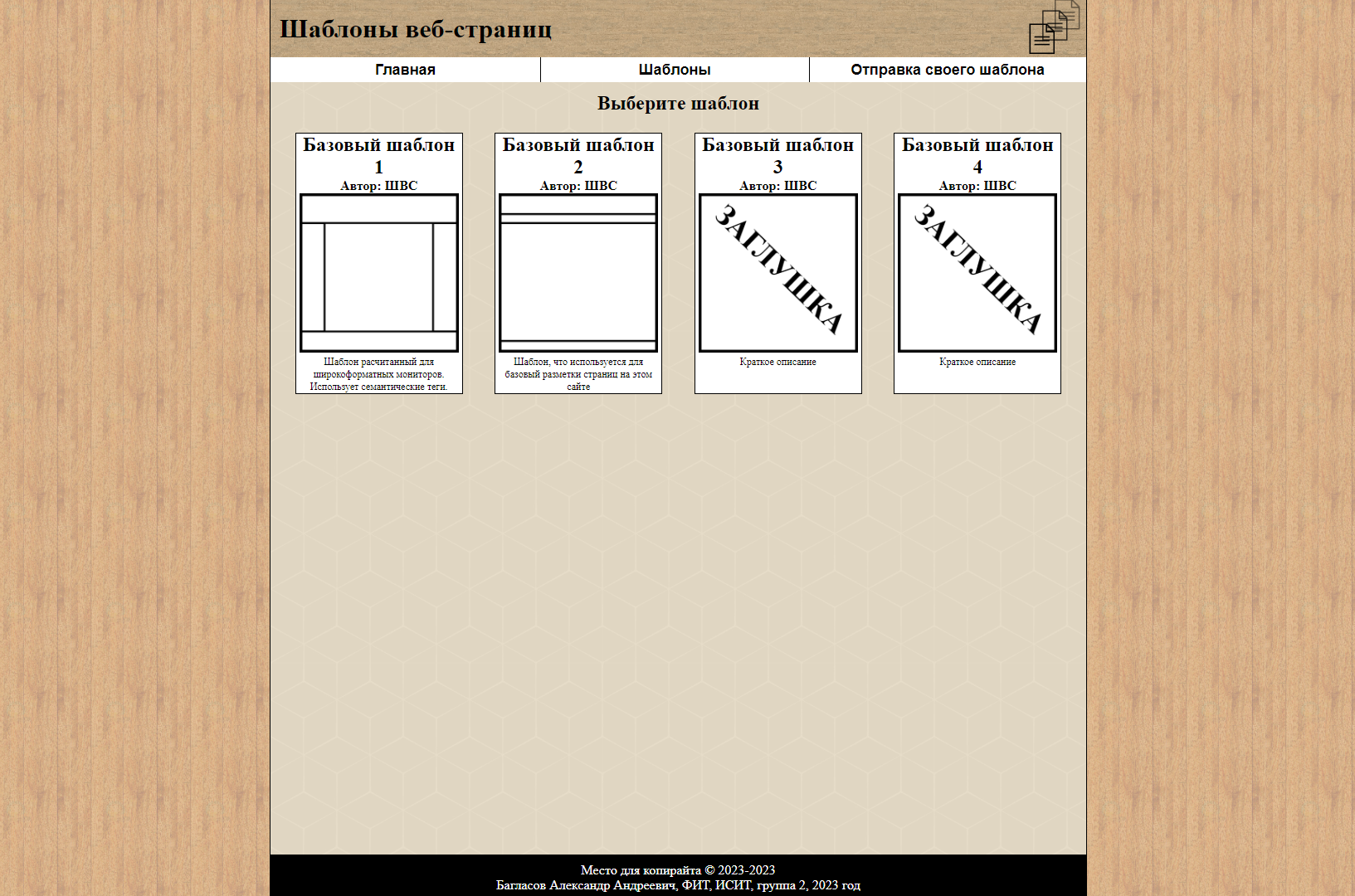


Рисунок 4.6 – Страница просмотра списка шаблонов

На странице просмотра шаблона, представленной на рисунке 4.7, пользователь может увидеть название шаблона, его автора, короткое описание, длинное описание, просмотреть код на HTML, CSS и JS, а также нажать на кнопку “Предпросмотр”, дабы открыть страницу с превью шаблона.

