

Визуализация данных на веб-странице с использованием библиотеки Chart.js

Кокухина Софья Александровна

Приамурский государственный университет им.Шолом-Алейхема

Студент

Аннотация:

Цель статьи – создание веб-страницы, которая визуализирует табличные данные с использованием библиотеки Chart.js. В работе демонстрируется процесс подключения Chart.js, подготовки входных данных и построения различных видов графиков (столбчатого, линейного и круговой диаграммы). Проект показывает, как современные библиотеки JavaScript упрощают реализацию интерактивных визуализаций в браузере.

Ключевые слова: Веб-разработка, JavaScript, Chart.js, визуализация данных, диаграммы.

Visualizing data on a web page using a library Chart.js

Kokukhina Sofya Alexandrovna

Sholom-Aleichem Priamursky State University

Student

Abstract

The purpose of the article is to create a web page that visualizes tabular data using the library Chart.js . The work demonstrates the process of connecting Chart.js, preparation of input data and construction of various types of graphs (columnar, linear and pie charts). The project shows how modern JavaScript libraries simplify the implementation of interactive visualizations in the browser.

Keywords: Web development, JavaScript, Chart.js, data visualization, diagrams.

1 Введение

1.1 Актуальность

Визуализация данных является важнейшим инструментом анализа и интерпретации информации. В современном веб-программировании библиотеки для построения графиков позволяют быстро и наглядно представить результаты исследований. Среди популярных решений можно выделить Chart.js, отличающуюся простотой подключения и широкими возможностями кастомизации.

1.2 Обзор исследований

В работе А.А. Халимова выполнено сопоставление инструментов клиентской визуализации: приводятся сильные и слабые стороны библиотек,

отдельно отмечается простота API Chart.js и её ограничения по скорости и глубине кастомизации; выводы релевантны учебным веб-графикам и небольшим информационным панелям [1]. Т.А. Ефимова рассматривает практики визуализации данных на сайтах и показывает применение D3.js как одной из популярных JS-библиотек; акцент делается на интерактивности и объясняемости графиков, что пересекается с целями нашей работы по наглядной подаче данных в браузере [2]. А.В. Мишуткин делает обзор графических библиотек, используемых в веб-приложениях, и обсуждает актуальность клиентской визуализации для современных интерфейсов; материалы полезны для обоснования выбора лёгких библиотек уровня Chart.js в учебных проектах [3]. А.С. Ломайкин анализирует роль фреймворков и библиотек в современной веб-разработке: на фоне React/SPA подчёркивается значение специализированных средств визуализации (D3.js, Canvas/WebGL-подходы) для построения интерактивной графики в браузере — тезисы поддерживают наш выбор client-side визуализации [4]. А.П. Кормилицына обсуждает визуализацию в медиасреде и цифровых коммуникациях; в контексте веб-страниц визуальные формы (диаграммы, инфографика) рассматриваются как инструмент повышения понятности данных для широкой аудитории — аргументация уместна для раздела «Актуальность» нашей статьи [5].

1.3 Цель исследования

Цель исследования – демонстрация процесса построения интерактивных диаграмм на веб-странице с использованием Chart.js и HTML5, а также анализ преимуществ такого подхода в образовательных и прикладных задачах.

2 Материалы и методы

В качестве основной среды разработки используется язык разметки HTML5, каскадные таблицы стилей CSS3 и язык программирования JavaScript. Для построения диаграмм применяется библиотека Chart.js, которая подключается через CDN и не требует установки дополнительных пакетов.

Работа выполняется в редакторе кода Visual Studio Code, обеспечивающем удобство написания HTML, CSS и JavaScript и возможность запуска проекта в любом современном браузере (например, Google Chrome).

2.1 Подготовка

На первом этапе создаётся папка проекта `chartjs_demo`. Внутри неё размещаются два файла:

`index.html` — основной HTML-документ, содержащий структуру страницы и подключение библиотеки Chart.js;

`script.js` — JavaScript-файл, отвечающий за подготовку данных и построение диаграмм.

Такое разделение кода делает проект удобным для расширения и поддерживает воспроизводимость эксперимента (Рис.1).

