Министерство науки и высшего образования Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ  
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«Санкт-Петербургский государственный университет  
промышленных технологий и дизайна»**

**ВЫСШАЯ ШКОЛА ПЕЧАТИ И МЕДИАТЕХНОЛОГИЙ**

**Институт:** Полиграфических технологий и оборудования

**Кафедра:** Информационных и управляющих систем

**Направление подготовки:** 09.03.02 Информационные системы и технологии

**Профиль подготовки:** Информационные технологии **в медиаиндустрии**

**ОТЧЕТ ПО КУРСОВОЙ РАБОТЕ**

**Дисциплина: Веб-программирование**

**Тема: Разработка и создание сайта для демонстрации особенностей культурных ценностей малых народов Севера**

**Выполнил:**

студент группы Иванова И.Г.

*(подпись)*

**Руководитель:**

к.т.н., доцент Пасечник П.А

*(уч. степень, звание) (подпись)*

**Дата защиты работы**

**Оценка**

**Санкт-Петербург**

**2021**

РЕФЕРАТ

Отчет 45 с., 3 кн., 55 рис., 2 табл., 16 источн., 1 прил.

ИНТЕРАКТИВНОСТЬ, ИНФОРМАЦИОННЫЙ ИСТОЧНИК, КУЛЬТУРА, ПРОЕКТИРОВАНИЕ, РАЗРАБОТКА, САЙТ, СПРАВОЧНИК, ЭТНОС

**Объект исследования:** Использование интерактивно-визуальных методов повествования для демонстрации особенностей культурных ценностей малых народов Севера.

**Цель курсовой работы:** разработать и создать сайт.

В процессе работы была изучена и проанализирована предметная область - велось посредством анализа и синтеза источников, также проводилось сравнение и обобщение схожих или смежных по тематике разработок, а сама разработка велась, опираясь на принципы схожие с принципами итеративно-инкрементной модели, с использованием методов объектно-ориентированного и функционального программирования.

По ходу курсовой были также рассмотрены различные технические среды и средства разработки, с помощью программ прикладного назначения был написан сам сайт.

Как результат, была реализована разработка сайта, демонстрирующего культурные особенности малых народов Севера, представляющий собой интерактивный справочник.

Данный проект может использоваться как справочный материал для детей школьного и дошкольного возраста, а также для заинтересованных в культуре малых народов.

В ходе тестирования было выявлено, что следует улучшить пользовательский интерфейс. Можно расширить информационную базу, и основательно проработать и внедрить серверную часть.

Разработку можно использовать как основу для прочих схожих или смежных проектов, но уже на отличные темы.

ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ

В настоящем отчете о курсовой работе применяют следующие термины с соответствующими определениями – они приведены в таблице 1.

Таблица 1 – Термины и определения

|  |  |
| --- | --- |
| Термин | Определение |
| Атрибут | Это спецификация, которая определяет свойство объекта, элемента или файла. Он также может ссылаться или устанавливать конкретное значение для данного экземпляра такового.[6] |
| Идентификатор | Уникальный признак объекта, позволяющий отличать его от других объектов, то есть идентифицировать[6]. |
| Интерфейс | Совокупность средств, при помощи которых пользователь взаимодействует с различными программами и устройствами[6]. |
| Метаданные, мета | Информация о другой информации, или данные, относящиеся к дополнительной информации о содержимом или объекте[6]. |
| Программа | Описание на языке программирования или в машинном коде действий, которые должна выполнить ЭВМ в соответствии с алгоритмом решения конкретной задачи или группы задач (синоним - машинная программа);  Упорядоченная последовательность команд, подлежащих обработке[2, раздел 4.1.1]. |
| Репозиторий | (от англ. repository — хранилище) — место, где хранятся и поддерживаются какие-либо данные. Чаще всего данные в репозитории хранятся в виде файлов, доступных для дальнейшего распространения по сети.[6] |
| Сервер | Это компьютер с высокой вычислительной мощностью и круглосуточной бесперебойной работой, предназначенный для хранения информации и обеспечения доступа к ней с любого устройства пользователя.[6] |
| Среда разработки | Комплекс программных средств, используемый программистами для разработки программного обеспечения (ПО)[6]. |
| Фреймворк | Программная платформа, определяющая структуру программной системы; программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта.[6] |
| Функция | Обособленный участок кода, который можно вызывать, обратившись к нему по имени, которым он был назван, в конце может быть возврат того или иного значения.[6] |
| Хостинг | Услуга по предоставлению ресурсов для размещения информации на сервере, постоянно имеющем доступ к сети.[17] |

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБРАЩЕНИЙ

В настоящем отчете о курсовой применяют следующие сокращения и обозначения – они приведены в таблице 2.

Таблица 2 – Сокращения, обращения и их расшифровка

|  |  |  |
| --- | --- | --- |
| Аббревиатура/  Сокращение | Расшифровка | Определение |
| ООП | Объектно-ориентированное программирование | Способ организации программы, позволяющий использовать один и тот же код многократно. Позволяет описать предметы реального мира в виде удобных сущностей – объектов, а также организовать связи между этими объектами[2, с. 244]. |
| ОС | Операционная система | Программа или совокупность программ, управляющая основными действиями ЭВМ, ее периферийными устройствами и обеспечивающая запуск всех остальных программ, а также взаимодействие с оператором[2, раздел 4.2.1]. |
| ПО | Программное обеспечение | Программное обеспечение представляет собой либо данные для использования в других программах, либо алгоритм, реализованный в виде последовательности инструкций для процессора[6]. |
| ПК | Персональный компьютер | Персональный компьютер |

СОДЕРЖАНИЕ

[РЕФЕРАТ 2](#_Toc108013292)

[ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ 3](#_Toc108013293)

[ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБРАЩЕНИЙ 4](#_Toc108013294)

[СОДЕРЖАНИЕ 5](#_Toc108013295)

[ВВЕДЕНИЕ 6](#_Toc108013296)

[1 Проектирование 8](#_Toc108013297)

[1.1 Обзор технологий 8](#_Toc108013298)

[1.2 Сфера применения 13](#_Toc108013299)

[1.3 Описание структуры разрабатываемого сайта 14](#_Toc108013300)

[1.4 Концепция и дизайн макетов интерфейса сайта 16](#_Toc108013301)

[2 Разработка сайта 18](#_Toc108013302)

[2.1 Верстка интерфейса и задание стилей для элементов 18](#_Toc108013303)

[2.2 Программирование работы функционала 22](#_Toc108013304)

[2.2.2 Описание функций. 22](#_Toc108013305)

[2.2.3 Описание интерактивной составляющей 24](#_Toc108013306)

[2.3 Серверная сторона. Динамические данные 28](#_Toc108013307)

[2.3.1 Репозиторий. Сервер. Хранение данных на веб-сервере 29](#_Toc108013308)

[2.3.2 Описание работы серверной части. Формат данных. Запросы 30](#_Toc108013309)

[3 Тест и отладка 33](#_Toc108013310)

[3.1 Публикация сайта на GitHub Pages 33](#_Toc108013311)

[3.2 GitHub Desktop. Фиксация и отслеживание изменений. 35](#_Toc108013312)

[3.3 Тест опубликованного сайта. 37](#_Toc108013313)

[ЗАКЛЮЧЕНИЕ 42](#_Toc108013314)

[СПИСОК ЛИТЕРАТУРЫ 43](#_Toc108013315)

[ПРИЛОЖЕНИЕ А 45](#_Toc108013316)

ВВЕДЕНИЕ

В полиэтническом пространстве России среди 150 народов, живущих на ее территории, малочисленные народы ассимилируются, численность многих народов уменьшается, численность же других возрастает. Численность народов Севера и Дальнего Востока постоянно сокращается.

