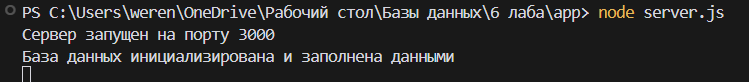
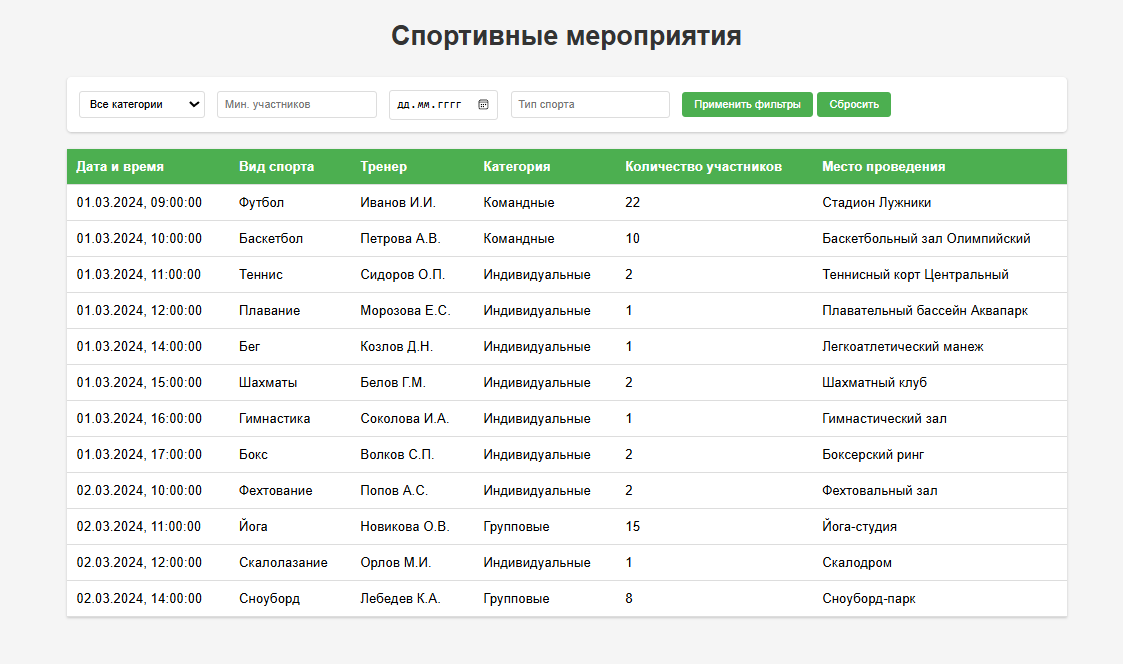
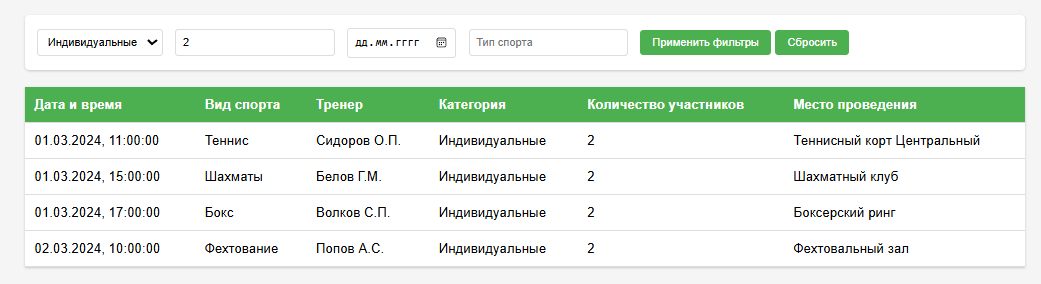
Был создан одностраничный сайт на который выводиться таблица из базы данных так же в таблице работают фильтры

1)Результат:





Фильтров (выбрана категория и минимальное количество участников )



2) код программы:  
index.html

<!DOCTYPE html>

<html lang="ru">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Спортивные мероприятия</title>

<style>

body {

font-family: Arial, sans-serif;

margin: 0;

padding: 20px;

background-color: #f5f5f5;

min-height: 100vh;

display: flex;

justify-content: center;

align-items: flex-start;

