ФГБОУ ВО «Ивановский государственный

энергетический университет имени В.И. Ленина»

Факультет информатики и вычислительной техники

Кафедра программного обеспечения компьютерных систем

**Отчет по самостоятельной работе**

**по дисциплине “Веб-разработка”**

Выполнил:

студент гр. 2-41 Прохоров М. В. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_

(подпись) (дата)

Руководитель:

доцент каф. ПОКС Зубков В.П. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_

(подпись) (дата)

# Задание

Создать веб-сайт, содержащий информацию по теме самостоятельной работы. Во время данного процесса выполнить следующие группы шагов:

1. Создание HTML-документа
   1. Создать структуру документа
   2. Заполнить метаинформацию
   3. Добавить Favicon
   4. Заполнить содержимое документа
   5. Проверить корректность разметки
2. Изображения. Таблицы. Многостраничный веб-сайт
   1. Создать новую страницу
   2. Задать гиперссылки на страницах
   3. Добавить изображение
   4. Добавить таблицу
   5. Проверить корректность разметки
3. Использование CSS для оформления веб-страниц
   1. Создать файл каскадных таблиц стилей
   2. Создать стили элемента body
   3. Выровнять текст заголовков
   4. Создать стили элемента header
   5. Создать стили элемента nav
      1. Задать стили элемента ul потомка nav
      2. Задать стили элемента li потомка nav
      3. Задать стили элемента a потомка li потомка nav
      4. Задать стили при наведении на элемент a потомка li потомка nav
   6. Создать стили элемента article
   7. Создать стили элемента footer
   8. Применить результат ко всем страницам веб-сайта
   9. Проверить корректность разметки
4. Применение каскадных таблиц стилей
   1. Создать новую страницу
   2. Добавить содержимое страницы
      1. Добавить содержимое страницы
      2. Добавить текст
      3. Добавить таблицу
   3. Настроить оформление элементов страницы
   4. Проверить корректность разметки и стилей
5. Создание формы ввода
   1. Создать новую страницу
   2. Использовать инструменты разработчика браузера
   3. Придумать форму анкеты
   4. Сверстать форму анкеты
   5. Проверить корректность разметки и стилей
6. Создание скриптов на языке JavaScript
   1. Создать новую страницу
   2. Создать файл со скриптами
   3. Добавить вызов функций
   4. Создать скрипты
   5. Проверить код
7. JavaScript и манипулирование DOM
   1. Создать файл со скриптами
   2. Создать скрипты манипулирования DOM
   3. Проверить код JavaScript
8. JavaScript и взаимодействие с сервером
   1. Создать файл со скриптами
   2. Создать скрипты получения данных от сервера
   3. Проверить код JavaScript
9. Применение библиотеки jQuery
   1. Создать файл со скриптами
   2. Подключить библиотеку jQuery
   3. Создать скрипты с использованием jQuery
   4. Проверить код JavaScript
10. Применение компонентов интерфейса Bootstrap 4
    1. Создать HTML-страницу
    2. Подключить библиотеку Bootstrap 4
    3. Создать разметки страницы с использованием Bootstrap 4
    4. Проверить корректность разметки и стилей

# Ход работы

Для выполнения первой группы шагов необходимо было использовать такие элементы, как DOCTYPE для обозначения типа документа, html для обозначения всего документа, head для обозначения заголовка документа, body для тела документа, footer для «подвала» документа. Использовались также теги ul, li для создания маркированного списка, article для статьи, aside для второстепенной части статьи, nav для полосы кнопок навигации.

Для добавления метаинформации были использованы теги типа meta, в каждом из которых содержалась информация по описанию, ключевым словам и автору сайта. Для подключения Favicon использовался тег link следующего вида: <link rel="Shortcut icon" href="favicon.ico" type="image/x-icon"><link rel="stylesheet" href="styles.css">

После применения полученных знаний, была создана страница, которая в дальнейшем была использована как шаблон для следующих страниц самостоятельной работы.

Для выполнения второй группы шагов была создана страница по шаблону самой первой страницы. Для создания гиперссылок на другие страницы необходимо использовать элемент a, в атрибуте href которого указывается непосредственно ссылка на страницу. Для добавления изображений используется тег img, для нумерованных списков – ol, для таблиц – table для обозначения всей таблицы, tbody для обозначения тела таблицы, tr для строк таблицы, td для клеток таблицы. Для объединения ячеек используется атрибут rowspan элемента td.

Для выполнения третьей группы шагов нужно было создать файл стилей и подключить его ко всем страницам, что делается с помощью элемента link вида <link rel="stylesheet" href="styles.css">, который помещается в заголовок документа. Сами стили, необходимые для выполнения данной группы шагов самостоятельной работы, представлены на рисунке 1.

body {

background-color: #f1f1f1;

font-family: "Times New Roman", Times, serif;

width: 90%;

margin: 0;

padding: 20px;

}

h1, h2 {

background-color: white;

padding: 20px;

}

header {

background-color: white;

padding: 20px;

}

nav ul {

list-style-type: none;

overflow: hidden;

background-color: #333;

margin: 0;

padding: 0;

}

nav li {

float: left

}

nav li a {

display: block;

color: #FFF;

text-align: center;

text-decoration: none;

padding: 14px 16px;

}

nav li a:hover {

background-color: #111;

}

article {

background-color: white;

padding: 20px;

}

footer {

background-color: #ddd;

margin-top: 12px;

padding: 10px;

}

Рисунок 1. Созданная таблица стилей

После выполнения первых трёх групп шагов, были получены страницы, представленные на рисунках 2 и 3

