ГУАП КАФЕДРА № 43

ОТЧЕТ

ЗАЩИЩЕН С ОЦЕНКОЙ ПРЕПОДАВАТЕЛЬ

|  |  |  |
| --- | --- | --- |
| ассистент |  | М.А. Мурашова |
| должность, уч. степень, звание | подпись, дата | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №7 |
| АСИНХРОННОЕ КЛИЕНТ-СЕРВЕРНОЕ ВЗАИМОДЕЙСТВИЕ С ИСПОЛЬЗОВАНИЕМ AJAX И JSON |
| по дисциплине: WEB-ТЕХНОЛОГИИ |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |
| --- | --- | --- |
| СТУДЕНТ ГР. | 4232 | А.В. Матвеев |
|  | подпись, дата | инициалы, фамилия |

Санкт-Петербург

2025

**Содержание**

[1. Цель работы 3](#_Toc192092890)

[2. Задание 3](#_Toc192092891)

[3. Вариант задания 3](#_Toc192092892)

[4. Названия средств, использованных при выполнении работы 3](#_Toc192092893)

[5. Выполнение задания 3](#_Toc192092894)

[6. Листинг программного кода 6](#_Toc192092895)

[7. Выводы 10](#_Toc192092896)

# 1. Цель работы

Изучение и применение на практике технологии AJAX.

# 2. Задание

**Базовое задание**

Разработать AJAX-приложение для доступа к базе данных, подготовленной в предыдущей лабораторной работе. Вариант задания и тип запроса указаны в таблице 1. Формулировку варианта адаптировать под свою базу данных. В сценарии javascript запрещается использование библиотек. Новая страница должна быть встроена в сайт, подготовленный в предыдущих разделах.

**Расширенное задание**

1. Использовать json.

2. Проверить работу в разных браузерах. В сценарии на javascript добавить вывод на страницу названия объекта, выполняющего асинхронный обмен, и название браузера. В отчете привести доказательства этой работы.

3. Использовать библиотеку jQuery и добавить еще один вариант применения AJAX.

# 3. Вариант задания

|  |  |  |
| --- | --- | --- |
| № варианта | Тип запроса | Гр 4232 |
| 6 | POST | Матвеев Андрей Владимирович |

Тема сайта: История компании Nintendo

# 4. Названия средств, использованных при выполнении работы

1. Редактор кода Visual Studio Code.

2. Браузер Mozilla Firefox.

3. Браузер Google Chrome.

# 5. Выполнение задания

На рисунках 1-2 представлена страница ajax\_database.php в браузерах Mozilla Firefox и Google Chrome.