}

.container {

max-width: 1200px;

width: 90%;

margin: 20px auto;

}

h1 {

color: #333;

text-align: center;

margin-bottom: 30px;

}

table {

width: 100%;

border-collapse: collapse;

background-color: white;

box-shadow: 0 1px 3px rgba(0,0,0,0.2);

margin-bottom: 20px;

margin: 0 auto;

}

th, td {

padding: 12px;

text-align: left;

border-bottom: 1px solid #ddd;

}

th {

background-color: #4CAF50;

color: white;

}

tr:hover {

background-color: #f5f5f5;

}

.filters {

margin-bottom: 20px;

padding: 15px;

background-color: white;

border-radius: 5px;

box-shadow: 0 1px 3px rgba(0,0,0,0.2);

}

.filters select, .filters input {

padding: 8px;

margin-right: 10px;

border: 1px solid #ddd;

border-radius: 4px;

}

button {

padding: 8px 15px;

background-color: #4CAF50;

color: white;

border: none;

border-radius: 4px;

cursor: pointer;

}

button:hover {

background-color: #45a049;

}

</style>

</head>

<body>

<div class="container">

<h1>Спортивные мероприятия</h1>

<div class="filters">

<select id="categoryFilter">

<option value="">Все категории</option>

<option value="Командные">Командные</option>

<option value="Индивидуальные">Индивидуальные</option>

<option value="Групповые">Групповые</option>

</select>

<input type="number" id="minParticipants" placeholder="Мин. участников">

<input type="date" id="dateFilter">

<input type="text" id="sportType" placeholder="Тип спорта">

<button onclick="applyFilters()">Применить фильтры</button>

<button onclick="resetFilters()">Сбросить</button>

</div>

<table>

<thead>

<tr>

<th>Дата и время</th>

<th>Вид спорта</th>

<th>Тренер</th>

<th>Категория</th>

<th>Количество участников</th>

<th>Место проведения</th>

</tr>

</thead>

<tbody id="eventsTable">

</tbody>

</table>

</div>

<script>

function **formatDateTime**(*dateTimeStr*) {

const date = new Date(*dateTimeStr*);

return date.toLocaleString('ru-RU');

}

async function **loadEvents**(*filters* = {}) {

try {

let url = '/api/events';

if (Object.keys(*filters*).length > 0) {

const params = new URLSearchParams(*filters*);

url = `/api/events/filter?${params}`;

}

const response = await fetch(url);

const events = await response.json();

const tbody = document.getElementById('eventsTable');

tbody.innerHTML = '';

events.forEach(*event* => {

const row = document.createElement('tr');

row.innerHTML = `

<td>${formatDateTime(*event*.date\_time)}</td>

<td>${*event*.sport\_type}</td>

<td>${*event*.coach}</td>

<td>${*event*.category}</td>

<td>${*event*.participants\_count}</td>

<td>${*event*.location\_name}</td>

`;

tbody.appendChild(row);

});

} catch (error) {

console.error('Ошибка при загрузке данных:', error);

}

}

function **applyFilters**() {

const filters = {};

const category = document.getElementById('categoryFilter').value;

const minParticipants = document.getElementById('minParticipants').value;

const date = document.getElementById('dateFilter').value;

const sportType = document.getElementById('sportType').value;

if (category) filters.category = category;

if (minParticipants) filters.minParticipants = minParticipants;

if (date) filters.date = date;

if (sportType) filters.sport\_type = sportType;

loadEvents(filters);

}

function **resetFilters**() {

document.getElementById('categoryFilter').value = '';

document.getElementById('minParticipants').value = '';

document.getElementById('dateFilter').value = '';

document.getElementById('sportType').value = '';

loadEvents();

}

*// Загрузка данных при открытии страницы*

loadEvents();

</script>

</body>

</html>

data.js

const { db } = require('./db');

const locations = [

"Стадион Лужники",

"Баскетбольный зал Олимпийский",

"Теннисный корт Центральный",

"Плавательный бассейн Аквапарк",

"Легкоатлетический манеж",

"Шахматный клуб",

"Гимнастический зал",

"Боксерский ринг",

"Фехтовальный зал",

"Йога-студия",

"Скалодром",

"Сноуборд-парк"