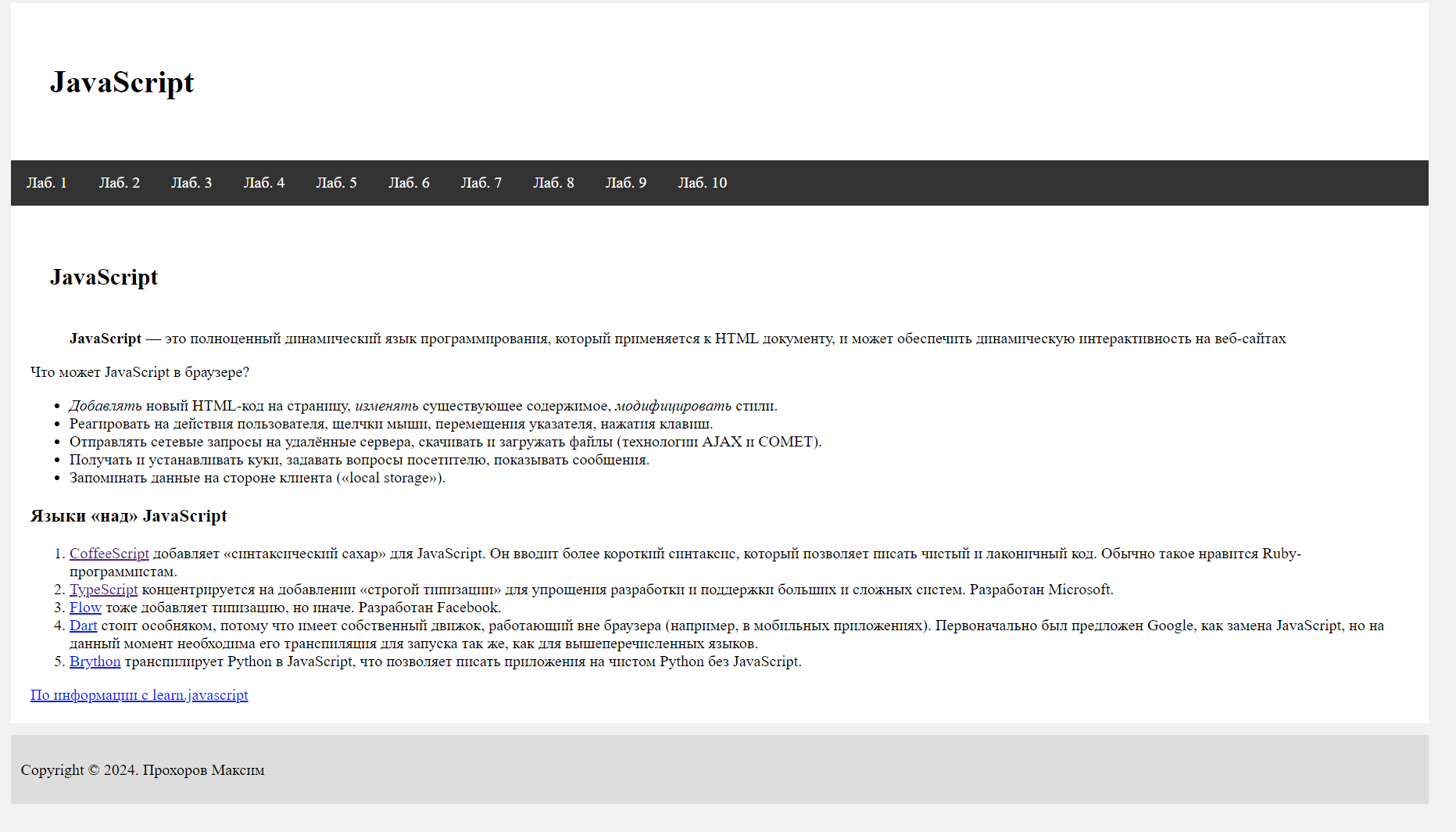


Рисунок 2. Первая страница сайта после выполнения первых трёх групп шагов

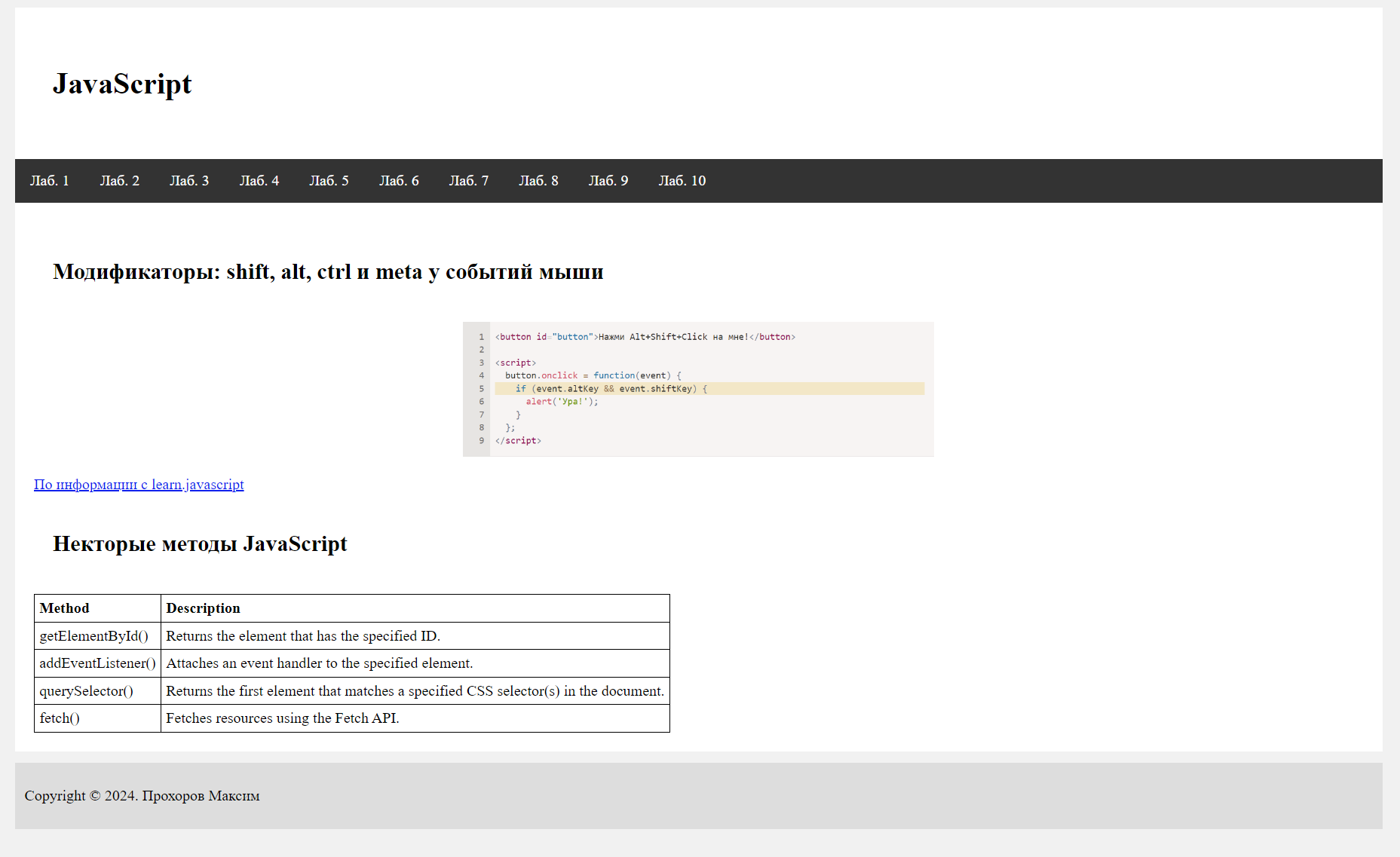


Рисунок 3. Вторая страница сайта после выполнения первых трёх групп шагов

В дальнейшем созданные стили будут использоваться и в последующих создаваемых страницах сайта.

Для четвёртой группы шагов, аналогично предыдущему, был создан файл стилей, который был подключён только к только что созданной странице для данной группы шагов.

Для задания самих стилей необходимо было создать классы стилей, обращение к которым происходит через символ точки перед именем класса. Также были использованы псевдоклассы first-child (первый дочерний элемент), before (стиль до самого элемента), last-child (последний дочерний элемент), nth-child (дочерний элемент n-го номера), hover (стиль при наведении на элемент), visited (стиль посещённого элемента). Для того, чтобы обозначить требования к принадлежности элемента к другому, используется их перечисление через пробел. Например, article p:last-child означает, что стиль будет применён только к тем последним абзацам текста, которые принадлежат элементу article.

Применяя полученные знания, была создана таблица стилей, представленная на рисунке 5.

article {

background-color: bisque;

}

article > h2:first-child {

border: 1px solid black;

text-decoration: underline;

background-color: transparent;

}

article > p:last-of-type {

width: 212px;

height: 100px;

overflow-y: scroll;

background-color: aliceblue;

}

.flex-container {

display: flex;

flex-direction: row;

justify-content: space-between;

}

img {

object-fit: cover;

border: 2px dashed black;

opacity: .5;

}

ol {

counter-reset: list-counter;

list-style: none;

}

ol > li {

counter-increment: list-counter;

margin-bottom: 5px;

}

ol > li::before {

content: counter(list-counter) '. ';

margin-right: 5px;

}

ol > li:last-child::before {

content: "12. ";

}

ul {

list-style: none;

padding: 0;

}

ul > li {

display: inline;

margin-right: 10px;

font-weight: bold;

}

table th {

background-color: gray;

}

table tr:nth-child(even) {

background-color: aliceblue;

}

article a {

text-decoration: none;

}

article a:hover {

text-decoration: underline;

}

article a:visited {

color:blueviolet;

}

**Рисунок 4. Созданная таблица стилей**

После подключения таблицы стилей к созданной до этого странице, она примет вид, показанный на рисунке 5.