С уменьшением населения уходят из быта этнокультурные традиции, произведения фольклора, язык. Проблемы знания и понимания этнокультуры стоят перед каждым народом. [3]

Активизация национальных культурных процессов порождает среди малых и средних этносов стремление к возвращению к этническим культурным ценностям. Оно реализуется путем расширения объема функционирования родного языка и культуры в учебных процессах, представляющих собой занятия имеющие этнокультурную направленность. [7]

Здесь большую роль играет содержание подобного этнокультурного образования - это важно, поскольку оно ориентирует воспитание и вектор развития личности ребенка.

Именно образованный человек причастен к исторической и культурной традиции. Он ощущает принадлежность к определенной общности и народу, у него сформированы культурные потребности и личностные, духовные ценности, а также есть понимание идеи принятия иной культуры.[7]

**Объект:** Использование интерактивно-визуальных методов повествования для демонстрации особенностей культурных ценностей малых народов Севера.

**Предмет:** Методы, средства и подходы при разработке и создании сайтов.

**Актуальность:** Ассимиляция малых народов России вызывает потребность в популяризации этнокультурного и народного образования, следовательно, и поиск новых способов и методов эту потребность удовлетворить.

**Теоретическая значимость:** Данный проектможет быть использован в качестве справочного, учебного, развивающего материала для детей школьного и дошкольного возраста, а также может быть полезным для заинтересованных в культуре малых народов.

**Практическая значимость:** Разработку можно использовать как основу для прочих схожих или смежных проектов, но уже на отличные темы.

**Целью** курсовой работы является разработка и создание интерактивного сайта для демонстрации особенности культурных ценностей малых народов Севера.

Я выбрала тему, связанную с народами Севера, поскольку сама являюсь представителем малых народов, и имею личный интерес в ознакомлении других с культурой народов Севера.

**Задачи:**

* Проектирование сайта или веб-приложения (сбор и анализ требований, разработка технического задания, проектирование интерфейсов),
* Разработка и создание креативной и дизайн концепций сайта,
* Вёрстка,
* Программирование (разработка функциональных инструментов),
* Оптимизация и размещение материалов сайта,
* Публикация проекта,
* Тестирование и отладка.

1. Проектирование
   1. Обзор технологий

Для разработки сайта будут использоваться следующие средства:

* язык разметки HTML,
* каскадные таблицы стилей CSS,
* фреймворк Bootstrap,
* язык программирования Javascript,
* редактор для написания кода VScode и Notepad++,
* редакторы для работы с векторной и растровой графикой Inkscape, Adobe Photoshop,
* Figma,
* веб-сервис для хостинга GitHub,
* система контроля версий Git, GitHub Desktop,
* технология для публикации сайтов GitHub Pages,
* технология асинхронных запросов (Ajax).

Далее о них более подробно.

*HTML.* HTML (Hyper Text Markup Language) – это язык разметки гипертекстовых документов, - документов, содержащих ссылки на другие документы. Он нужен, чтобы отображать в браузере специальным образом отформатированный документ с множеством вложенных элементов: заголовками, абзацами, списками, гиперссылками, источниками медиа, расположением изображений, видео и аудио.[18]

HTML-документ — это текстовый файл с расширением .html или .htm. В браузере он преобразуется в веб-страницу и состоит из набора тегов. Они как раз и помогают представлять текст на экране: благодаря им браузер понимает, что он читает не просто текст, а структурированную информацию, разбитую на блоки.

*CSS.* CSS (Cascading Style Sheets) — это каскадные таблицы стилей. Это язык, который отвечает за описание внешнего вида HTML-документа. Подавляющее большинство современных веб-сайтов работают на основе связки HTML+CSS.[5,16]

CSS используется для определения стилей (правил) оформления документов — включая дизайн, вёрстку и вариации макета для различных устройств и размеров экрана.

Класс. У языка CSS относительно простой синтаксис. Сначала прописывается селектор — он выбирает конкретный элемент на странице. Потом, после фигурных скобок, указываются свойства со значениями — между ними ставится двоеточие. Сами свойства отделяются друг от друга точкой с запятой.

Селектор - это конструкция, которая позволяет выбрать отдельные или однотипные элементы на странице, чтобы их стилизовать, - с него начинается каждый блок объявлений в CSS.[6]

*Bootstrap.* Bootstrap — это открытый и бесплатный HTML, CSS и JS фреймворк, который используется веб-разработчиками для быстрой вёрстки адаптивных дизайнов сайтов и веб-приложений.[10]

Фреймворк Bootstrap – это набор CSS и JavaScript файлов. Чтобы использовать эти файлы необходимо просто подключить к странице. После этого появляется доступ к инструментам данного фреймворка: колоночная система (сетка Bootstrap), классы и компоненты.[14]

Bootstrap состоит из:

* инструментов для создания макета (обёрточных контейнеров, системы сеток, гибких медиа-объектов, адаптивных утилитных классов);
* классов для стилизации базового контента: текста, изображений, кода, таблиц и фигур;
* готовых компонентов: кнопок, форм, горизонтальных и вертикальных навигационных панелей, слайдеров, выпадающих списков, аккордеонов, модальных окон, всплывающих подсказок и др.;
* утилитных классов для решения традиционных задач наиболее часто возникающими перед веб-разработчиками: выравнивание текста, отображение и скрытие элементов, задания цвета, фона, margin и padding отступов, и т.д.

*JavaScript.* JavaScript - это мультипарадигменный язык программирования. Он поддерживает объектно-ориентированный, императивный и функциональный стили программирования. Является реализацией спецификации ECMAScript.[9]

JavaScript обычно используется как встраиваемый язык для программного доступа к объектам приложений. Наиболее широкое применение находит в браузерах как язык сценариев для придания интерактивности веб-страницам, т.е. с помощью этого языка реализуется возможность реакции страницы или отдельных ее элементов на действия посетителя.