];

const events = [

["2024-03-01 09:00", "Футбол", "Иванов И.И.", "Командные", 22, 1],

["2024-03-01 10:00", "Баскетбол", "Петрова А.В.", "Командные", 10, 2],

["2024-03-01 11:00", "Теннис", "Сидоров О.П.", "Индивидуальные", 2, 3],

["2024-03-01 12:00", "Плавание", "Морозова Е.С.", "Индивидуальные", 1, 4],

["2024-03-01 14:00", "Бег", "Козлов Д.Н.", "Индивидуальные", 1, 5],

["2024-03-01 15:00", "Шахматы", "Белов Г.М.", "Индивидуальные", 2, 6],

["2024-03-01 16:00", "Гимнастика", "Соколова И.А.", "Индивидуальные", 1, 7],

["2024-03-01 17:00", "Бокс", "Волков С.П.", "Индивидуальные", 2, 8],

["2024-03-02 10:00", "Фехтование", "Попов А.С.", "Индивидуальные", 2, 9],

["2024-03-02 11:00", "Йога", "Новикова О.В.", "Групповые", 15, 10],

["2024-03-02 12:00", "Скалолазание", "Орлов М.И.", "Индивидуальные", 1, 11],

["2024-03-02 14:00", "Сноуборд", "Лебедев К.А.", "Групповые", 8, 12]

];

function **insertData**() {

return new Promise((**resolve**, **reject**) => {

db.serialize(() => {

try {

*// Начинаем транзакцию*

db.run('BEGIN TRANSACTION');

*// Вставка локаций*

const locationStmt = db.prepare('INSERT OR IGNORE INTO locations (name) VALUES (?)');

locations.forEach(*location* => {

locationStmt.run(*location*);

});

locationStmt.finalize();

*// Вставка событий*

const eventStmt = db.prepare(`

INSERT INTO sports\_events

(date\_time, sport\_type, coach, category, participants\_count, location\_id)

VALUES (?, ?, ?, ?, ?, ?)

`);

events.forEach(*event* => {

eventStmt.run(...*event*);

});

eventStmt.finalize();

*// Завершаем транзакцию*

db.run('COMMIT', (*err*) => {

if (*err*) reject(*err*);

else resolve();

});

} catch (error) {

db.run('ROLLBACK');

reject(error);

}

});

});

}

module.exports = { **insertData** };

db.js  
const sqlite3 = require('sqlite3').verbose();

const path = require('path');

const fs = require('fs');

const dbPath = path.join(\_\_dirname, 'sports.db');

*// Удаляем существующую БД если она есть*

if (fs.existsSync(dbPath)) {

fs.unlinkSync(dbPath);

}

const db = new sqlite3.Database(dbPath);

function **initDatabase**() {

return new Promise((**resolve**, **reject**) => {

db.serialize(() => {

try {

*// Создание таблицы locations*

db.run(`

CREATE TABLE IF NOT EXISTS locations (

id INTEGER PRIMARY KEY AUTOINCREMENT,

name TEXT NOT NULL UNIQUE

)

`);

*// Создание таблицы sports\_events*

db.run(`

CREATE TABLE IF NOT EXISTS sports\_events (

id INTEGER PRIMARY KEY AUTOINCREMENT,

date\_time TEXT NOT NULL,

sport\_type TEXT NOT NULL,

coach TEXT NOT NULL,

category TEXT NOT NULL CHECK(category IN ('Командные', 'Индивидуальные', 'Групповые')),

participants\_count INTEGER NOT NULL CHECK(participants\_count > 0),

location\_id INTEGER,

FOREIGN KEY (location\_id) REFERENCES locations(id)

)

`);

resolve();

} catch (error) {

reject(error);

}

});

});

}

module.exports = { db, **initDatabase** };

package.json

{

"name": "lab6-sqlite",

"version": "1.0.0",

"main": "server.js",

"scripts": {

"start": "node server.js",

"dev": "nodemon server.js"

},

"dependencies": {

"cors": "^2.8.5",

"express": "^4.18.2",

"sqlite3": "^5.1.7"

},

"devDependencies": {

"nodemon": "^2.0.22"

}

}