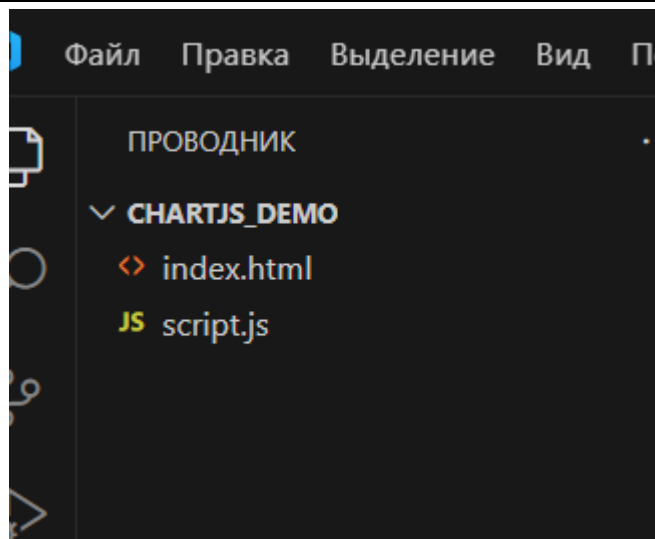


Рисунок 1 – Создание папки и файлов проекта

3 Результаты

3.1 Создание разметки страницы и подключение Chart.js

На первом этапе создаётся HTML-документ с тремя контейнерами `<canvas>` под разные типы диаграмм. Библиотека Chart.js подключается через CDN. Добавляется минимальная стилизация и простой переключатель набора данных (Рис. 2).

```

<? index.html > <? html > <? head > <? title
1  <!DOCTYPE html>
2  <html lang="ru">
3  <head>
4    <meta charset="UTF-8" />
5    <title>Визуализация данных с Chart.js</title>
6    <!-- Подключение Chart.js (v4) -->
7    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
8    <style>
9      :root { --card-width: 900px; }
10     body { margin: 0; font-family: Arial, sans-serif; background: #f4f6f8; }
11     .container { max-width: var(--card-width); margin: 24px auto; background: #fff;
12       border-radius: 12px; padding: 24px; box-shadow: 0 6px 24px rgba(0,0,0,.08); }
13     h1 { margin: 0 0 12px; font-size: 22px; }
14     .toolbar { display: flex; gap: 12px; align-items: center; margin: 8px 0 20px; }
15     .charts { display: grid; grid-template-columns: 1fr; gap: 24px; }
16     .card { background: #fafafa; border: 1px solid #e5e7eb; border-radius: 10px; padding: 16px; }
17     .card h2 { margin: 0 0 8px; font-size: 18px; }
18     /* Фиксированная высота для аккуратной отрисовки */
19     .chart-wrap { position: relative; height: 320px; }
20     @media (min-width: 1024px) {
21       .charts { grid-template-columns: 1fr 1fr; }
22       .charts .card:nth-child(3) { grid-column: 1 / -1; }
23     }
24     select { padding: 6px 10px; border-radius: 8px; border: 1px solid #cbd5e1; background: #fff; }
25   </style>
26 </head>
27 <body>
28   <div class="container">
29     <h1>Пример визуализации с Chart.js</h1>
30
31     <div class="toolbar">
32       <label for="datasetSelect">Набор данных:</label>
33       <select id="datasetSelect">
34         <option value="grades">Успеваемость (%)</option>
35         <option value="attendance">Посещаемость (%)</option>
36       </select>
37     </div>
38
  
```

Рисунок 2 – Главная страница проекта

Создаются три «карточки» с холстами Canvas — под столбчатую, линейную и круговую диаграммы. В панели инструментов (.toolbar) размещается выпадающий список для переключения между двумя учебными наборами данных.

3.2 Подготовка данных и базовых настроек

Сначала задаются категории (подписи оси X и сегментов диаграмм). const labels = ["Математика", "Физика", "Информатика", "Химия", "Биология"]; массив labels используется всеми диаграммами одновременно, чтобы обеспечить сопоставимость графиков. Далее формируются два учебных набора для демонстрации переключения: «успеваемость» и «посещаемость», и для наглядности задаются базовые цвета для каждого типа диаграммы (Рис.3).

```
1 // 1) Категории (оси X / метки сегментов)
2 const labels = ["Математика", "Физика", "Информатика", "Химия", "Биология"];
3
4 // 2) Два учебных набора данных для демонстрации обновления графиков
5 const datasetsPool = {
6   grades: { label: "Успеваемость (%)", values: [85, 90, 78, 70, 95] },
7   attendance: { label: "Посещаемость (%)", values: [92, 88, 80, 75, 90] }
8 };
9
10 // 3) Палитры (для наглядности задаются явно)
11 const barColors = "rgba(75, 192, 192, 0.6)";
12 const lineColor = "rgba(255, 99, 132, 1)";
13 const pieColors = ["#FF6384", "#36A2EB", "#FFCE56", "#4BC0C0", "#9966FF"];
14
15 // Ссылки на элементы DOM
```