*AJAX*. AJAX – это аббревиатура от «Asynchronous JavaScript and XML», которая дословно переводится как «асинхронный JavaScript и XML».[5]

Что означают эти слова?

* асинхронный – действие выполняется в фоне (не в основном потоке), другими словами, таким образом, что оно не мешает пользователю взаимодействовать со страницей;
* JavaScript – язык, на котором всё это делается (т.е. создание и настройка запроса, отправка его на сервер, получение ответа и его разбор, обновление страницы);
* XML – формат для хранения и передачи данных, в настоящее время вместо него чаще используется JSON, но кроме них можно использовать и другие форматы.

Что такое AJAX? AJAX – это термин, который описывает как можно используя существующие технологии получать данные с сервера в фоновом режиме и использовать их для обновления страницы (без перезагрузки). Основная цель AJAX – это сделать сайты и веб-приложения более удобными, быстрыми и отзывчивыми.

Основные преимущества использования AJAX:

* снижение трафика (из-за уменьшения объёма передаваемых данных между клиентом и сервером);
* уменьшение нагрузки на сервер (не нужно генерировать всю страницу, а только ту часть, которую нужно обновить);
* увеличение быстродействия и отзывчивости (нет необходимости в полные перезагрузки страницы, достаточно обновить содержимое только отдельных блоков);
* повышение интерактивности (с помощью AJAX можно сразу отображать результаты и сделать ресурс более удобным для пользования).

Взаимодействие с сервером через асинхронные запросы осуществляется посредством XHR или метода fetch().

*GitHub, Система контроля версий Git, GitHub Pages*

GitHub — веб-сервис для хостинга IT-проектов и их совместной разработки. Веб-сервис основан на системе контроля версий Git.[6,19]

Система контроля версий — это система, записывающая изменения в файл или набор файлов в течение времени и позволяющая вернуться позже к определённой версии.

Git — система управления версиями с распределенной архитектурой. В отличие от некогда популярных систем вроде CVS (Concurrent Versions System, Система одновременных версий) и Subversion (SVN), где полная история версий проекта доступна лишь в одном месте, в Git каждая рабочая копия кода сама по себе является репозиторием. Это позволяет всем разработчикам хранить историю изменений в полном объеме.

Распределенные системы контроля версий (Distributed Version Control System, DVCS) позволяют хранить репозиторий (его копию) у каждого разработчика, работающего с данной системой. При этом можно выделить центральный репозиторий (условно), в который будут отправляться изменения из локальных и, с ним же эти локальные репозитории будут синхронизироваться. При работе с такой системой, пользователи периодически синхронизируют свои локальные репозитории с центральным и работают непосредственно со своей локальной копией.

Разработка в Git ориентирована на обеспечение высокой производительности, безопасности и гибкости распределенной системы.

GitHub Pages — это бесплатный хостинг для статических файлов.

Веб-сервер пересылает запросы на генерацию данных от клиента и полученный ответ отправляет обратно клиенту. Это динамические данные, от типа запроса тут ничего не зависит. Статика - это файлы, которые лежат на сервере и отдаются пользователю в том виде, в каком есть (HTML, CSS, JS, изображения, другие файлы).

GItHub Pages — это бесплатный хостинг с открытой структурой. Это значит, что загрузить туда свои файлы может кто угодно. И любой пользователь может посмотреть все загруженные файлы проекта.

*Visual Studio (VS code).* Visual Studio (VS code) — редактор исходного кода, разработанный Microsoft для Windows, Linux и macOS. Позиционируется как «лёгкий» редактор кода для кроссплатформенной разработки веб- и облачных приложений. Включает в себя отладчик, инструменты для работы с Git, подсветку синтаксиса, IntelliSense и средства для рефакторинга. Имеет широкие возможности для кастомизации: пользовательские темы, сочетания клавиш и файлы конфигурации.[6]

*SVG.* SVG — это язык описания двухмерной графики. В качестве отдельного формата или в сочетании с другим XML он использует синтаксис XML. Код SVG, используемый внутри документов HTML, использует синтаксис HTML.[6]

SVG допускает три типа графических объектов: векторные графические фигуры (например, контуры, состоящие из прямых линий и кривых), изображения и текст. Графические объекты можно группировать, стилизовать, преобразовывать и комбинировать. Набор функций включает вложенные преобразования, обтравочные контуры, альфа-маски, эффекты фильтров и объекты шаблонов.

Рисунки SVG могут быть интерактивными и динамичными. Анимации можно определять и запускать либо путем встраивания элементов анимации SVG в содержимое SVG, либо с помощью сценариев.

Сложные приложения SVG возможны за счет использования дополнительного языка сценариев, который обращается к объектной модели документа SVG (DOM), что обеспечивает полный доступ ко всем элементам, атрибутам и свойствам. Любому графическому объекту SVG можно назначить богатый набор обработчиков событий. На веб-странице одни и те же сценарии могут работать как с элементами HTML, так и с элементами SVG.[8]

Инструкции SVG оптимизированы для веб-разработки, и отрисовка происходит прямо в браузере. SVG-изображение можно зашить прямо в код веб-страницы, поскольку все команды SVG представлены в виде текста и координат, а не отдельных файлов.

*Inkscape.* Inkscape – это бесплатный векторный редактор, многофункциональный инструмент для создания и работы с художественной и технической графикой.[6]

Являясь кроссплатформенным редактором, Inkscape имеет открытый исходный код, позволяющий любому пользователю внести необходимые коррективы в работу программы.

Особенность Inkscape – использование открытого формата Scalable Vector Graphics.

*Adobe Photoshop.* Adobe Photoshop — многофункциональный графический редактор, разрабатываемый и распространяемый компанией Adobe Systems.

* 1. Сфера применения

Младший школьный возраст – период интенсивной социализации, усвоения различных нравственных норм. Поэтому именно в этом возрасте правомерно уделять значительное внимание духовно-нравственному развитию и воспитанию личности, формированию вектора культурно-ценностных ориентаций. [3]

Реализация содержания этнокультурного образования осуществляется на основе личностно-деятельностного подхода. Рекомендуется отбор педагогических средств, соответствующих эмоционально-действенным характеристикам возрастных особенностей младших школьников, учитывающих субъектно-объектную направленность их познавательной деятельности. [7]

Предполагается применение педагогических технологий развивающего обучения, а также игровых, информационных, коммуникативных, здоровье сберегающих технологий.

Проект можно использовать как учебный, ознакомительный материал для школьных и дошкольных учреждений. Наглядная, интерактивная форма представления помогает лучшему восприятию информации, за счет осознанного взаимодействия с объектом интереса.

Саму форму, функционал проекта можно использовать как каркас для ведения информационного источника, справочника и прочих смежных изданий практического назначения, где упор делается на краткое изложение сведений в систематической форме, с расчетом на выборочное чтение, на отличные темы.