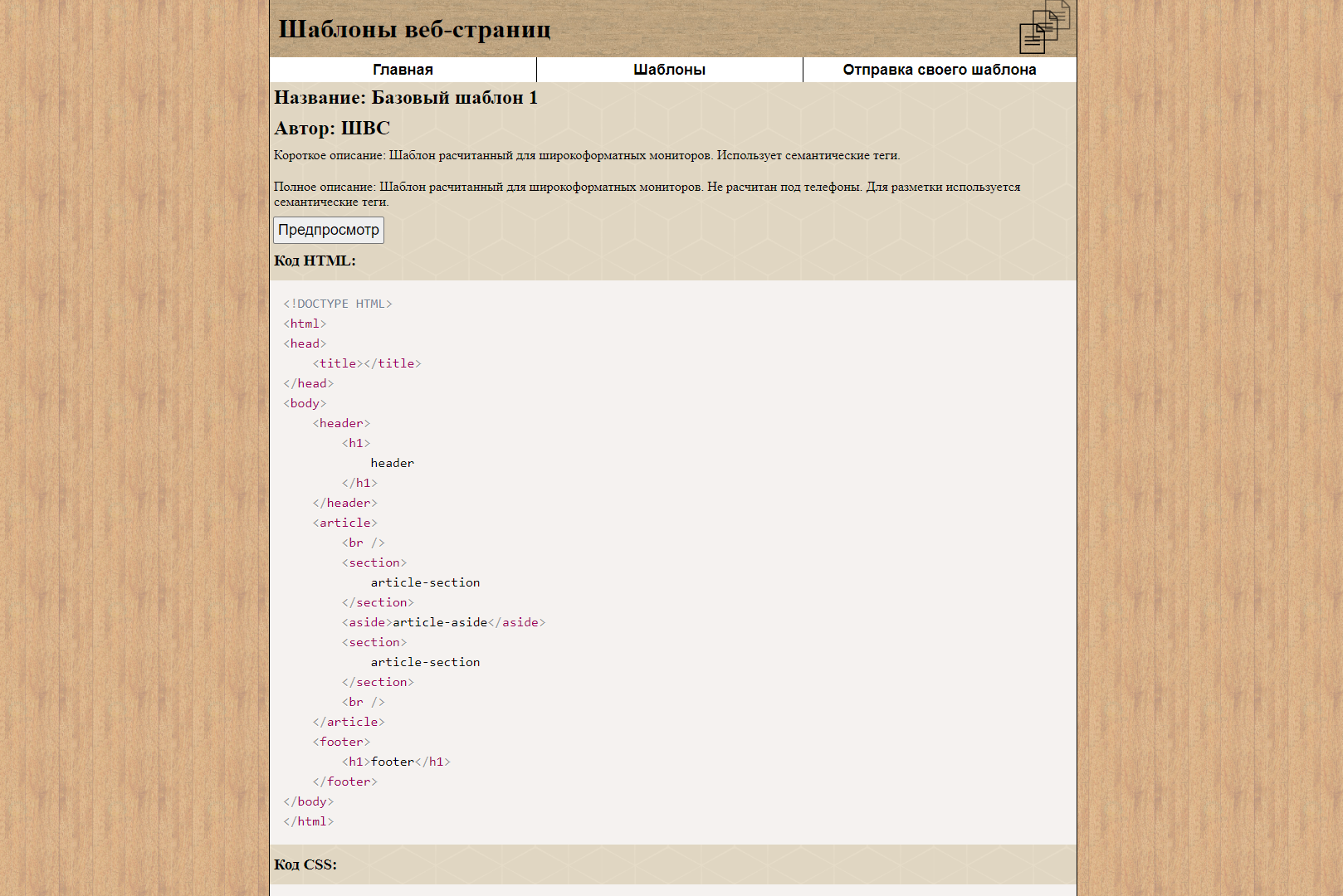


Рисунок 4.7 – Страница просмотра шаблона

На странице просмотра превью шаблона, представленной на рисунке 4.8, пользователь может просмотреть, что за шаблон он собирается выбрать, дабы дать пользователю полное понимание того, как выглядит шаблон.