Рисунок 3 – Подготовка данных и палитры цветов

Далее необходимо сделать небольшую функцию, которая по ключу (grades/attendance) возвращает объект data в формате Chart.js (Рис.4).

```
function buildData(poolKey) {
  const ds = datasetsPool[poolKey];
  return {
    labels,
    datasets: [{
      label: ds.label,
      data: ds.values
    }]
  };
}
```

Рисунок 4 – Функция buildData

Создается столбчатая диаграмма с базовыми опциями, подходящими для учебных значений (Рис.5).

```
35  ∨ const barChart = new Chart(barCtx, {
36      type: "bar",
37      data: buildData(select.value),
38  ∨  options: {
39      responsive: true,
40      maintainAspectRatio: false,
41  ∨  plugins: {
42      legend: { display: true },
43      title: { display: true, text: "Столбчатая диаграмма" },
44      tooltip: { enabled: true }
45  },
46  ∨  scales: {
47      y: { beginAtZero: true, ticks: { stepSize: 10 } },
48      x: { ticks: { autoSkip: false } }
49  }
50  }
51  });
52  // Цвета для столбиков (один цвет на датасет)
53  barChart.data.datasets[0].backgroundColor = barColors;
54  barChart.update();
55
```

Рисунок 5 – Столбчатая диаграмма

Далее идет написание линейного графика с настройками подсказок и параметрами рисования линии (Рис.6).

```

57 const lineChart = new Chart(lineCtx, {
58   type: "line",
59   data: buildData(select.value),
60   options: {
61     responsive: true,
62     maintainAspectRatio: false,
63     plugins: {
64       legend: { display: true },
65       title: { display: true, text: "линейный график" },
66       tooltip: { intersect: false, mode: "index" }
67     },
68     elements: { point: { radius: 4 } },
69     scales: {
70       y: { beginAtZero: true, ticks: { stepSize: 10 } },
71       x: { ticks: { autoSkip: false } }
72     }
73   }
74 });
75 // Оформление линии
76 lineChart.data.datasets[0].borderColor = lineColor;
77 lineChart.data.datasets[0].backgroundColor = "rgba(255, 99, 132, 0.1)";
78 lineChart.data.datasets[0].fill = false;
79 lineChart.data.datasets[0].tension = 0.2;
80 lineChart.update();
81

```

Рисунок 6 – Линейный график

Также нужно написать код круговой диаграммы с отдельной палитрой сегментов и легендой внизу (Рис.7).

```

const pieChart = new Chart(pieCtx, {
  type: "pie",
  data: {
    labels,
    datasets: [{
      label: datasetsPool[select.value].label,
      data: datasetsPool[select.value].values,
      backgroundColor: pieColors
    }]
  },
  options: {
    responsive: true,
    maintainAspectRatio: false,
    plugins: {
      legend: { position: "bottom" },
      title: { display: true, text: "Круговая диаграмма" },
      tooltip: { enabled: true }
    }
  }
});

```

Рисунок 7 – Круговая диаграмма

Добавляется реакция на изменение выпадающего списка: при выборе «Посещаемость» или «Успеваемость» все три диаграммы обновляются согласованно (Рис.8).

```
function updateAllCharts(poolKey) {  
  // Обновление столбчатой  
  barChart.data = buildData(poolKey);  
  barChart.data.datasets[0].backgroundColor = barColors;  
  barChart.update();  
  
  // Обновление линейной  
  lineChart.data = buildData(poolKey);  
  lineChart.data.datasets[0].borderColor = lineColor;  
  lineChart.data.datasets[0].backgroundColor = "rgba(255, 99, 132, 0.1)";  
  lineChart.update();  
  
  // Обновление круговой  
  pieChart.data.labels = labels;  
  pieChart.data.datasets[0].label = datasetsPool[poolKey].label;  
  pieChart.data.datasets[0].data = datasetsPool[poolKey].values;  
  pieChart.data.datasets[0].backgroundColor = pieColors;  
  pieChart.update();  
}  
  
// Слушатель выбора набора данных  
select.addEventListener("change", (e) => {  
  updateAllCharts(e.target.value);  
});
```

Рисунок 8 – Обновление списков

После завершения написания кода проект открывается в браузере. Для этого достаточно дважды щёлкнуть по файлу index.html или запустить его через встроенный сервер редактора (например, Live Server в Visual Studio Code).