* 1. Описание структуры разрабатываемого сайта

Для описания технического задания будем условно придерживаться следующих пунктов:

* Цели и назначение сайта,
* Аудитория сайта,
* Технические характеристики,
* Содержание сайта (структура сайта с подробным описанием элементов и функций каждой страницы),
* Интерактивные элементы и сервисы,
* Требования к материалам,
* Перенос на хостинг.

Разрабатываемый сайт предназначен для демонстрации культурных особенностей народов Севера. Он будет представлять собой небольшой интерактивный, информационный ресурс.

Материал сайта можно использовать для ознакомительных целей в образовательных учреждениях, ориентировочно на школьный и дошкольный возраст, а также в качестве справочной информации для заинтересованных в культуре народов Севера.

Сайт будет состоять из 3 страниц - главной (начальной, приветственной), страницы с основным функционалом и наполнением, и страницы с контактной и личной информацией:

* на главной странице будет представлена краткая информация о сайте, проекте;
* на второй странице будет представлен основной интерактивный и информационный сегмент сайта;
* на последней странице будет представлена информация о разработчике и о сайте более подробно.

Упомянутый выше основной функционал будет являть собой интерактивный справочник-словарь.

На странице будет большой блок, разделенный на две части. Смотря от уместности с точки зрения визуала и верстки, вертикальное или горизонтальное расположение и распределение элементов будет решено на этапе разработки макетов. В одной будет представлен список с народами, на другой – место для изображения - интерактивной карты.

Участки изображения, представляющие определенный предмет (элемент одежды, украшения, жилище или элемент окружения/быта и т.п.), будут выделены контуром, и к ним будет привязана соответствующая мета – определение или справка.

При наведении на участок изображения с тем или иным предметом, его контур будет подсвечиваться, а при нажатии будет всплывать окно справки.

При выборе названия народа из списка, на место старого изображения-карты, встанет другое, соответствующее по смыслу. также будет еще один блок, куда при переключении будет появляться общая информация о соответствующем народе.

* 1. Концепция и дизайн макетов интерфейса сайта

Дизайн, концепт, макеты страниц были сделаны в Фигма. Было условно определено размещение элементов интерфейса, их можно увидеть в рисунках 1-5, далее:

|  |
| --- |
|  |
| Рисунок 1 – Макет главной страницы |

Представлены на выбор два варианта размещения кнопок и фрейма с контентом для второй страницы с основным интерактивом и функционалом, а также полный макет страницы с блоком описания выбранного народа, ниже на рисунках 2 и 3:

|  |  |
| --- | --- |
|  |  |
| Рисунок 2 – Макет второй страницы (первый вариант) | Рисунок 3 - Макет второй страницы (второй вариант) |

Далее представлены вариант макета второй страницы при раскрытом блоке с информацией о народе, в качестве блока с интерактивом здесь представлен второй вариант, как на рисунке 3, а также макет последней страницы, - можно увидеть на рисунках 4 и 5.

|  |
| --- |
|  |
| Рисунок 4 – Макет второй страницы (вариант с раскрытым текстовым блоком) |

|  |
| --- |
|  |
| Рисунок 5 – Макет 3 страницы |

1. Разработка сайта
   1. Верстка интерфейса и задание стилей для элементов

Для облегчения и ускорения процесса верстки, использовался фреймворк Bootstrap.

Для доступа к инструментам фреймворка Bootstrap, нужно подключить соответствующие ссылки к странице, как показано на рисунках 6 и 7, - актуальные можно взять с официального сайта.

Размещаем ссылки на CSS и JS - файлы, Bootstrap наравне со своими, в логически определенных местах, в начале для стилей, чтобы при загрузке элементов страницы учитывались стили, и JS-файлы в конце после основного контента страницы, чтобы действия написанных программ распространялись на уже полностью загруженные элементы страницы во избежание сбоев.

|  |
| --- |
|  |
| Рисунок 6 – Подключение файлов со стилями css |

|  |
| --- |
|  |
| Рисунок 7 – Подключение файлов javascript |

Формируем основные блоки страниц сайта, используя колоночную систему Bootstrap.

*Колоночная система.* Система Bootstrap Grid состоит из ряда колонок, расположенный внутри одного или нескольких контейнеров.

Используя классы колонок со специальными контрольными точками, можно настроить характер размещения блоков для контента внутри ряда. За счет flexbox, специальных классов для выравнивания, автоматического регулирования размеров блока и прочих возможностей классов инструментов Bootstrap, верстку можно сделать в документе HTML и по мере наполнения получить более-менее наглядную разметку будущего сайта.[10,14]

Общую же настройку стилей, вроде параметров текста на все страницы, отступы и тому подобное, или же более тонкую настройку отображения отдельных элементов можно оставить на CSS.

В качестве примера использования сетки и классов Bootstrap можно показать футер сайта, код которого представлен на рисунке 8, остальные блоки на сайте сверстаны по его подобию:

|  |
| --- |
|  |
| Рисунок 8 – Код HTML футера |

Колонки равномерно распределяются по третям ряда, поскольку в классе, описывающем колонку не указано, то какую часть из 12 будет та занимать, поэтому процесс распределения происходит автоматически. При изменении размеров окна браузера футер адаптируется, сохраняя положение своих элементов, благодаря адаптивной сетке Bootstrap.

Контент представленный внутри центральной колонки размещается ровно посередине за счет дополнительного класса centered, где в стилях указано, что элементы должны выравниваться по центру относительно горизонтальной плоскости. По такому же принципу, по мере нужды будут добавляться и другие предопределенные классы Bootstrap и наши собственные. Как футер отображается в браузере можно увидеть на рисунке 9.

|  |
| --- |
|  |
| Рисунок 9 – Отображение футера в браузере |

Bootstrap предлагает готовые шаблоны-компонентов сайта, к проекту были пристроены следующие:

Навигационная панель. С сайта был скопирован код данного элемента, затем в нужных местах введены свои значения и ссылки, код можно увидеть на рисунке 10.

|  |
| --- |
|  |
| Рисунок 10 – Код HTML навигационной панели |

Можно удобно ориентироваться по сайту - по представленным на навигационной панели ссылкам можно передвигаться по страницам, - активная подсвечивается, как это отображается можно увидеть на рисунке 11.

|  |
| --- |
|  |
| Рисунок 11 – Отображение навигационной панели |

Карусель. Из скачанного файла со стилями Bootstrap, были найдены и скопированы стили для элементов управления каруселью - кнопки и индикаторы слайдера, в них, в имеющиеся атрибуты и свойства, были внесены изменения, таким образом, удалось кастомизировать уже имеющиеся элементы под себя - изменить цвета, добавить картинку на задний фон.