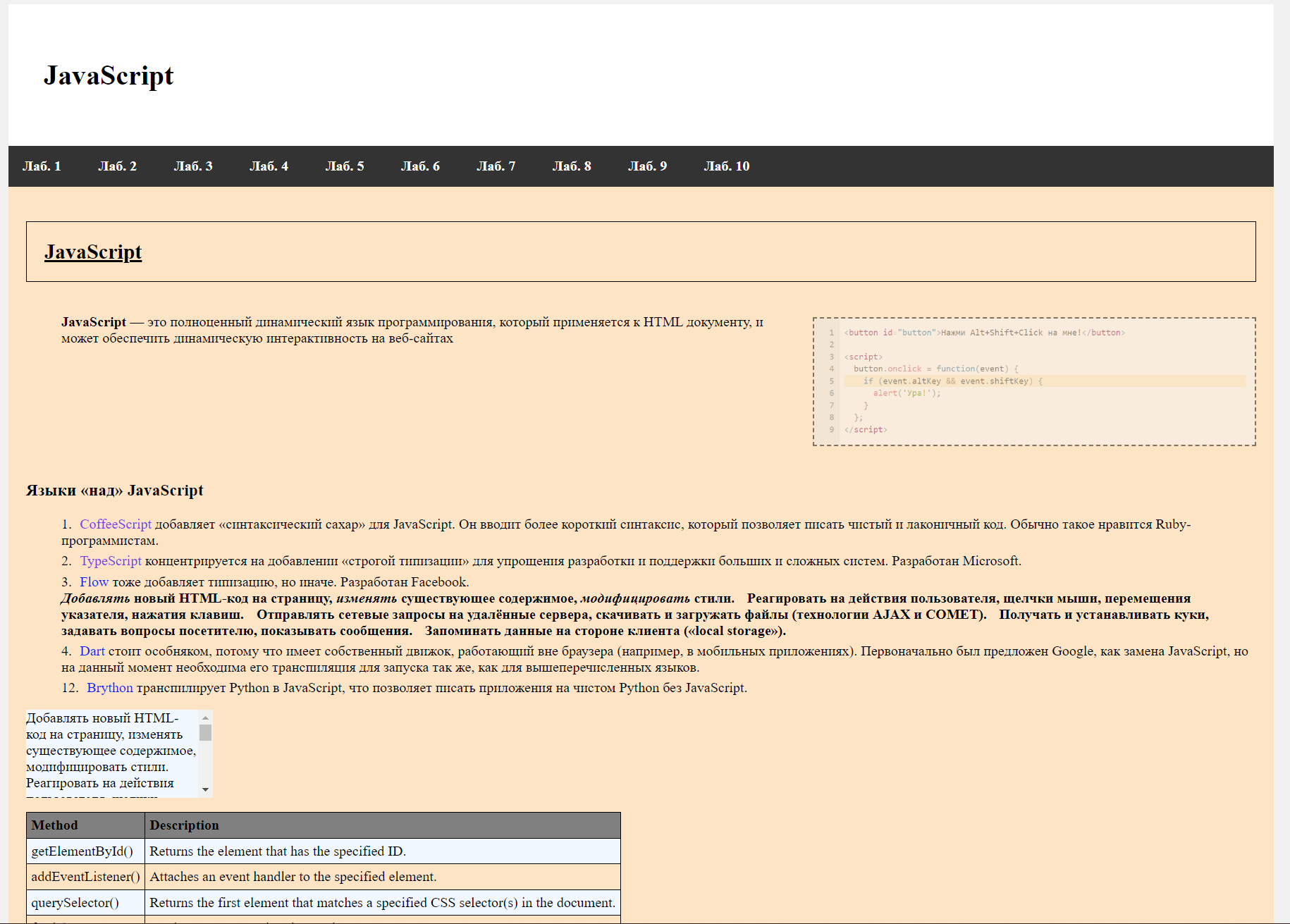


Рисунок 5. Созданная страница после подключения стилей

Для выполнения пятой группы шагов необходимо было создать форму. Для этого нужно использовать элемент form, внутри которого добавляются элементы типа input, в атрибутах которого обязательно указание значения (value) и имени (name) поля формы. Также были использованы теги fieldset для создания группы селекторов (там, где требовалось выбрать один вариант), select и option для создания выпадающего списка и его элементов соответственно, textarea для создания блока для ввода текста. Для отправки значений формы используется элемент button типа (type) submit, для сброса – кнопка типа reset. Для указания обязательности выбора или ввода значения в элемент формы используется атрибут required.

Для выполнения шестой группы шагов была создана страница, к которой был подключён скрипт JS с помощью тега script, в атрибуте src которого был указан путь до самого скрипта.

В самом коде скрипта были использованы конструкции, во многом похожие на их аналоги в других языках программирования:

* if … else для условий
* let … = … и var … = … для объявления переменных
* function (…) {…} для объявления функций
* return … для определения возвращаемого значения функции
* new … для создания объекта

Помимо этого, были использованы следующие встроенные функции:

* alert для вывода сообщения в браузере
* confirm для создания запроса подтверждения
* console.log для вывода сообщения в консоль

Для создания объекта использовалась функция, внутри которой идёт обращение к this.

Для вызова скрипта по нажатию кнопки, используется элемент button, в атрибуте onclick которого указывается вызываемый код.

Применяя полученные знания, была создана страница, представленная на рисунке 6, на котором также показывается результат выполнения всех созданных функций.