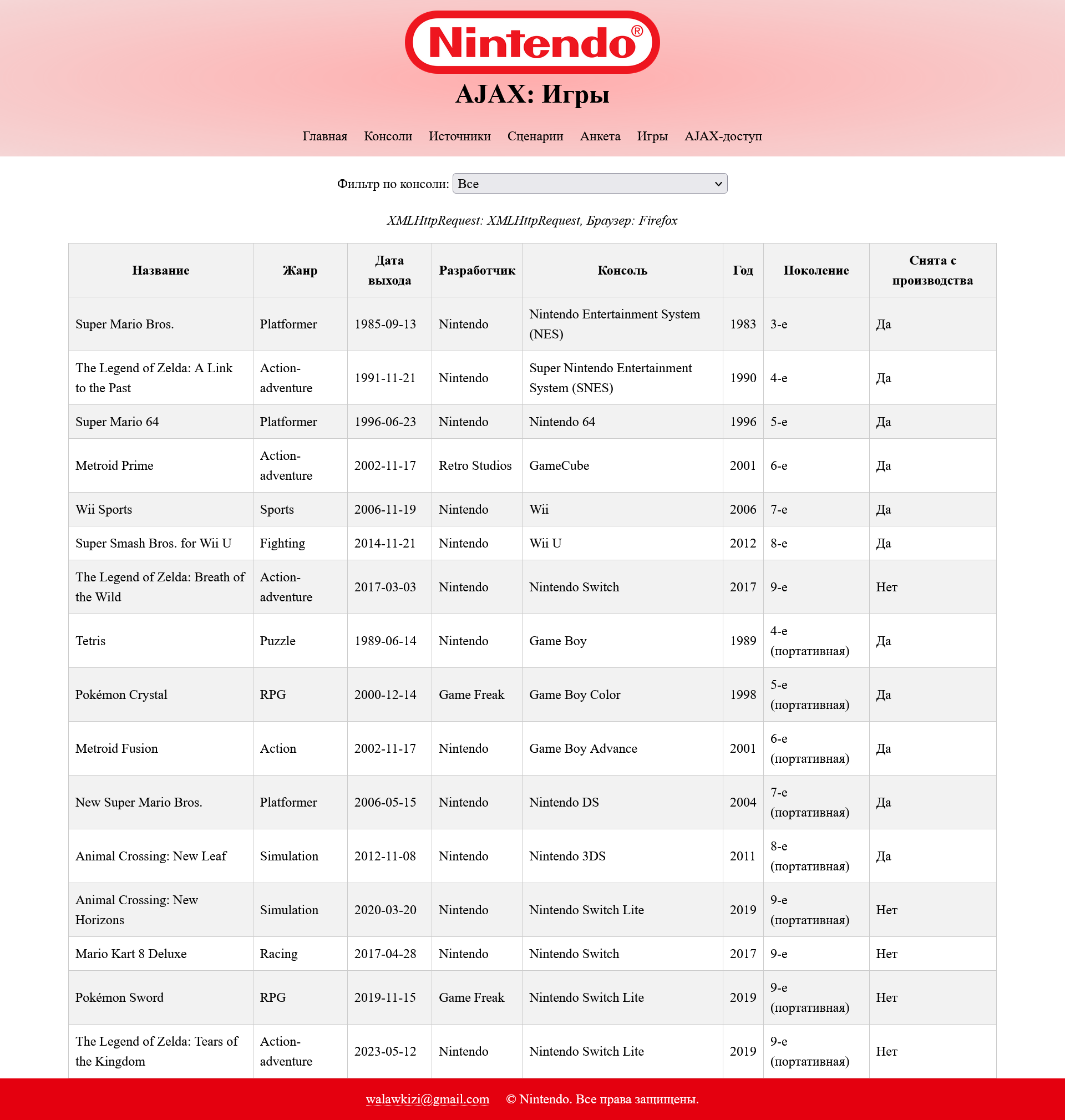


Рисунок 1 – Исходная страница в браузере Mozilla Firefox

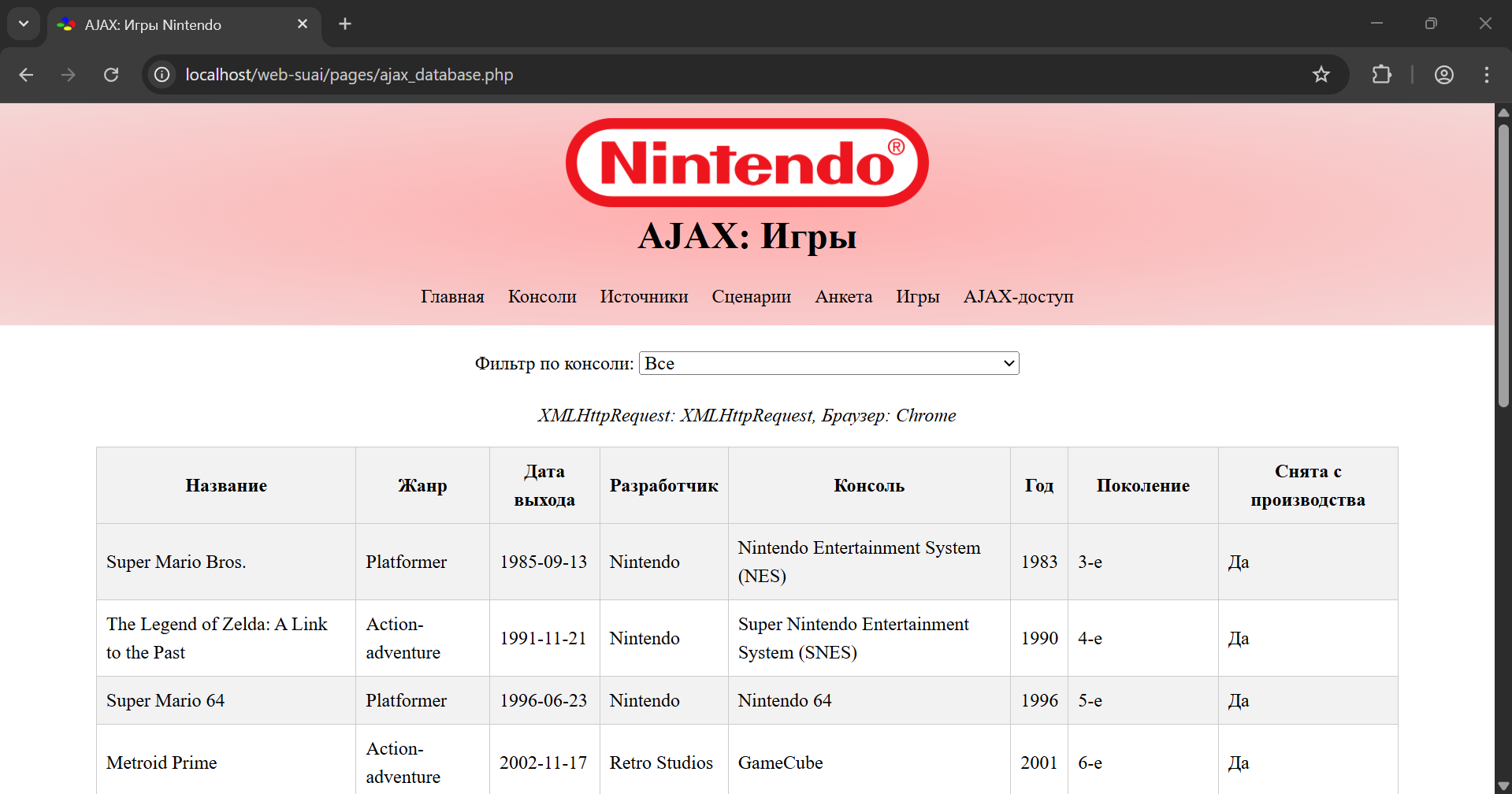


Рисунок 2 – Исходная страница в браузере Google Chrome

На рисунке 3 представлен выпадающий список с выбором консоли.