Таким же образом, можно вносить изменения и в другие элементы Bootstrap, которые будут внедрены в проект, настраивать под себя свойства отображения и поведения объектов.

Карусель - хороший инструмент для быстрого и компактного описания сайта, учитывая малый масштаб текста на размещение, к тому же оно приносит разнообразие в визуальное наполнение сайта, ее отображение в браузере можно увидеть на рисунке 12.

|  |
| --- |
|  |
| Рисунок 12 – Отображение карусели-слайдера в браузере |

Карусель на главной странице воспроизводится автоматически при открытии, слайды сменяются через определенные промежутки времени, но пользователь может также переключать их, - для этого по бокам расставлены соответствующие кнопки, также внизу есть индикатор продвижения, указывающий на текущий слайд, а по нажатию на один из них можно перейти на соответствующий.

Также на слайдере, ниже блока с описанием сайта, есть кнопка со ссылкой, по нажатию которой, пользователь может перейти на страницу с основным функционалом сайта.

* 1. Программирование работы функционала

Следует пояснить, что разработка велась, опираясь на принципы схожие с принципами итеративно инкрементальной модели (англ. iterative and incremental development, IID).

Модель IID предполагает разбиение жизненного цикла проекта на последовательность итераций, каждая из которых напоминает «мини-проект», включая все процессы разработки в применении к созданию меньших фрагментов функциональности, по сравнению с проектом в целом. Цель каждой итерации — получение работающей версии программной системы, включающей функциональность, определенную интегрированным содержанием всех предыдущих и текущей итерации. Результат финальной итерации содержит всю требуемую функциональность продукта. Таким образом, с завершением каждой итерации продукт получает приращение — инкремент — к его возможностям, которые, следовательно, развиваются эволюционно.[12]

Для получения представления о работе и логике функционала интерактивных элементов страницы, для определения требований к ним, а также анализа и оценки дальнейших возможностей и ограничений, функционал и структура проекта создавались сначала на более простой модели, далее по мере изменений и усложнения они “прирощались” к коду.

* + 1. Описание функций.

Основной блок на странице с интерактивом разделен на два. Первая часть включает в себя медиа-наполнение, вторая - кнопки для переключения содержимого первой.

На блоке с кнопками cont выставлен слушатель, действующий при нажатии. Функция принимает в обиход только кнопки, элементы “button”. За таргет, элемент, с которым произойдут дальнейшие события, принимается ближайшая нажатая кнопка. Так быстрее и легче осуществить выборку, и можно существенно сократить код, ежели это было бы при использовании циклов, полном прохождении через все участвующие элементы, и оперированием с id кнопок, - он же, перехватывается на этом этапе, после определения таргета, и будет нужен в дальнейшем использовании. Код данного функционального блока представлен на рисунке 13.

На данной стадии проектирования и разработки, элементы для отображения, то есть будущий медиа-контент, - представлены “локально”, - они лежат непосредственно в документе HTML, данная функция лишь меняет характер отображения элементов.

При нажатии на соответствующую кнопку, соответствующий контент показывается и становится активным, а тот что был ранее скрывается и деактивируется, выборка осуществляется по id кнопок.

|  |
| --- |
|  |
| Рисунок 13 – Слушатель кнопок, функция изменения  отображения графической информации внутри блоков |

На первый блок также выставлен слушатель, который реагирует на клики.

В качестве потенциального таргета принимаются границы контура элемента SVG, - если изъясняться более наглядно, то это контур вокруг какого-либо осязаемого предмета на изображении, представленном в этом блоке, - координаты которого описаны в атрибуте-классе “path”, следовательно, данный слушатель (функция) отслеживает его наличие в классах рассматриваемого элемента.

Код данного функционального блока представлен на рисунке 14.

|  |
| --- |
|  |
| Рисунок 14 – Слушатель фрейма с контурами,  функции выборки и появления информации |

После определения таргета, считывается информация, хранящаяся внутри объекта, в рамках данной работы, это название объекта (title), его описание (desc). Они заключаются в ранее созданные элементы заголовка (h2) и абзаца (p), которые предварительно очищаются, затем добавляются в состав блока, отображение которого становится доступным после активации этой функции. Этот блок будет представлять собой, всплывающее окно со справочной информацией.

Код данного функционального блока представлен на рисунке 15.

|  |
| --- |
|  |
| Рисунок 15 – Элементы на добавление в HTML-документ |

Помимо выше упомянутых элементов, блок будет содержать кнопку, которая закрывает его, код можно посмотреть на рисунке 16, ниже:

|  |
| --- |
|  |
| Рисунок 16 – Функция закрытия кнопки |

Как говорилось ранее, интерактивный элемент страницы будет реализован с помощью интерактивной “карты изображения”. Участки изображения выделяются контуром и им назначается соответствующая мета, которая активизируется и показывается пользователю в виде справки при взаимодействии.

* + 1. Описание интерактивной составляющей

SVG — XML язык разметки на основе векторной графики. По сути, это текстовый файл, который является открытым веб-стандартом для описания двумерных векторных изображений без потери качества при масштабировании. У SVG-рисунка есть своя объектная модель (DOM), которыми можно оперировать, применять к ним CSS-правила и привязывать события с помощью JS.[8]

Примерная структура SVG в документе HTML представлена ниже на рисунке 17:

|  |
| --- |
|  |
| Рисунок 17 – Структура SVG |

Отрисовка контуров для обозначения путей SVG-изображения будет произведена в Inkscape. Он оснащен нужными инструментами для работы с векторной графикой, а также там есть удобный редактор XML, откуда будут браться координаты для контуров.

|  |
| --- |
|  |
| Рисунок 18 – Отрисовка контура в окне редактора Inskape |

На изображении вокруг предметов, которые надо подчеркнуть, с помощью кривых, кривых Безье обрисовывается контур, как показано на рисунке 18. После замыкания, на панели с редактором XML отобразится текстовое описание полученного контура-рисунка в формате SVG. Из этого описания необходимо скопировать значения атрибута d - это описание координат контура, - и поместить внутри тега path.

|  |
| --- |
|  |
| Рисунок 19 – Контур вокруг парного предмета |

Копируем каждое значение координат контуров предметов, представляющих группу как показано на рисунке 19, - это может быть парные предметы, вроде обуви, пары рукавиц и т.п., - так же, как и обычные, внутрь тега path, поскольку описание контура в пределах тега группы будет распространяться на все контуры, координаты которых находятся в этой группе.

Внизу на рисунке 20 представлен пример описания SVG для некоторых предметов изображения группы народа “Чукчи”.

|  |
| --- |
|  |
| Рисунок 20 – Описание SVG для группы парных, составных предметов |

Для определения координат предмета, части которого не удается обрисовать в один контур, сначала вокруг этих частей отдельно вырисовываются контуры, а затем производится объединение контуров.