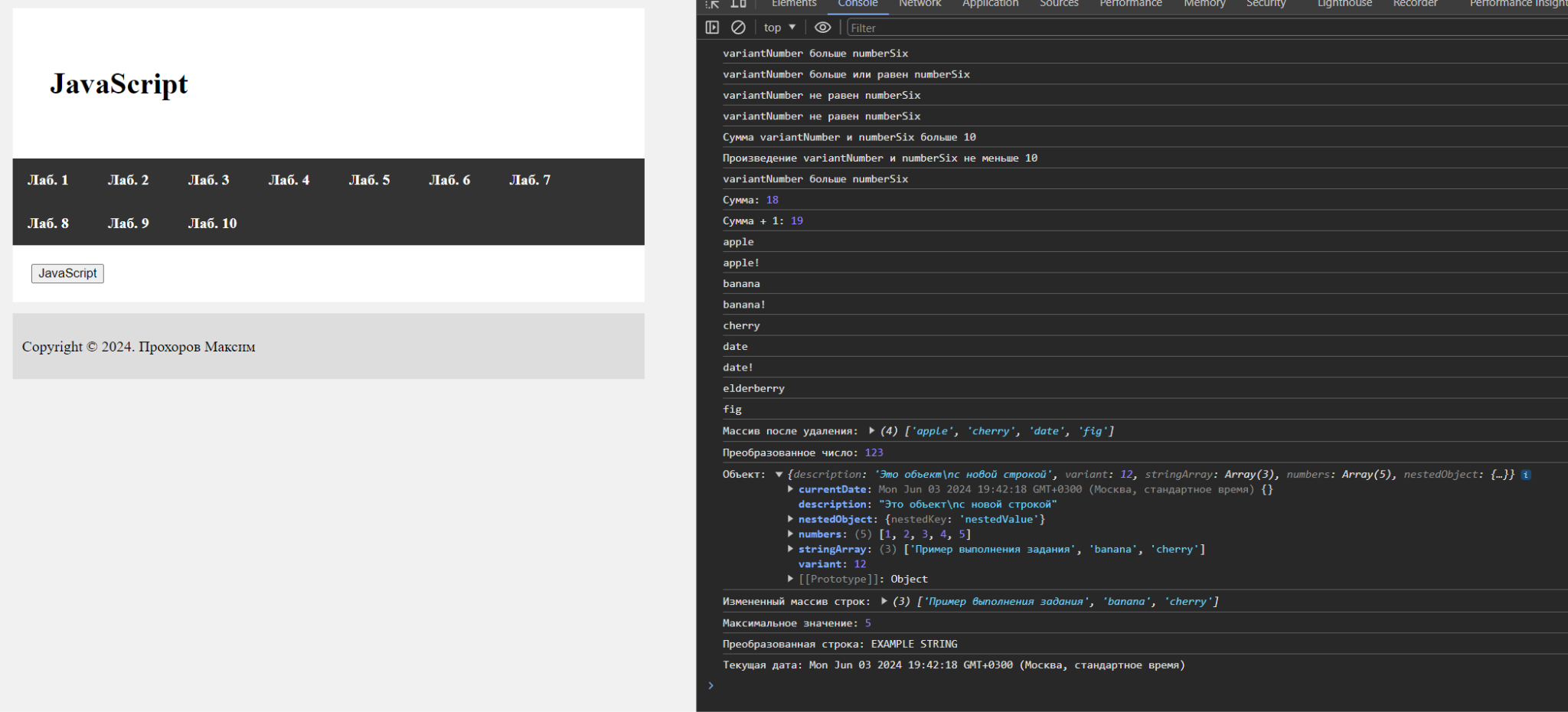


Рисунок 6. Результат выполнения созданных функций

Седьмая группа шагов предполагала создание скрипка, который должен был быть подключен к странице созданной ранее формы. В этом скрипте необходимо было осуществить манипулирование DOM страницы во время её загрузки. Для этого с помощью метода document.addEventListener был добавлен обработчик события DOMContentLoader.

Для создания элемента используется функция document.createElement, в аргументе которой указывается название тега элемента, чтобы добавить к ним атрибуты, используется setAttriburte. Чтобы добавить к существующему элементу другой, используется метод append в аргументы которого передаём элемент, который мы присоединяем. Помимо них, также использовался метод insertBefore для вставки элемента перед другим. Для редактирования текста внутри элемента используется свойство элемента innerText.

Для формирования объекта используется синтаксис название\_объекта = {ключ: значение, …}

После выполнения данной группы шагов, страница формы, созданной в пятой группе шагов, примет следующий вид, показанный на рисунке 7.