На экране отображается веб-страница с заголовком и тремя диаграммами: столбчатой, линейной и круговой. Все диаграммы построены на основании массива категорий и выбранного набора данных (Рис.9).

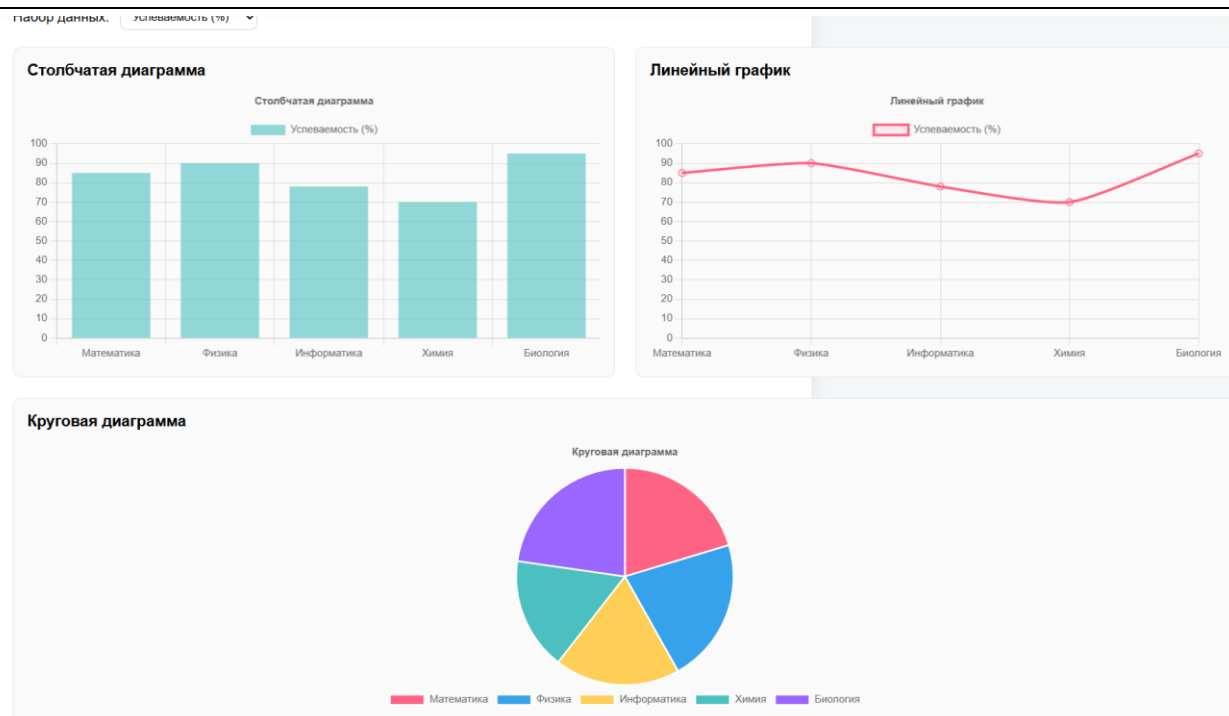


Рисунок 9 – Отображение диаграмм в браузере

В верхней части страницы располагается выпадающий список для переключения набора данных. При изменении значения в списке (например, с «Успеваемость (%)» на «Посещаемость (%)») все три диаграммы мгновенно обновляются, демонстрируя новую выборку (Рис.10).