Объединение проводится с помощью комбинации Ctrl+K, или выбора соответствующей операции на панели управления. На рисунке 21 представлен уже объединенный контур вокруг украшения на голову девушки, он специально выделен оранжевым. После можно получить координаты объединенного контура и спокойно поместить в код.

|  |
| --- |
|  |
| Рисунок 21 – Контур вокруг предмета,  который невозможно поместить в единый контур |

Теперь по наведению на участок одной из частей данного составного контура будет выделяться весь контур предмета, как показано на рисунке 22.

|  |
| --- |
|  |
| Рисунок 22 – Отображение в браузере |

Настройка стилей отображения, позиционирование векторного изображения относительно других элементов страницы будет сделано в CSS, а сценарии при взаимодействии с пользователем и прочие активности будут описаны в JS.

Далее описываем CCS-правила для контура, таким образом, чтобы они визуально выделялись на фоне основного изображения при взаимодействии с пользователем.

|  |
| --- |
|  |
| Рисунок 23 – Описание стилей для контуров |

На рисунке 23 описано позиционирование объекта SVG в документе, относительно других элементов, а также “поведение” контуров при наведении пользователя. Важно, чтобы векторный рисунок с контурами и изображение поверх которого они будут расположены не расходились, полностью, вплотную накладывались друг на друга.

* 1. Серверная сторона. Динамические данные

На этом этапе, у нас готов своеобразный прототип сайта, со сверстанными страницами, и рабочим функционалом с более-менее выверенной логикой. Однако, здесь в рамках проектирования и разработки включены не все данные, только показательные примеры, а сам разрабатываемый информационный источник предполагает работу с большим массивом данных. Даже с учетом выбора небольшого списка народов, объем данных будет весьма внушительный, что плохо скажется на общей производительности сайта. Поэтому возникает необходимость во внедрении серверной части сайта, которая будет отвечать за хранение данных.

В имеющейся структуре HTML, в блоке с медиа контентом, отображаемые элементы выстраиваются по единому шаблону, во избежание большого числа однообразных строк кода, следует упорядочить, структурировать и определить эти данные в такой формат, из которого будет легко вытащить их и предоставить по запросу пользователя, согласно вышеупомянутому шаблону.

Таким образом, в некотором хранилище должны будут содержаться изображения, тексты с общим описанием народов и их быта, и нужно создать и описать такой объект, с помощью которого можно будет легко и быстро извлечь вышеупомянутые элементы в структурированном виде.

Объект будет сформирован в формате json, опираясь на имеющуюся структуру SVG, и будет выстроен следующим образом, как показано на рисунке24:

|  |
| --- |
|  |
| Рисунок 24 – Описание структуры объекта JSON |

Выбор данного формата обусловлен тем, что он отлично подходит для хранения и передачи структурированных данных по сети. JSON легко сочетается с любой современной средой программирования, в частности, код для введения и обработки данных в этом формате присутствует в языках PHP, Python, Java и Ruby.[11]

Как видно на рисунке, структура объектов выстроена на паре “ключ-значение”. В качестве основных объектов представлена мета по каждому отдельному народу, включающая в себя: название, справочную информацию, ссылку на изображение, а также данные SVG - данные об области просмотра всего SVG рисунка, сгруппированные данные отдельных контуров (координаты, справка, название).

От серверной части сайта требуется только формировать ответы на запросы по получению данных (GET-запросы).

* + 1. Репозиторий. Сервер. Хранение данных на веб-сервере

Сервер — это компьютер с высокой вычислительной мощностью и круглосуточной бесперебойной работой, предназначенный для хранения информации и обеспечения доступа к ней с любого устройства пользователя. Его можно запустить и на своем локальном компьютере, если мощность отвечает требуемым запросам, и объем памяти достаточный. Однако, потребность в постоянной активной работе и дополнительном обслуживании, вынуждает отказаться от этой идеи.[6]

Поэтому рассмотрим вариант удаленного сервера - веб-сервера, в нашем случае. Отлично подойдет хостинг сервис Github, предоставляющий бесплатную среду для размещения кода и своих проектов. К тому же, поскольку наш сайт не требует сложных взаимодействий с пользователями, он практически статический, что не возникнет проблем с оптимизацией хостинга.

Также воспользуемся технологией Github Pages для публикации сайта.

Был создан репозиторий my-site - туда добавлена папка с файлами проекта, оттображение файлов и папок проекта показано на рисунке 26.

Файл db.json, в котором все нужные нам данные, был загружен в папку text.

|  |
| --- |
|  |
| Рисунок 25 – Файлы репозитория |

* + 1. Описание работы серверной части. Формат данных. Запросы

Реализовывать запросы будем, используя асинхронные запросы. При таком раскладе, после отправки запроса на сервер и получения соответствующего ответа страница обновится, не перезагружаясь - нужные операции пройдут фоново.

|  |
| --- |
|  |
| Рисунок 26 – Запрос к серверу на получение данных |

Для отправки запроса и подгрузки данных будем пользоваться методом fetch. Этот метод предполагает наличие url-адреса файла, в котором будут описаны данные на извлечение, а также дополнительные параметры в виде используемого метода, заголовка и т.п., но у нас он будет по умолчанию, - простой GET-запрос, скачивающий содержимое по url. Код данного функционального блока представлен на рисунке 26.

Тело ответа на запрос, получаем в commits, используя метод. json, который декодирует ответ в формате JSON. Для наглядности, на рисунке 27 представлен выведенный в консоль ответ.

|  |
| --- |
|  |
| Рисунок 27 – Полученный ответ на запрос в консоли браузера |

Теперь имея в распоряжении этот json-объект, мы можем обращаться к его элементам и проводить с ними или с их помощью некоторые операции.

На тестовой страничке есть две кнопки и пустой блок. По нажатию любой из них, учитывая id выбранной кнопки, будет показан соответствующий рисунок в блоке.

Код данного функционального блока представлен на рисунке 28.

|  |
| --- |
|  |
| Рисунок 28 – Функция картинки на выбор |

Далее, сформируем некий шаблон для SVG, который будет добавлен на страницу HTML, и поместим его в “контейнер”. Данный “контейнер” представляет собой фрагмент документа, который может содержать произвольное количество элементов DOM. Их всех сразу можно в нужный момент добавить к самому документу - вывести на страницу. Для этого, достаточно обратиться к этому “контейнеру”, он сам “растворяется”, вместо него в указанное место добавляются его дочерние узлы. Структура и код описывающий данный функциональный блок представлены на рисунках 29 и 30 соответственно.

|  |
| --- |
|  |
| Рисунок 29 – Создание элементов, которые  будут добавлены в фрагмента документа |

Добавим этот сценарий для кнопок. Теперь, по мере выбора кнопки, учитывая её id, по шаблону будет формироваться соответствующий SVG, и он будет добавлен на страницу - в код HTML-документа. Разницу можно увидеть на рисунках 31 и 32.

|  |
| --- |
|  |
| Рисунок 30 – Оформление SVG на добавление |
|  |
| Рисунок 31 – Элементы тестовой странички  до добавления фрагмента |
|  |
| Рисунок 32 – Элементы тестовой странички  после добавления фрагмента |

К сожалению, серверную часть не удалось внедрить в проект ввиду нехватки знаний и опыта. Данную наработку следует внедрить в будущем, в случае расширения проекта и увеличения базы данных. Сейчас для демонстрации сайта можно обойтись небольшим числом данных и существующим функционалом.