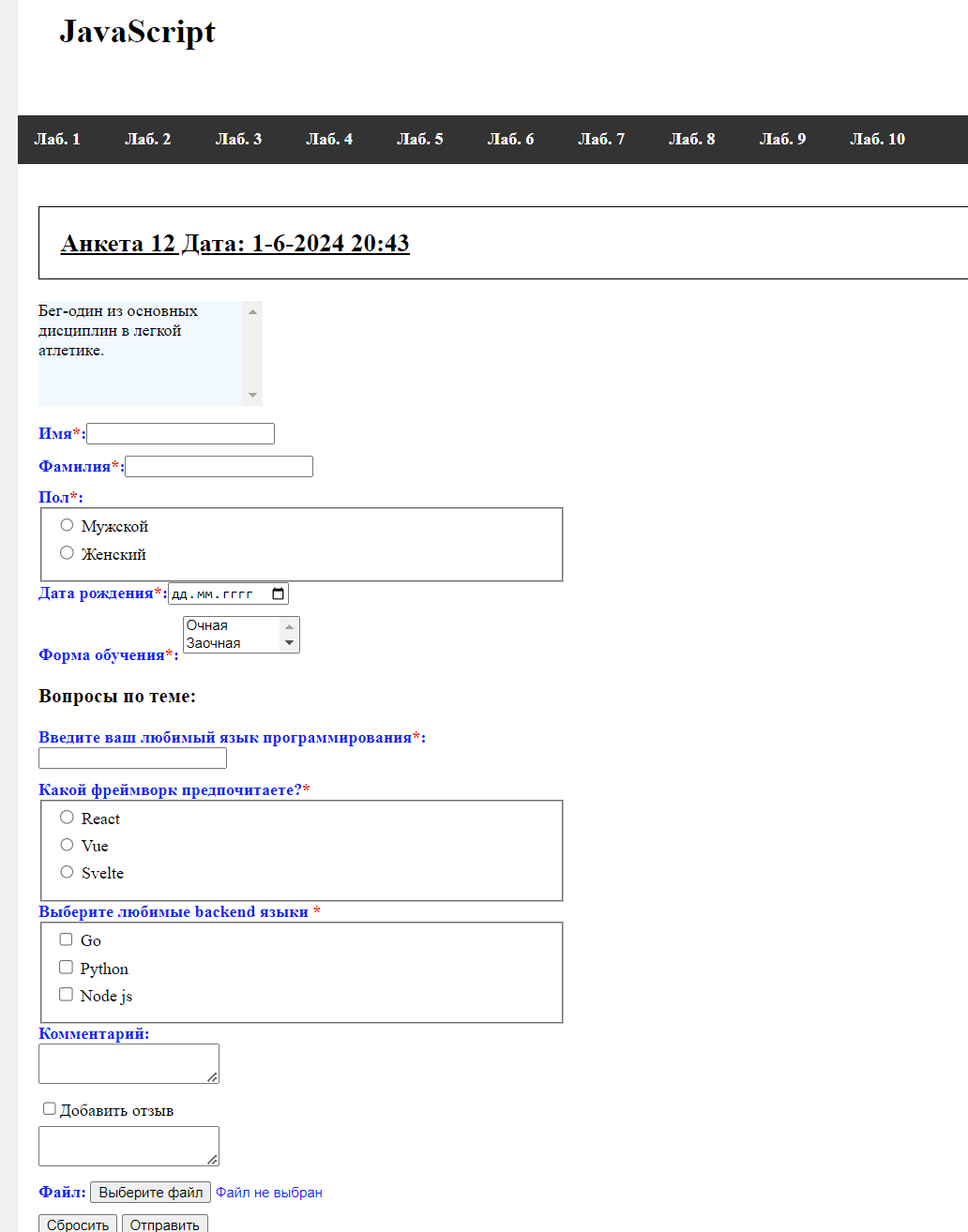


Рисунок 7. Страница формы

Восьмая группа шагов предполагала взаимодействие с серверами. Для этого используется объект типа XMLHttpRequest. Сначала запрос открывается с помощью метода open, параметрами которого являются метод запроса и адрес, затем устанавливается обработчик события onload для запроса, после чего отправляется с помощью метода send. Многие запросы возвращают JSON, потому его необходимо преобразовать в JavaScript объект с помощью метода JSON.parse. Для вывода результата производилась манипуляция DOM страницы для данной группы шагов, создавались новые элементы, которые добавлялись в div контейнер на этой странице.

Применяя полученные знания, была создана страница, которая после выполнения всех запросов принимает вид, показанный на рисунке 8.