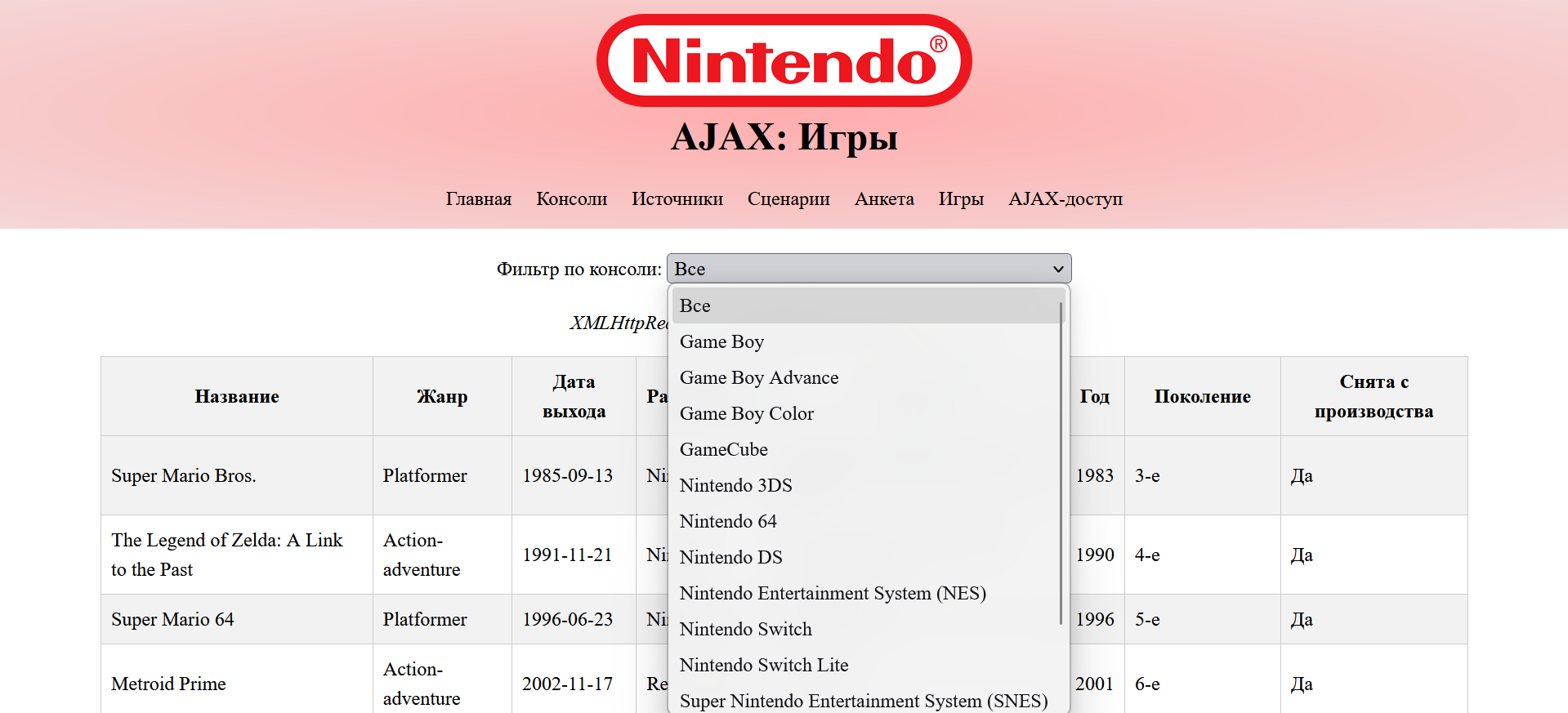


Рисунок 3 – Выбор консоли

На рисунке 4 представлены данные, выводящиеся на страницу асинхронно (без обновления страницы) при выборе конкретной консоли.