1. Тест и отладка
   1. Публикация сайта на GitHub Pages

Теперь, имея рабочую заготовку сайта, можно попробовать разместить её в сеть. Для этого воспользуемся функцией GitHub Pages.

Переходим во вкладку Settings в интерфейсе репозитория, разделе GitHub Pages нужно установить в качестве источника файлов страницы ветку main созданного репозитория и сохранить, как показано на рисунке 33. После сформируется ссылка на будущий сайт, но нужно будет немного подождать пока не изменится статус публикации на готовый.

|  |
| --- |
|  |
| Рисунок 33 – Подключение к GitHub Pages |

Сформированная ссылка, по сути, ведет на репозиторий, - чтобы она открывала именно страницу сайта, её HTML-файл должен лежать в корне проекта.

Однако, структура сайта у нас устроена таким образом, что страницы сайта и прочие файлы находятся в специально отведенных папках, перестраивать ее будет накладно. Поэтому создадим фиктивную страницу, файл index.html, с которой будет идти переадресация на нужные страницы, и поместим её в корень проекта, как показано на рисунках 34 и 35.

|  |
| --- |
|  |
| Рисунок 34 – Добавление файла index.html в корень репозитория |
|  |
| Рисунок 35 – Код файла index.html на переадресацию |

* 1. GitHub Desktop. Фиксация и отслеживание изменений.

Для удобства внесения изменений и правок в проект клонируем наш репозиторий на компьютер с помощью десктопной версии GitHub. Сама программа уже установлена на компьютер и вход в аккаунт произведен. Во вкладке Current repository, в блоке ниже панели меню, выбираем Add - Clone repository - появится список доступных, существующих на аккаунте репозиториев, как на рисунке 36, выбираем нужный.

|  |
| --- |
|  |
| Рисунок 36 - Список доступных репозиториев |

Затем нужно немного подождать пока завершится процесс клонирования репозитория. После, он отобразиться в списке локально загруженных репозиториев, а в самой папке с проектами GitHub появится папка репозитория, как на рисунках 37 и 38.

|  |  |
| --- | --- |
|  |  |
| Рисунок 37 – Список склонированных репозиториев | Рисунок 38 – Папка добавленного репозитория |

Структура такая же как была в репозитории, можно увидеть на рисунках 40 и 41.

|  |  |
| --- | --- |
|  |  |
| Рисунок 40 – Корневая часть | Рисунок 41 – Папки проекта под  структуру сайта |

Теперь любые локально внесенные изменения, будь то новая строка в коде, добавленный или отредактированный медиафайл (текст, картинка и т.п.), будут автоматически фиксироваться программой.

В списке будут отображены файлы, места, в которых произошли изменения, последние можно подробно рассмотреть, также сравнить версии с предыдущей что можно увидеть на рисунке 42. Изменения обязательно нужно будет описать и “принять” (закоммитить), чтобы данные записались и были готовы к отправке на сервер, или куда там нужно, а, чтобы правки вступили в силу и на удаленном исходнике, их нужно “отправить” (запушить), изменения тогда загрузятся уже на сервер.

А также в программе можно посмотреть и отследить историю изменений, и в случае надобности вернуться к одной из предыдущих версий, как на рисунке 43.

|  |  |
| --- | --- |
|  |  |
| Рисунок 42 - Изменения | Рисунок 43 – История изменений |

* 1. Тест опубликованного сайта.

Главная страница со слайдером, рисунок 44. Слайдер автоматически прокручивается через определенные отрезки времени, можно управлять этим процессом, как и было задумано. Ссылки на панели навигации работают, активная страница выделяется. Ссылки на соцсети по умолчанию деактивированы. По нажатию на кнопку “Попробовать” происходит переход к странице с интерактивом.

|  |
| --- |
|  |
| Рисунок 44 – Главная, приветственная страница |

После перехода на вторую страницу, страницу с интерактивом, нас встречает практически пустое окно, что выглядит, на самом деле, не очень хорошо и следовало бы добавить какое-нибудь дополнительное наполнение, возможно, всплывающее окно с подсказкой или туториалом по пользованию данным справочником. По завершению “обучения” оно бы свернулось, но ссылка для его активации была бы где-то на видном месте, для повтора получения справки по желанию пользователя.

В окне две кнопки, в качестве показательного примера, в проект введены данные только о двух народах, как на рисунке 45.

|  |
| --- |
|  |
| Рисунок 45 – Вторая страница, до взаимодействия |

По нажатию одной из кнопок, разворачивается блок с описанием соответствующего народа, как на рисунке 46, ниже:

|  |
| --- |
|  |
| Рисунок 46 – Вторая страница после взаимодействия,  блок с описанием народов |

Блок с кнопками, вместе с появившейся картинкой, “съезжает” вниз. При переключении на другую кнопку, блок сверху с информацией о народе обновляется, также и картинка снизу. Запланированное выравнивание элементов и отступы соблюдены, ничего не рушится. При наведении или активации кнопка визуально выделяется, как и было описано, что можно увидеть на рисунке 47.

|  |
| --- |
|  |
| Рисунок 47 – Изображение народа при активированной кнопке |

При наведении на участок изображения, где есть контур, тот характерно затемняется, визуально выделяясь, как положено. По истечению нескольких секунд рядом появляется название выделенного контуром объекта, как на рисунке 48.

|  |
| --- |
|  |
| Рисунок 48 – Поведение контура рисунка  на наведение пользователя |

По нажатию на данный контур появляется окно со справочной информацией, как на рисунке 49. Оно имеет определенную максимальную высоту, по достижению которой, блок с текстом становится прокручиваемым. Информация (название, описание) соответствует выделенному объекту, отступы соблюдены. Кнопка закрыть работает.