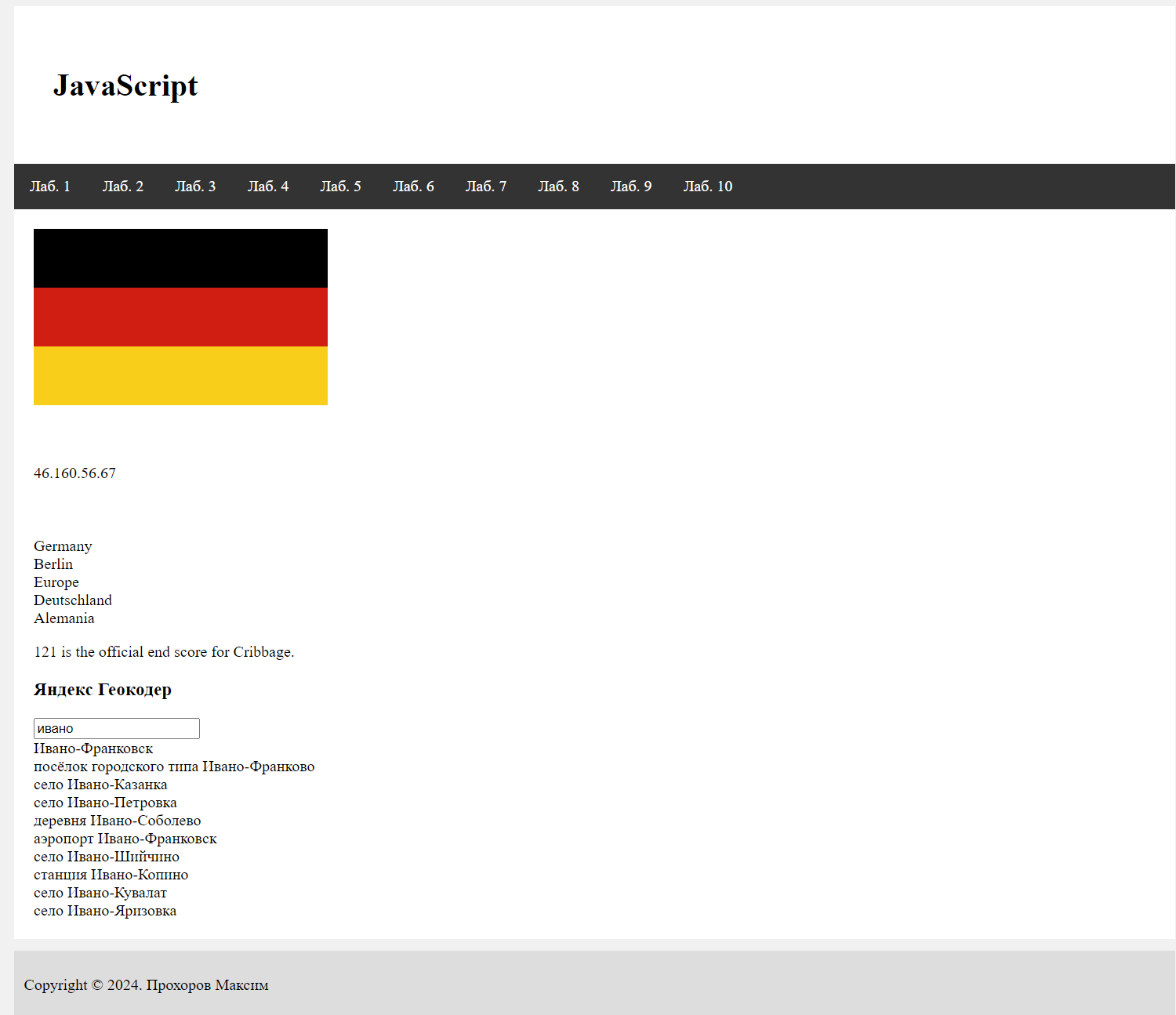


Рисунок 8. Страница после выполнения всех запросов

Для выполнения девятой группы шагов, необходимо было скачать библиотеку jQuery и подключить её.

jQuery предполагает обращение к элементам с помощью $(), внутри которого передаётся строка, обозначающая название элемента (или его идентификатор, если входная строка начинается с #.

Использовались следующие методы элементов из библиотеки jQuery:

* val() – значение внутри элемента ввода
* prepend() – добавить в начало элемента
* appendTo() – добавить в конец элемента
* clone() – клонировать элемент
* prop() – установление атрибута элемента
* remove() – удаление элемента
* show() – показать элемент
* filter() – фильтрация массива элементов
* text() – текст внутри элемента
* indexOf() – индекс первого вхождение подстроки в строке
* parent() – родительский элемент
* hide() – скрыть элемент
* on() – установка обработчика события
* animate() – анимирование объекта.