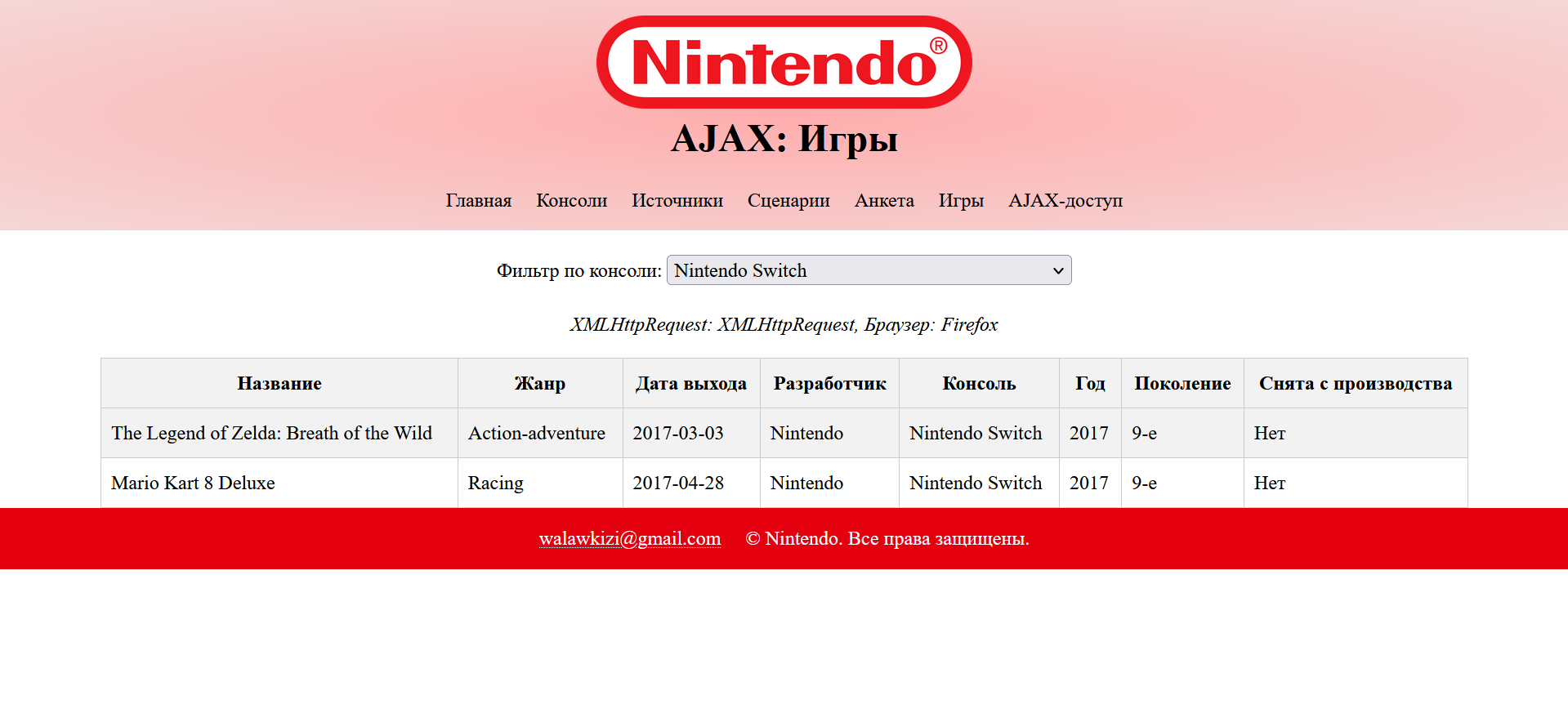


Рисунок 4 – Строки консоли «Nintendo Switch»

# 6. Листинг программного кода

**get\_data.php для получения json-ответа**

*<?php*

header(*'Content-Type: application/json; charset=utf-8'*);

$mysqli = **new** mysqli(*"localhost"*, *"root"*, *""*, *"nintendo"*);

**if** ($mysqli->connect\_errno) {

**echo** json\_encode([*'error'* => *'Ошибка подключения к БД: '* . $mysqli->connect\_error]);

**exit**();

}

$mysqli->set\_charset(*"utf8"*);

$console\_name = $\_POST[*'console'*] ?? *''*;

*// Запрос с фильтром по консоли, если задана*

**if** ($console\_name) {

$stmt = $mysqli->prepare(*"*

*SELECT g.title, g.genre, g.release\_date, g.developer,*

*c.name AS console\_name, c.release\_year, c.generation, c.discontinued*

*FROM games g*

*JOIN consoles c ON g.console\_id = c.id*

*WHERE c.name = ?*

*"*);

$stmt->bind\_param(*"s"*, $console\_name);

$stmt->execute();

$result = $stmt->get\_result();

} **else** {

$result = $mysqli->query(*"*

*SELECT g.title, g.genre, g.release\_date, g.developer,*

*c.name AS console\_name, c.release\_year, c.generation, c.discontinued*

*FROM games g*

*JOIN consoles c ON g.console\_id = c.id*

*"*);

}

$data = [];

**while** ($row = $result->fetch\_assoc()) {

$row[*'discontinued'*] = $row[*'discontinued'*] ? *'Да'* : *'Нет'*;

$data[] = $row;