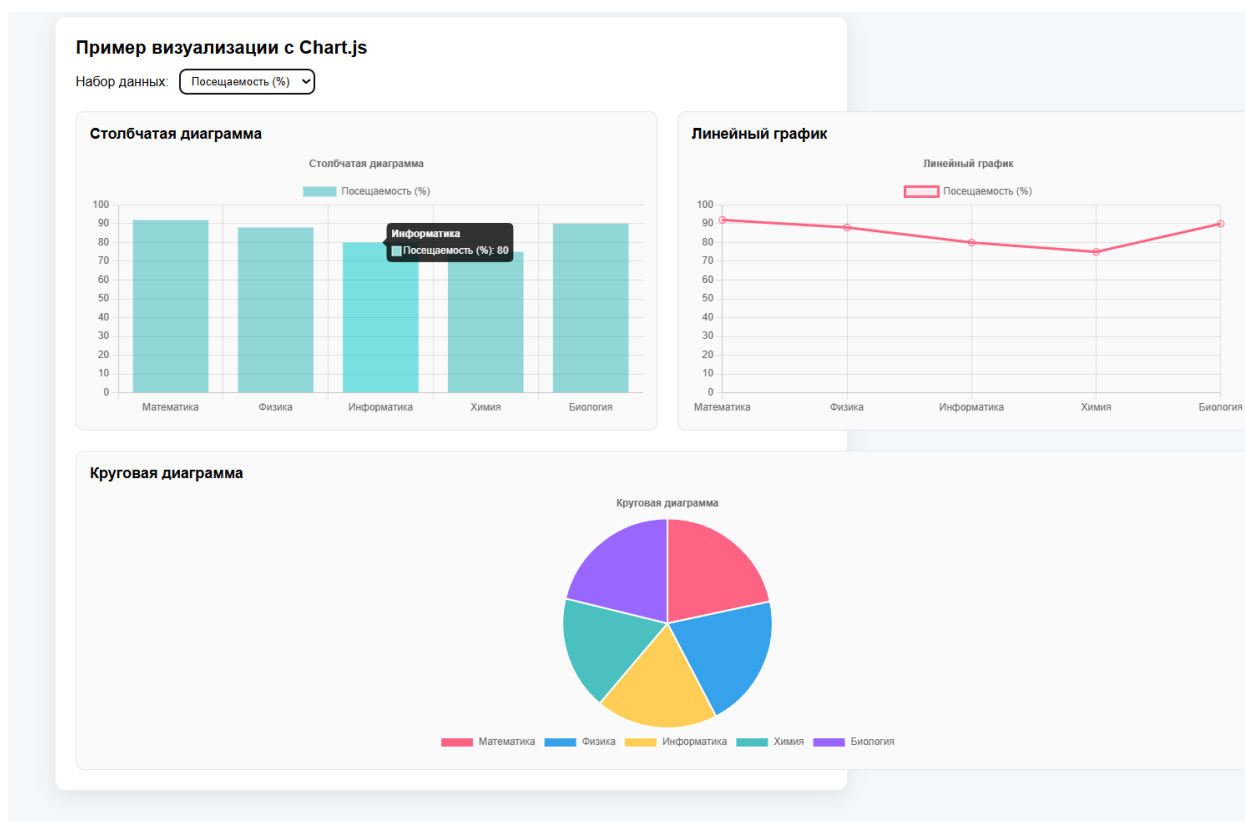


Рисунок 10 – Состояние веб-страницы после переключения набора данных

Кроме того, при наведении курсора на элементы графиков появляются интерактивные подсказки (tooltips), содержащие точные числовые значения. Это позволяет пользователю получать более детальную информацию о каждом элементе визуализации (Рис.11).