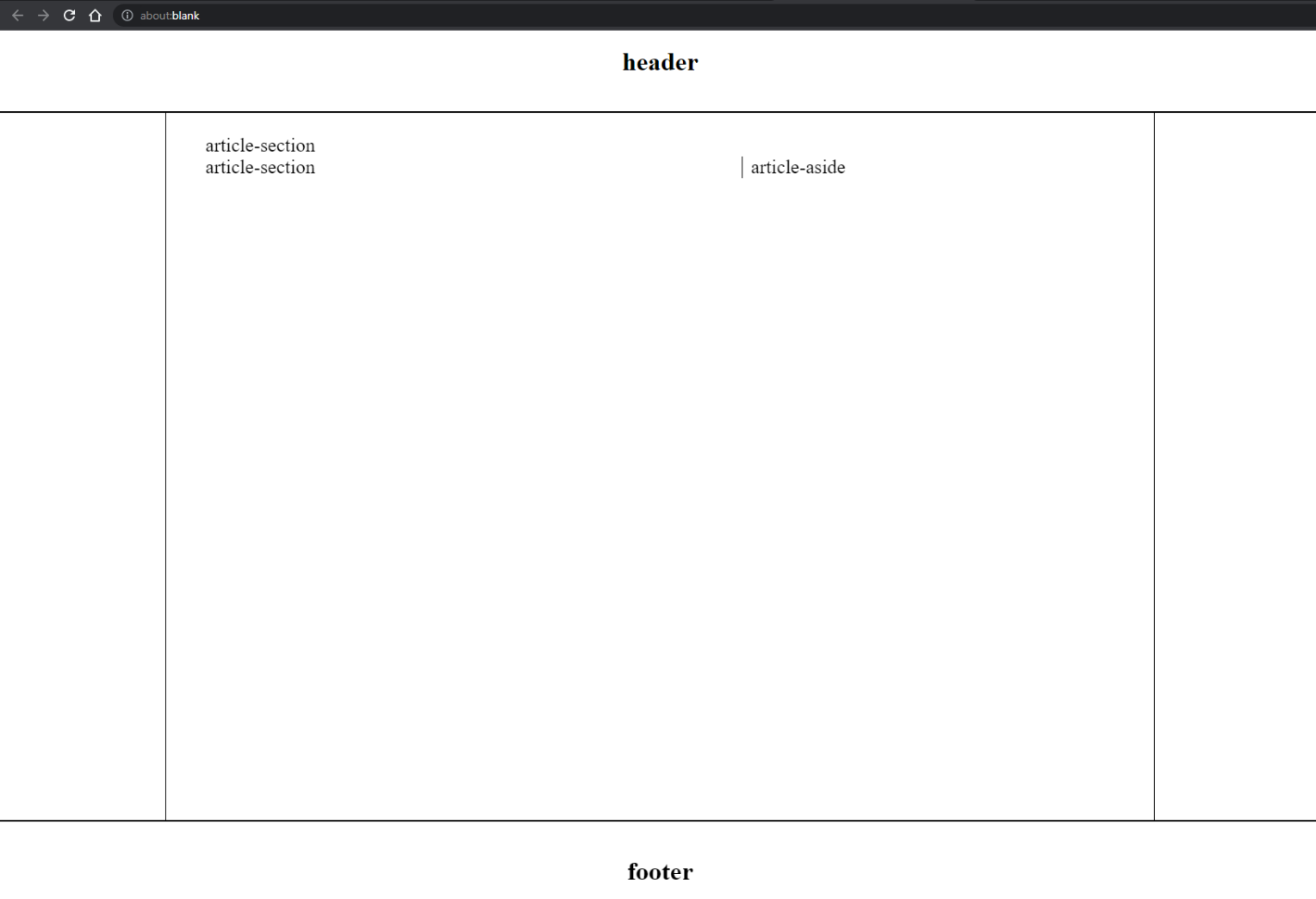


Рисунок 4.8 – Страница просмотра превью шаблона

Страница отправки шаблона, представлена на рисунке 4.9. На ней пользователь может указать различные данные, навроде названия шаблона, как к нему обращаться, почту, файлы HTML, CSS, JS и изображение превью, ввести короткое и полное описание, а затем отправить его для рассмотрения. В случае соблюдения всех требований, и скорее всего небольшой редакции, шаблон будет добавлен в базу данных и появится в списке шаблонов для всех пользователей.