}

**echo** json\_encode($data);

*?>*

**ajax\_database.php:**

<!DOCTYPE html>

<html lang="ru">

<head>

<meta charset="UTF-8">

<title>AJAX: Игры Nintendo</title>

<link rel="icon" type="image/x-icon" href="../images/favicon.ico">

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

<style>

table { width: 90%; margin: auto; border-collapse: collapse; }

th, td { border: 1px solid #ccc; padding: 8px; }

th { background-color: #f2f2f2; }

.filter-container { text-align: center; margin: 20px; }

#browserInfo { text-align: center; margin-bottom: 15px; font-style: italic; }

</style>

</head>

<body>

<header class="header">

<div class="header\_\_inner container">

<div class="header\_\_logo-title">

<img class="header\_\_logo" src="../images/header.png" alt="Шапка сайта">

<h1 class="header\_\_title">AJAX: Игры</h1>

</div>

<nav class="header\_\_menu">

<ul class="header\_\_menu-list">

<li class="header\_\_menu-item">

<a class="header\_\_menu-link" href="../index.html">Главная</a>

</li>

<li class="header\_\_menu-item">

<a class="header\_\_menu-link" href="../pages/consoles.html">Консоли</a>

</li>

<li class="header\_\_menu-item">

<a class="header\_\_menu-link" href="../pages/sources.html">Источники</a>

</li>

<li class="header\_\_menu-item">

<a class="header\_\_menu-link" href="../pages/scenarios.html">Сценарии</a>

</li>

<li class="header\_\_menu-item">

<a class="header\_\_menu-link" href="../pages/form.html">Анкета</a>

</li>

<li class="header\_\_menu-item"><a href="../pages/games.php" class="header\_\_menu-link">Игры</a></li>

<li class="header\_\_menu-item"><a href="../pages/ajax\_database.php" class="header\_\_menu-link">AJAX-доступ</a></li>

</ul>

</nav>

</div>

</header>

<main class="main container">

<div class="filter-container">

<label>Фильтр по консоли:

<select id="consoleSelect" onchange="loadData()">

<option value="">Все</option>

*<?php*

$conn = **new** mysqli(*"localhost"*, *"root"*, *""*, *"nintendo"*);

$conn->set\_charset(*"utf8"*);

$res = $conn->query(*"SELECT DISTINCT name FROM consoles ORDER BY name"*);

**while** ($row = $res->fetch\_assoc()) {

**echo** *"<option value='"* . htmlspecialchars($row[*'name'*]) . *"'>"* . htmlspecialchars($row[*'name'*]) . *"</option>"*;

}

$conn->close();

*?>*

</select>

</label>

</div>

<div id="browserInfo"></div>

<table>

<thead>

<tr>

<th>Название</th>

<th>Жанр</th>

<th>Дата выхода</th>

<th>Разработчик</th>

<th>Консоль</th>

<th>Год</th>

<th>Поколение</th>

<th>Снята с производства</th>

</tr>

</thead>

<tbody id="dataBody"></tbody>

</table>

</main>

<footer class="footer">

<address class="footer\_\_address">

<a href="mailto:walawkizi@gmail.com">walawkizi@gmail.com</a>

<p> © Nintendo. Все права защищены.</p>

</address>

</footer>

<script>

function loadData() {

let xhr = new XMLHttpRequest();

let console = document.getElementById("consoleSelect").value;

let params = "console=" + encodeURIComponent(console);

// Определим браузер

let ua = navigator.userAgent;

let browser = "Неизвестный";

if (ua.includes("Chrome")) browser = "Chrome";

else if (ua.includes("Firefox")) browser = "Firefox";

else if (ua.includes("Safari")) browser = "Safari";

else if (ua.includes("Edg")) browser = "Edge";

document.getElementById("browserInfo").innerText =

"XMLHttpRequest: " + xhr.constructor.name + ", Браузер: " + browser;

xhr.open("POST", "../get\_data.php", true);

xhr.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");

xhr.onload = function () {

let tbody = document.getElementById("dataBody");

tbody.innerHTML = "";

try {

let data = JSON.parse(this.responseText);

if (data.error) {

tbody.innerHTML = `<tr><td colspan="8">${data.error}</td></tr>`;

return;

}

data.forEach(row => {

let tr = document.createElement("tr");

tr.innerHTML = `

<td>${row.title}</td>

<td>${row.genre}</td>

<td>${row.release\_date}</td>

<td>${row.developer}</td>

<td>${row.console\_name}</td>

<td>${row.release\_year}</td>

<td>${row.generation}</td>

<td>${row.discontinued}</td>

`;

tbody.appendChild(tr);

});

} catch (e) {

tbody.innerHTML = `<tr><td colspan="8">Ошибка загрузки данных</td></tr>`;

}

};

xhr.send(params);

}

window.onload = loadData;

</script>

</body>

</html>

# 7. Выводы

В ходе выполнения работы было разработано AJAX-приложение на странице ajax\_database.php, позволяющее асинхронно получать данные из базы данных с фильтрацией по консолям с использованием JavaScript и JSON. Проведено тестирование работы приложения в различных браузерах (Chrome, Firefox), подтверждено корректное определение браузера и объекта AJAX.