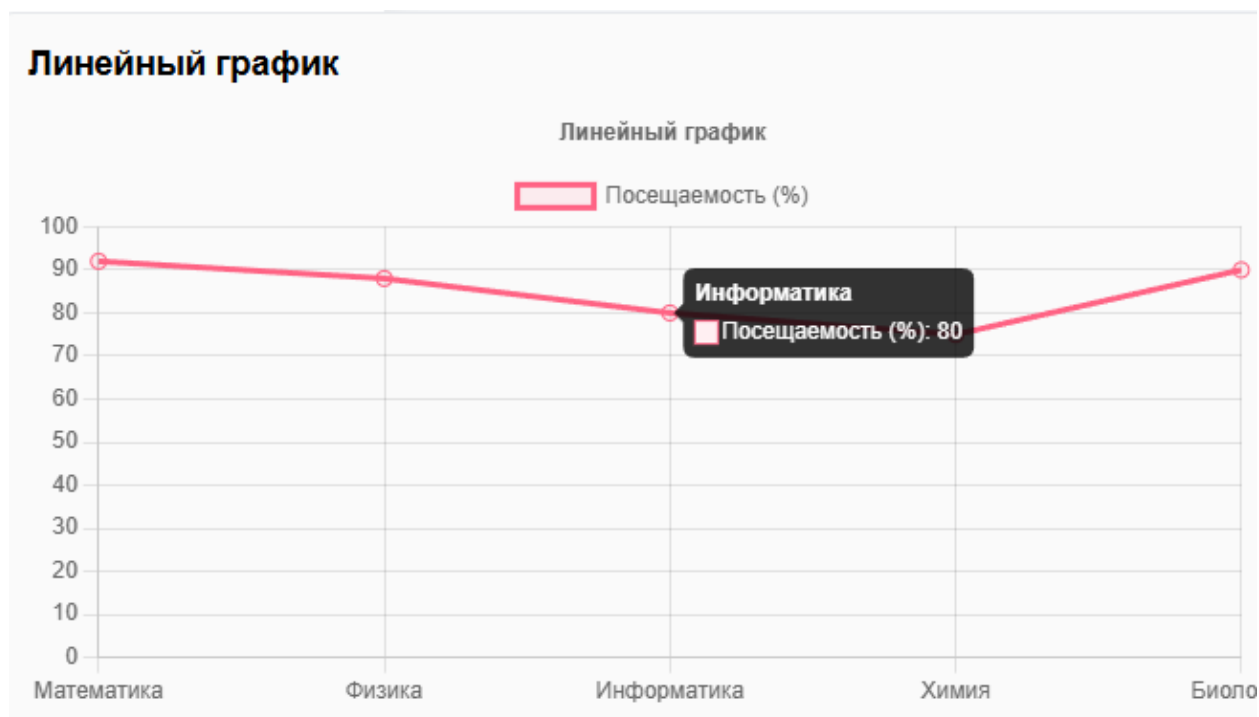


Рисунок 11 – Пример всплывающей подсказки над точкой линейного графика

4

Выводы. Реализация веб-страницы для визуализации данных с помощью библиотеки Chart.js показала, что данная библиотека предоставляет простой и наглядный API для построения диаграмм прямо в браузере. Использование контейнеров <canvas> и конструктора new Chart() делает код компактным и легко расширяемым, а параметризация (типы графиков, цвета, подписи) обеспечивает переиспользуемость. Добавление выпадающего списка с переключением наборов данных демонстрирует базовую интерактивность без сложных фреймворков. Встроенные подсказки (tooltips) повышают информативность графиков.

Перспективы дальнейшей работы: подключение внешних источников данных (CSV или API), использование дополнительных типов диаграмм (radar, doughnut), а также интеграция таких визуализаций в образовательные и аналитические веб-приложения.

Библиографический список

1. Халимов А. А., Крехов Д. С. Сравнение инструментов для отображения графиков параметров автоматических систем управления на клиентской стороне // Тинчуринские чтения – 2022 «Энергетика и цифровая

- трансформация. 2022 г. / под общ. ред. Э. Ю. Абдуллазянова. Т. 2. Казань : Казан. гос. энергет. ун-т, 2022. С. 466–468. URL: <https://www.elibrary.ru/item.asp?id=49190519> (дата обращения: 21.09.2025).
2. Ефимова Т. А. Визуализация данных на сайтах интернета: общие понятия и технология представления табличных данных в виде интерактивного дерева // Студенческий электронный журнал СтРИЖ. 2019. № 4-2(27). С. 12–17. URL: <https://www.elibrary.ru/item.asp?id=38547972> (дата обращения: 21.09.2025).
 3. Мишуткин А. В., Степачева А. В., Добуш И. М. Обзор графических библиотек для визуализации характеристик радиоэлектронных компонентов и устройств в веб-приложениях // Электронные средства и системы управления : материалы докл. Междунар. науч.-практ. конф. 2016. № 1-2. С. 140–143. URL: <https://www.elibrary.ru/item.asp?id=28831340> (дата обращения: 21.09.2025).
 4. Ломайкин А. С. Роль фреймворков и библиотек в современной веб-разработке // Материалы XXIV науч.-практ. конф. молодых ученых, аспирантов и студентов Национального исследовательского Мордовского гос. ун-та им. Н. П. Огарёва : в 3 ч. Сост. А. В. Столяров ; отв. за вып. П. В. Сенин. Ч. 2. Саранск : Нац. исслед. Мордов. гос. ун-т им. Н. П. Огарёва, 2021. С. 268–273. URL: <https://www.elibrary.ru/item.asp?id=47367186> (дата обращения: 21.09.2025).
 5. Кормилицына А. П. Визуализация данных в современном информационном пространстве // Творчество молодых: искусство, дизайн, медиатехнологии. Омск : Омский гос. техн. ун-т, 2021. С. 29–33. URL: <https://www.elibrary.ru/item.asp?id=46581216> (дата обращения: 21.09.2025).