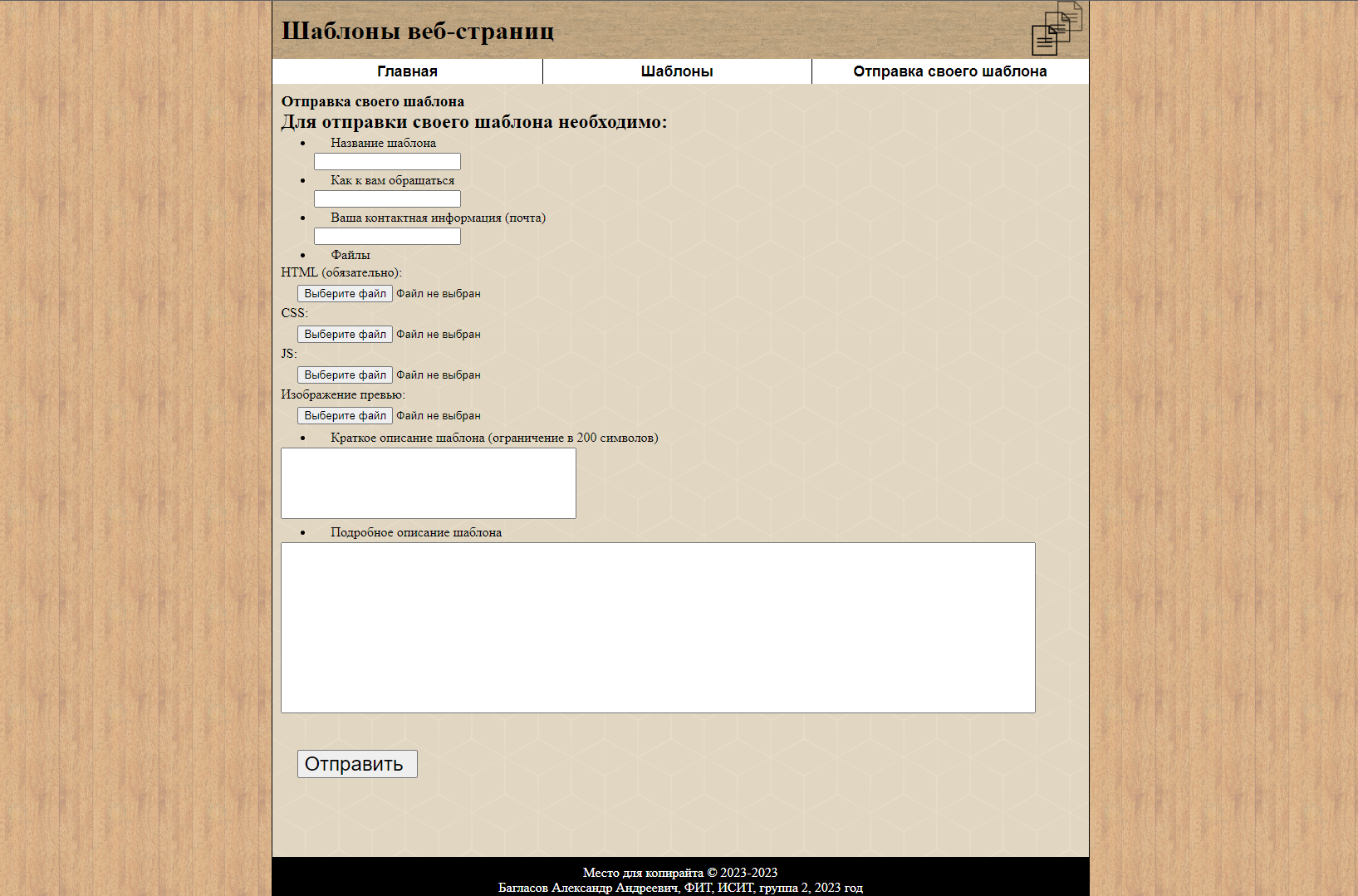


Рисунок 4.9 – Страница отправки шаблона

# **4.4 Валидность**

Валидность HTML-верстки – соответствие HTML кода со стандартами W3C (World Wide Web Consortium). Сайт прошёл проверку на валидность в сервисе involta.ru [6]. Ошибки не были обнаружены. Результат представлен на рисунке 4.10.



Рисунок 4.10 – Проверка на валидность

# **4.5 Выводы**

В данной главе было проведено тестирование и рассмотрена адаптивность кода, поддержка кроссбраузерности, проверена валидность кода и разработано руководство пользователя.

# **Заключение**

В ходе курсового проекта был разработан web-сайт «Шаблоны веб-страниц», созданный с целью упростить работу начинающим программистам и предоставить хорошую базу для начала работы. Был реализован функционал:

1. просмотр списка шаблонов;
2. просмотр отдельных шаблонов;
3. предпросмотр шаблонов в отдельном окне.