Применяя полученные знания, получим страницу, показанную на рисунке 9

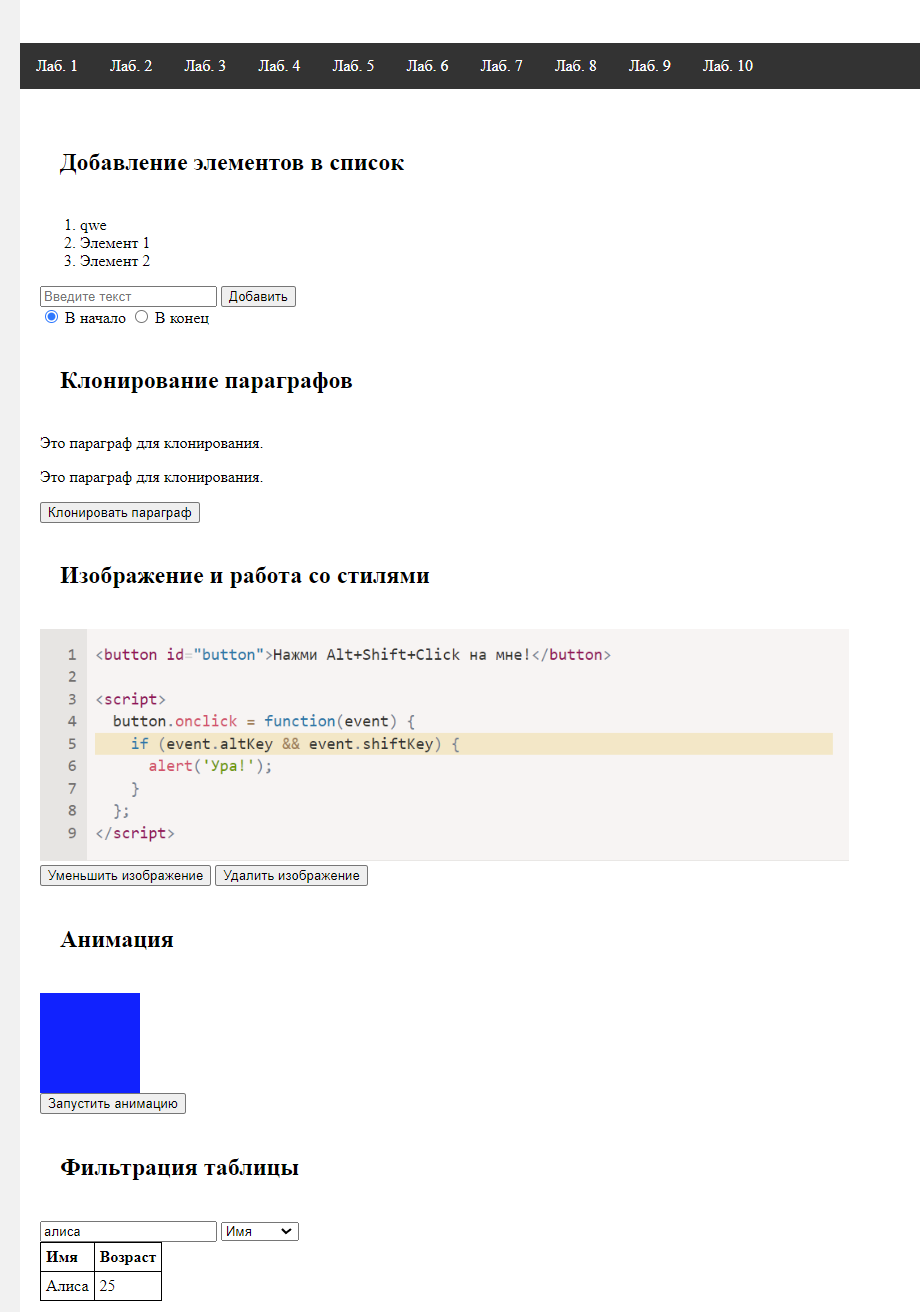
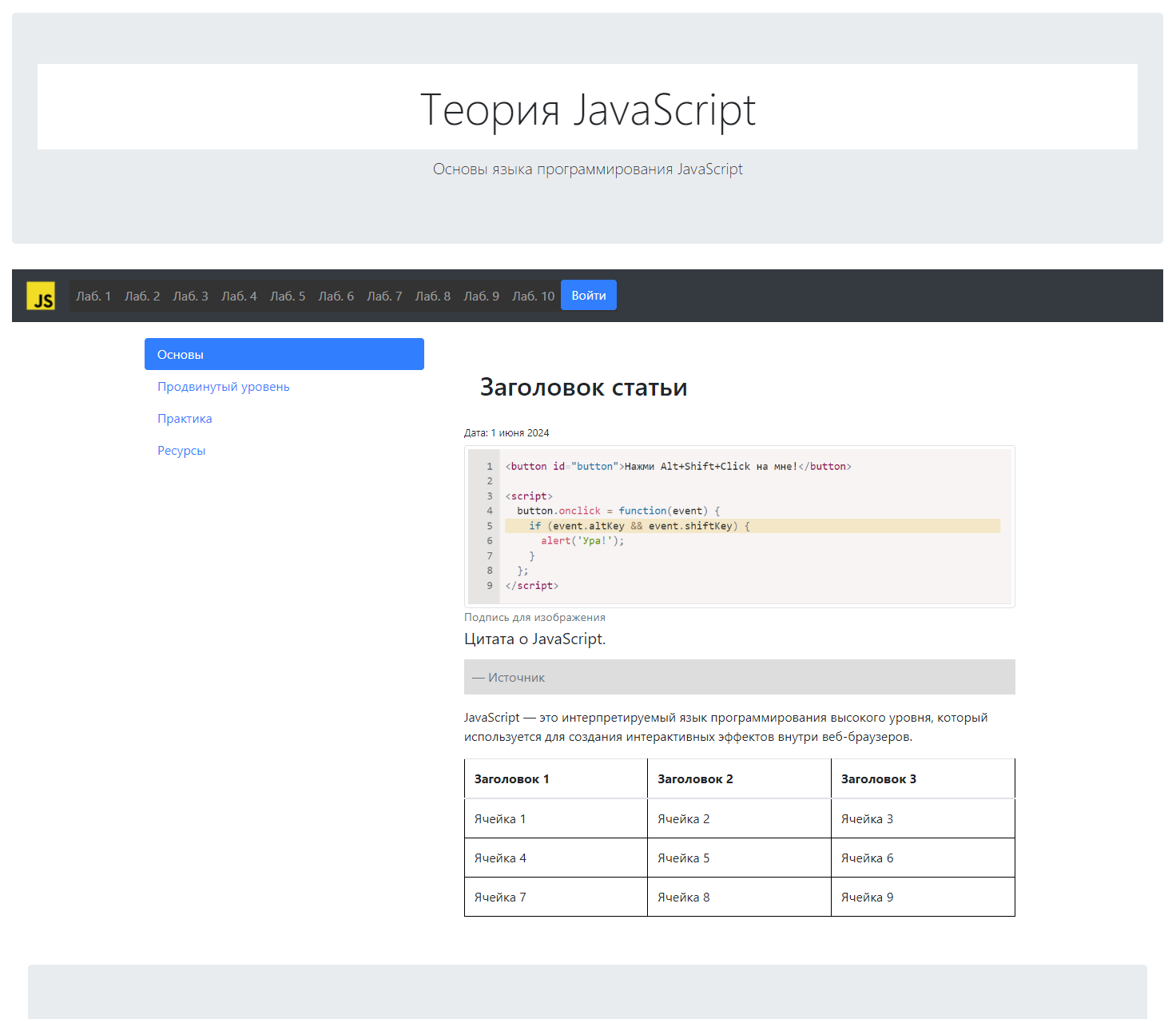


Рисунок 9. Страница примера работы с jQuery

Для выполнения десятой, последней группы шагов надо было скачать библиотеку Bootstrap 4, которую потом надо подключить к новой странице, после чего сверстать страницу с использованием библиотеки.

По сути, использование библиотеки предполагает использование классов Bootstrap, которые присваиваются элементам страницы через атрибут class. Ориентир был на официальный пример использования Bootstrap (<https://www.w3schools.com/bootstrap4/bootstrap_templates.asp>).

Ориентируясь на него, была создана страница, показанная на рисунке 10.

Рисунок 10. Страница с использованием Bootstrap 4

# Выводы по выполнению работы

В ходе выполнения данной самостоятельной работы были изучены:

* Методы создания и верстки HTML страниц
* Методы работы с каскадными таблицами стилей
* Методы написания скриптов на JavaScript для HTML-страниц, а также их подключения на сайт.
* Способы манипулирования DOM
* Способы обращения к серверу за данными с помощью XMLHttpRequest в JavaScript
* Технологии вёрстки HTML-формы, получение и обработка данных из неё
* Методы использования библиотеки jQuery, также её некоторый функционал.
* Классы библиотеки Bootstrap 4, а также способы их использования при верстке страниц HTML