Если при активном окне выбрать другой контур, то информация в окне соответственно меняется, но данное действие, по большей части, будет сложно осуществить, поскольку данное окно почти полностью перекрывает изображение и соответственно другие контуры. Следует добавить для пользователя возможность передвигать это окно, на странице, как ему будет удобнее, или пересмотреть место появления окошка.

|  |
| --- |
|  |
| Рисунок 49 – Окно с описанием |

Последняя страница с информацией о разработчике и о самом сайте более подробно отображается как надо, рисунок 51.

|  |
| --- |
|  |
| Рисунок 51 – Последняя страница, о себе |

Сайт за счет использования адаптивной сетки Bootstrap и дополнительно прописанных классов для обеспечения адаптивности изображений, автоматически подстраивается под размеры окна браузера, в целом и по большей части без нарушений со стороны отступов и выравнивания. Однако, здесь всё же не предусмотрены спецификации для различных экранов. Размер текста не меняется, не подстраивается, нет специальной верстки для других, более мелких разрешений, чтобы при достижении определенной точки (точки остановы) элементы выстраивались таким образом, чтобы это гармонично выглядело для определенного размера экрана, что можно увидеть на следующих скриншотах, представленных на рисунках 51-55.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Рисунок 51 – Приветственная страница. | Рисунок 52 – Второй экран, панель навигации. | Рисунок 53 – Описание народа. |

|  |  |
| --- | --- |
|  |  |
| Рисунок 54 – Окно с описанием предмета | Рисунок 55 – Изображение народа |

В последующем расширении проекта, следует учесть описанные на этапе теста недочеты и рекомендации для повышения качества и эффективности разработки.

ЗАКЛЮЧЕНИЕ

Целью данной курсовой работы являлось разработать и создать сайт.

При подготовке курсовой работы была изучена соответствующая литература и отобран наиболее удачный материал. Для реализации функционала изучение материала велось посредством анализа и синтеза источников, проводилось сравнение и обобщение схожих или смежных по тематике разработок, а сама разработка велась, опираясь на принципы схожие с принципами итеративно инкрементной модели, с использованием методов объектно-ориентированного и функционального программирования.

При выполнении данной курсовой работы были более основательно освоены подходы при создании сайтов, как с клиентской стороны, так и серверной, а также улучшены навыки программирования и изучены новые инструменты для верстки и разработки сайтов.

Инструментальной средой разработки послужила MS Visual Studio Code и Notepad++.

По ходу курсовой были также рассмотрены различные технические среды и средства разработки и редактуры. Изучена и проанализирована сфера применения и выявлена аудитория пользователей. Проведен анализ существующих аналогов сайтов, источников на эту тему и на их основе приняты решения по созданию концепции и дизайна сайта, далее были определены требования к проекту и составлены его структура и план, а также описана логика поведения элементов. С помощью программ прикладного назначения был написан сам сайт. Тест и отладка сайта проводились посредством работы с системой управления версиями, которая позволяла быстро и оперативно зафиксировать локальные изменения и отправлять данные на выложенный сайт для наглядности.

Как результат, была реализована разработка сайта, демонстрирующего культурные особенности малых народов Севера, представляющий собой интерактивный справочник.

Данный проект может использоваться как справочный материал для детей школьного и дошкольного возраста, а также для заинтересованных в культуре малых народов. Он способствует популяризации культурного и этнического наследия, развитию представления о быте малых народов.

Назначенные цели и задачи курсовой были выполнены.

СПИСОК ЛИТЕРАТУРЫ

1. ГОСТ 7.32-2017 ОТЧЕТ О НИР. Структура и правила оформления. — URL: https://files.stroyinf.ru/Index/655/65555.htm (дата обращения: 05.07.2022).
2. Государственная публичная научно-техническая библиотека России. Новый систематизированный толковый словарь. - URL: <http://www.gpntb.ru/win/book> (Дата обращения 20.06.2022)
3. Авексентьев, А.В Этнические проблемы современности и культура межнационального общения: учеб. пособие [Текст] / А.В. Авексентьев, В.Л. – Ставрополь, 1983. – 222 с.
4. Кантор И. Современный учебник Javascript. Часть 2. URL: https://library-it.com/wp-content/uploads/2020/12/ilja\_kantor\_sovremennyj\_uchebnik-2chast.pdf (дата обращения: 11.12.2021).
5. Никсон Робин. Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и HTML5, 5-е изд — Питер-Петербург, 2019 — 816 с.
6. Онлайн-энциклопедия Википедия. – URL: <https://ru.wikipedia.org/> (дата обращения 01.06.22)
7. О.М.Макеева. ЭТНОКУЛЬТУРНОЕ ОБРАЗОВАНИЕ В СОВРЕМЕННОЙ РОССИЙСКОЙ ОБРАЗОВАТЕЛЬНОЙ СИТУАЦИИ— URL: https://college4.ru/2-uncategorised/37-avtorskaya-statya-etnokulturnoe-obrazovanie-v-sovremennoj-rossijskoj-obrazovatelnoj-situatsii-makeeva-o-m (дата обращения: 02.03.22).
8. Журнал “Код» программирование без снобизма. SVG — URL: https://thecode.media/SVG/ (дата обращения: 27.05.22).
9. Императивное и декларативное программирование — URL: https://tproger.ru/experts/imperative-and-declarative-programming (дата обращения: 28.06.22).
10. ИтШеф. Что такое Bootstrap и зачем он нужен. — URL: https://itchief.ru/bootstrap/introduction (дата обращения: 17.05.22).
11. О формате JSON — URL: https://apix-drive.com/ru/blog/useful/chto-takoe-json (дата обращения: 25.05.22).
12. Процесс разработки ПО. Методологии разработки. — URL: https://intellect.icu/protsess-razrabotki-programmnogo-obespecheniya-vodopadnaya-iteratsionnaya-i-spiralnaya-model-i-metodologiya-razrabotki-5186 (дата обращения: 19.02.22).
13. Atlassian. What is git. Tutorial. — URL: https://www.atlassian.com/ru/git/tutorials/what-is-git (дата обращения: 20.03.22).
14. Bootstrap. Documentation. — URL: https://bootstrap-4.ru/ (дата обращения: 20.01.22).
15. Dudley Story. The new code. — URL: http://thenewcode.com/537/SVG-Responsive-Imagemap-with-Text-Information-Popups (дата обращения: 28.12.21).
16. GeekBrains. Что такое CSS простыми словами. — URL: https://gb.ru/posts/chto-takoe-css-obyasnyaem-prostymi-slovami/ (дата обращения: 21.05.22).
17. Hexlet. Введение в разработку. Хостинг — URL: https://ru.hexlet.io/courses/intro\_to\_web\_development/lessons/hosting/theory\_unit (дата обращения: 28.02.22).
18. Skillfactory. Про HTML. — URL: https://blog.skillfactory.ru/glossary/html/ (дата обращения: 26.03.22).
19. vc.ru Создание сайта на GitHub. — URL: https://vc.ru/dev/133908-cozdanie-sayta-na-github (дата обращения: 26.04.22).

ПРИЛОЖЕНИЕ А

***(обязательное)***

**Исходный код программ**