В процессе работы над курсовым проектом были использованы JavaScript, HTML, СSS, применялась библиотека PrismJS, предоставляющая подсветку синтаксиса HTML, CSS и JavaScript. Был использован Flexbox, упрощающий расположение элементов на странице и способствующий созданию адаптивной верстке.

Был проведен анализ веб-сайтов данной тематики для выявления достоинств и недостатков, поиска аналогов с приблизительно похожим функционалом и изучение актуальности проекта.

Работа с прототипом производилась в сервисе Figma. Работа над макетом осуществлялась в Photoshop.

По завершению работы над программным продуктом был создан кроссбраузерный адаптивный сайт для просмотра, поиска и использования базовых шаблонов веб-страниц. Так же были выделены следующие недостатки:

1. нечеткие изображения для ячеек с шаблонами;
2. отсутствие реализации функционала передачи данных из отправки своего шаблона на сервер или на почту;
3. нехватка контента на главной странице.

Сайт может быть дополнен и модернизирован. Исходники проекта расположены на сервисе GitHub: https://github.com/Devel-II/KursovoiProject.

**Список использованных источников**

* 1. Библиотека PrismJS [Электронный ресурс]. – Режим доступа: https://prismjs.com. – Дата доступа: 2023.
  2. Nicepage.com [Электронный ресурс]. – Режим доступа: https://nicepage.com/ru/html-shablony. – Дата доступа: 2023.
  3. freehtmlthemes.ru [Электронный ресурс]. – Режим доступа: https://freehtmlthemes.ru/. – Дата доступа: 2023.
  4. html-templates.info [Электронный ресурс]. – Режим доступа: https://html-templates.info/. – Дата доступа: 2023.
  5. psd-html-css.ru [Электронный ресурс]. – Режим доступа: http://psd-html-css.ru/. – Дата доступа: 2023.
  6. involta.ru [Электронный ресурс]. – Режим доступа: https://involta.ru/tools/validator-html/. – Дата доступа: 2023.

ПРИЛОЖЕНИЕ А

**Макеты страниц веб-сайта**

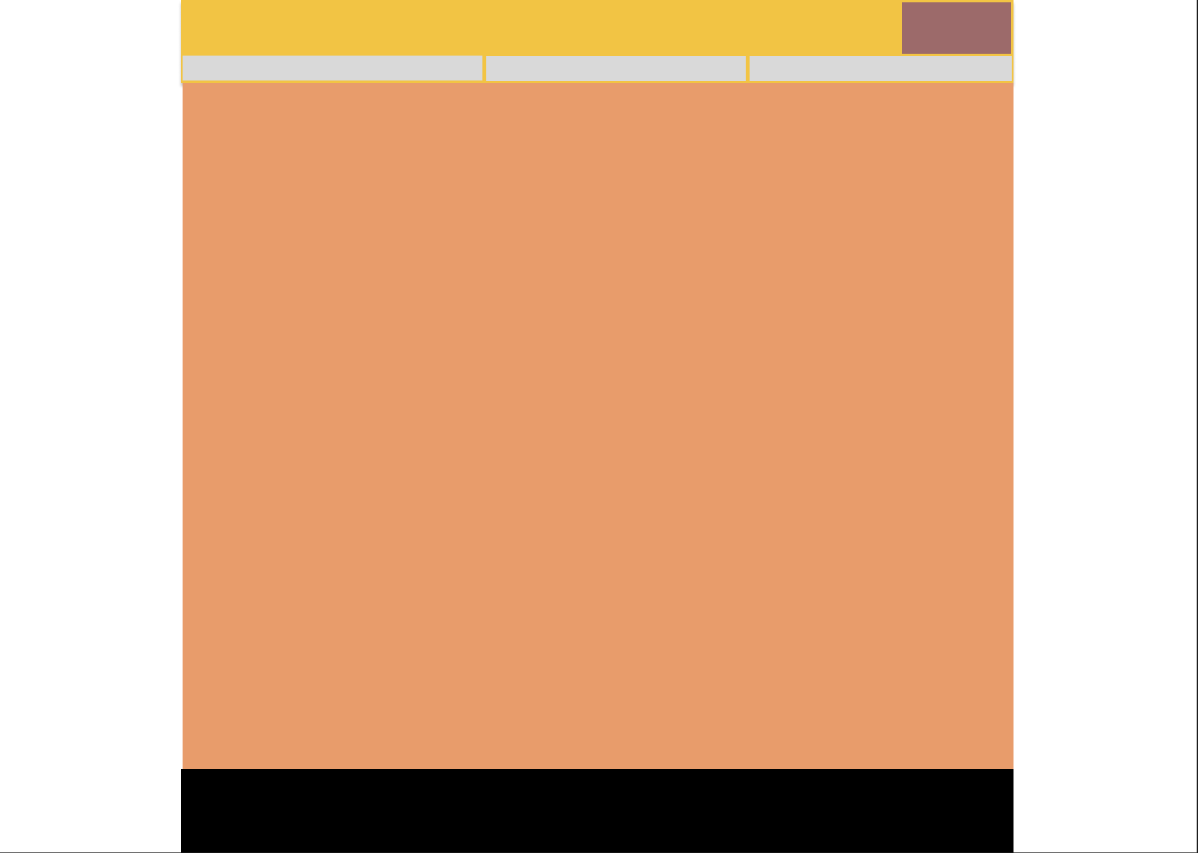


Рисунок А.1 — Прототип разметки страницы

# ПРИЛОЖЕНИЕ Б

1. **Макет структуры веб-сайта**

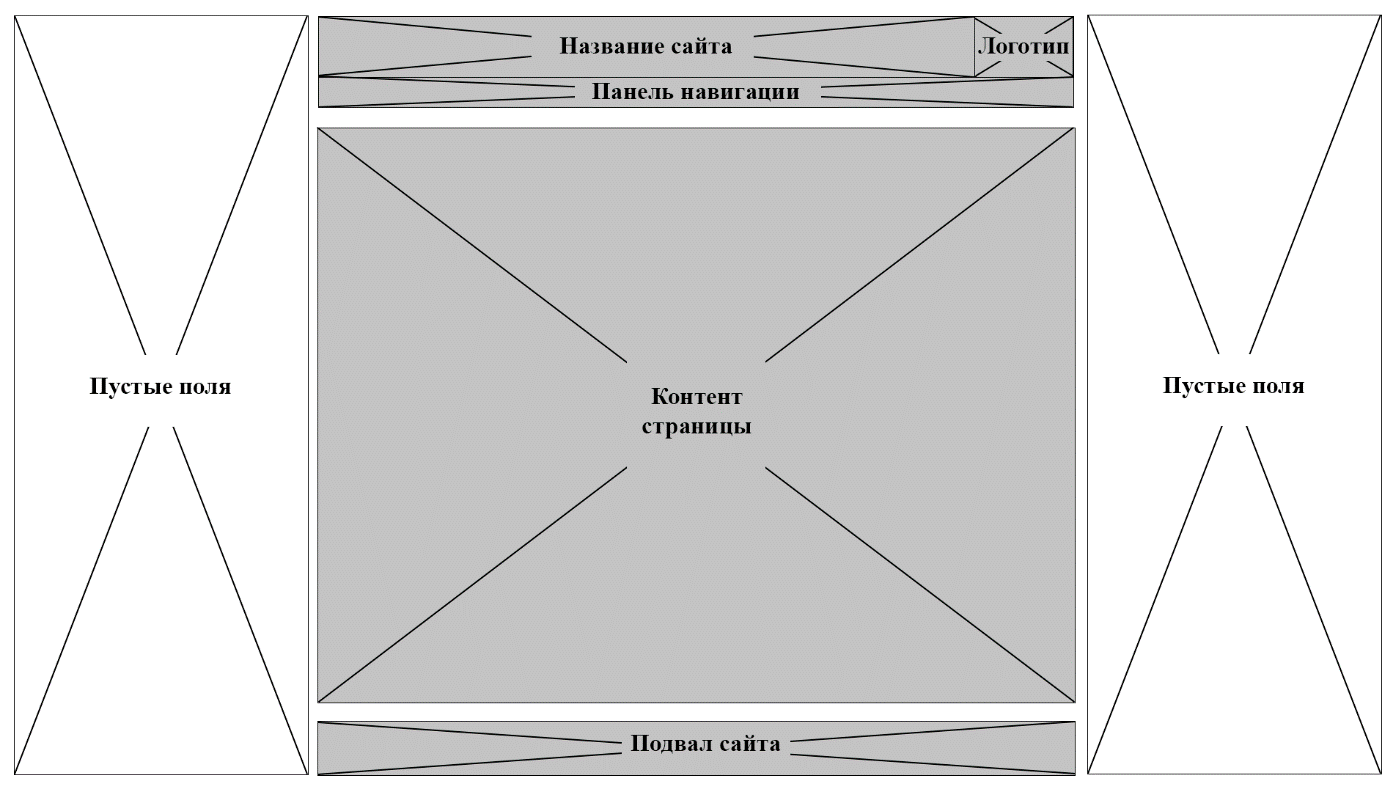


Рисунок Б.1 — Макет структуры сайта

# ПРИЛОЖЕНИЕ В

1. **Листинг кода для работы с xml**

|  |
| --- |
| let xmlContent = '';  var csscodeBuf; var htmlcodeBuf; var jscodeBuf;  fetch('xml.xml').then((response) => {  response.text().then((xml) => {  xmlContent = xml; let parser = new DOMParser();  let xmlDOM = parser.parseFromString(xmlContent, 'application/xml');  let element = xmlDOM.querySelectorAll('element');  let id = (window.location.search.replace(/^.\*?\=/, ''));  htmlcodeBuf = element[id].children[5].innerHTML.replace("<![CDATA[", "").replace("]]>", "");  csscodeBuf = element[id].children[6].innerHTML.replace("<![CDATA[", "").replace("]]>", "");  jscodeBuf = element[id].children[7].innerHTML.replace("<![CDATA[", "").replace("]]>", "");  document.getElementById('name').innerText = "Название: " + element[id].children[0].innerHTML;  document.getElementById('author').innerText = "Автор: " + element[id].children[1].innerHTML; document.getElementById('shortDescription').innerText = "Короткое описание: " + element[id].children[2].innerHTML;  document.getElementById('previewImage').src = "preview\_img/" + element[id].children[3].innerHTML; document.getElementById('longDescription').innerText = "Полное описание: " + element[id].children[4].innerHTML;  document.getElementById('htmlcode').innerHTML = window.Prism.highlight(htmlcodeBuf, window.Prism.languages.markup, 'markup');  document.getElementById('csscode').innerHTML = window.Prism.highlight(csscodeBuf, window.Prism.languages.css, 'css');  document.getElementById('jscode').innerHTML = window.Prism.highlight(jscodeBuf, window.Prism.languages.javascript, 'javascript');  });}); |

1. Листинг В.1 – Код для работы с xml

# ПРИЛОЖЕНИЕ Г

1. **Листинг XML-файлов**

|  |
| --- |
| 1. <?xml version="1.0" encoding="utf-8"?> 2. <main> 3. <element> 4. <name>Базовый шаблон 1</name> 5. <creator>ШВС</creator> 6. <shortDescription>Шаблон расчитанный для широкоформатных мониторов. Использует семантические теги.</shortDescription> 7. <imgPreviewName>Template 1.png</imgPreviewName> 8. <longDescription>Шаблон расчитанный для широкоформатных мониторов. Не расчитан под телефоны. Для разметки используется семантические теги.</longDescription> 9. <htmlcode><![CDATA[<!DOCTYPE HTML> 10. <html> 11. <head> 12. <title></title> 13. </head> 14. <body> 15. <header> 16. <h1> 17. header 18. </h1> 19. </header> 20. <article> 21. <br /> 22. <section> 23. article-section 24. </section> 25. <aside>article-aside</aside> 26. <section> 27. article-section 28. </section> 29. <br /> 30. </article> 31. <footer> 32. <h1>footer</h1> 33. </footer> 34. </body> 35. </html>]]></htmlcode> 36. <csscode><![CDATA[\* { 37. margin: 0px; 38. padding: 0px; 39. } 40. header { 41. border: solid; 42. border-width: 0; 43. border-bottom-width: 2px; 44. border-color: black; 45. position: sticky; 46. height: 80px; 47. padding-top: 20px; 48. width: 100%; 49. text-align: center; 50. } 51. article { 52. font-size: 24px; 53. padding-left: 3%; 54. padding-right: 3%; 55. height: calc(100% - 202px); 56. position: fixed; 57. right: 12.5%; 58. left: 12.5%; 59. border-left: 1px solid black; 60. border-right: 1px solid black; 61. } 62. footer { 63. border: solid; 64. border-width: 0; 65. border-top-width: 2px; 66. height: 80px; 67. width: 100%; 68. text-align: center; 69. position: fixed; 70. bottom: 0px; 71. display: grid; 72. align-items: end; 73. padding-bottom: 20px; 74. } 75. aside { 76. width: 40%; 77. margin-left: 10px; 78. float: right; 79. display: inline; 80. padding-left: 10px; 81. box-shadow: -1px 0px 0px black; 82. }]]></csscode> 83. <jscode><![CDATA[console.log("test");]]></jscode> 84. </element> 85. <element> 86. <name>Базовый шаблон 2</name> 87. <creator>ШВС</creator> 88. <shortDescription>Шаблон, что используется для базовый разметки страниц на этом сайте</shortDescription> 89. <imgPreviewName>Template 2.png</imgPreviewName> 90. <longDescription>Данный шаблон используется для разметки этого сайта. Можете использовать его по своему желанию. Нужды в дополнительных материалах нету.</longDescription> 91. <htmlcode><![CDATA[<!DOCTYPE html> 92. <html> 93. <head> 94. <meta charset="utf-8" /> 95. <title>Главная</title> 96. </head> 97. <body> 98. <div class="main"> 99. <div class="top"> 100. <div class="top\_1"> 101. Верхняя часть верха 102. </div> 103. <div class="top\_2"> 104. Нижняя часть верха 105. </div> 106. </div> 107. <div class="center"> 108. Контент 109. </div> 110. <div class="bottom"> 111. Низ 112. </div> 113. </div> 114. </body> 115. </html>]]></htmlcode> 116. <csscode> 117. <![CDATA[/\*Базовая верстка:\*/ 118. \* { 119. margin: 0px; 120. } 121. html, body { 122. height: 100%; 123. position: relative; 124. } 125. /\*Для лучшего предосмотра:\*/ 126. .main > \* { 127. border: 1px solid black; /\*Для обозначения границ для лучшего просмотра\*/ 128. } 129. /\*Основа\*/ 130. @media screen and (min-width: 1200px) { 131. .main { 132. border-left: 1px solid black; 133. border-right: 1px solid black; 134. width: 60%; 135. left: 20%; 136. } 137. } 138. @media screen and (max-width: 1200px) { 139. .main { 140. width: calc(100% - 2px); 141. } 142. } 143. .main { 144. min-height: 100%; 145. position: relative; 146. } 147. /\*Верх\*/ 148. .top { 149. height: 100px; 150. width: 100%; 151. } 152. .top\_1 { 153. display: flex; 154. align-items: center; 155. padding-left: 10px; 156. height: 70px; 157. } 158. .top\_2 { 159. display: flex; 160. flex-wrap: wrap; 161. width: 100%; 162. } 163. .top\_2 > \* { 164. border: 0px; 165. height: 100%; 166. /\*Место для указания размеров элементов для нижней части топа (например: кнопок)\*/ 167. padding: 0px; 168. } 169. .center { 170. width: 100%; 171. height: auto; 172. padding-bottom: 50px; 173. } 174. /\*Низ\*/ 175. .bottom { 176. position: absolute; 177. bottom: 0px; 178. height: 40px; 179. width: 100%; 180. text-align: center; 181. }]]></csscode> 182. <jscode></jscode> 183. </element> 184. <element> 185. <name>Базовый шаблон 3</name> 186. <creator>ШВС</creator> 187. <shortDescription>Краткое описание</shortDescription> 188. <imgPreviewName>Template TEST.png</imgPreviewName> 189. <longDescription>Длинное описание</longDescription> 190. <htmlcode> 191. <![CDATA[ 192. <html> 194. </html> 195. ]]> 196. </htmlcode> 197. <csscode> 198. <![CDATA[ 199. div { width: 20px; } 200. ]]> 201. </csscode> 202. <jscode> 203. <![CDATA[ 204. console.log("test"); 205. ]]> 206. </jscode> 207. </element> 208. <element> 209. <name>Базовый шаблон 4</name> 210. <creator>ШВС</creator> 211. <shortDescription>Краткое описание</shortDescription> 212. <imgPreviewName>Template TEST.png</imgPreviewName> 213. <longDescription>Длинное описание</longDescription> 214. <htmlcode> 215. <![CDATA[ 216. <html> 218. </html> 219. ]]> 220. </htmlcode> 221. <csscode> 222. <![CDATA[ 223. div { width: 20px; } 224. ]]> 225. </csscode> 226. <jscode> 227. <![CDATA[ 228. console.log("test"); 229. ]]> 230. </jscode> 231. </element> 232. </main> |

1. Листинг 2 – файл xml.xml