Министерство науки и высшего образования РФ

Федеральное государственное автономное

образовательное учреждение высшего образования

«**СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ**»

Институт управления бизнес-процессами

Кафедра «Бизнес информатика и моделирование бизнес-процессов»

**КУРСОВАЯ РАБОТА**

«Создание web-приложения с применением современных web-технологий по тематике «Спорт»»

тема

Руководитель \_\_\_\_\_\_\_\_ А.И. Никитин

подпись, дата инициалы, фамилия

Студент УБ22-08Б, 432218330 \_\_\_\_\_\_\_\_ Т.Д. Гришаев

номер группы, зачетной книжки подпись, дата инициалы, фамилия

Красноярск 2023

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc154443562)

[1 Теоретическая часть 5](#_Toc154443563)

[1.1 HTML и CSS 5](#_Toc154443564)

[1.2 JavaScript 6](#_Toc154443565)

[1.3 PHP 8](#_Toc154443566)

[1.4 MySQL 10](#_Toc154443567)

[2 Практическая часть 12](#_Toc154443568)

[2.1 Требования и описание реализации 12](#_Toc154443569)

[2.2 Общая структура 13](#_Toc154443570)

[2.3 Создание базы данных 14](#_Toc154443571)

[2.4 Регистрация в web-приложении 17](#_Toc154443572)

[2.5 Авторизация в web-приложении 24](#_Toc154443573)

[2.6 Бронирование столика в web-приложении 27](#_Toc154443574)

[2.7 Личный кабинет пользователя в web-приложении 33](#_Toc154443575)

[2.1.7 Функционал обычного пользователя 33](#_Toc154443576)

[2.2.7 Функционал администратора 41](#_Toc154443577)

[ЗАКЛЮЧЕНИЕ 46](#_Toc154443578)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 47](#_Toc154443579)

# ВВЕДЕНИЕ

В настоящее время в связи с глобальным развитием сети Интернет, в программировании все более резко начала выделяться отдельная его отрасль web-программирование. Система общественного питания как никогда нуждается в автоматизации различных внутренних процессов, в частности в области спорта. Web-разработка позволяет сделать высокотехнологичный продукт с интересным функционалом как для пользователя, так и для администратора.

Создание базы данных MySQL для web-приложения является важным шагом для эффективной работы приложения. Ниже перечислены несколько важных аспектов, которые обосновывают важность создания такой базы данных:

База данных позволяет эффективно управлять данными, хранить информацию о пользователях, просматривать список спортивных мероприятий, оставлять отзывы и оценки, а также смотреть дату и место спортивного мероприятия. Это позволяет упростить процесс поиска и обработки данных.

Нельзя не упомянуть о хранении большого объема данных: web-приложение, связанное со спортивной деятельностью, может содержать значительное количество информации о пользователях, мероприятиях, оценках пользователей, спортивных событиях и других связанных данных. База данных MySQL обеспечивает надежное хранение этих данных и позволяет эффективно управлять ими, даже при большом объеме информации.

Поддержка множества пользователей: Просмотр мероприятий может быть доступен для множества пользователей, как для посетителей, так и для администраторов. MySQL обеспечивает многопользовательскую поддержку, позволяя одновременно работать с базой данных нескольким пользователям без проблем с доступом и интеграцией данных.

В целом, создание базы данных MySQL для веб-приложения по тематике спорт — это ключевой аспект, который способствует эффективной работе приложения, улучшает управление данными, обеспечивает безопасность и защиту данных, а также позволяет взаимодействие с другими системами.

Целью курсовой работы является разработка сайта, используя технологии HTML, CSS, Javascript, PHP, MySQL, согласно теме «Создание web-приложения с применением современных web-технологий по тематике «Спорт»», а также разработка базы данных для управления пользователями, используя СУБД MySQL.

Объектом исследования является web-приложение с применением различных web-технологий.

Предметом исследования является процесс создания web-приложения, базы данных.

Методами иcследования являются: методы анализа и синтеза информации, проектирование, практическая реализация.

# 1 Теоретическая часть

## HTML и CSS

HTML (HyperText Markup Language) - это язык разметки, используемый для создания структуры и содержания web-страниц. Он состоит из серии элементов (тегов), которые определяют различные части страницы, такие как заголовки, параграфы, изображения, таблицы и т.д. HTML позволяет структурировать содержимое, чтобы браузер мог правильно отображать страницу.

CSS (Cascading Style Sheets) - это язык таблиц стилей, используемый для описания внешнего вида web-страницы. Он позволяет определить различные свойства элементов HTML, такие как шрифт, цвет, размер, отступы, позиционирование и др. CSS позволяет разработчикам создавать стильные и привлекательные веб-страницы, облегчая работу с разметкой HTML.

Применение HTML и CSS в веб-разработке является критически важным. Вот несколько причин, почему:

1. Структура и семантика: HTML помогает определить структуру и логическую семантику веб-страницы. Это делает содержимое более доступным для поисковых систем и улучшает взаимодействие со скринридерами для людей с ограниченными возможностями.

2. Внешний вид и пользовательский опыт: CSS позволяет разработчикам создавать привлекательные и стильные веб-страницы. Он дает возможность настраивать внешний вид элементов HTML, что позволяет создавать индивидуальный и уникальный дизайн в соответствии с брендингом и предпочтениями пользователей. Это важно для создания положительного пользовательского опыта.

3. Адаптивность и отзывчивость: HTML и CSS позволяют создавать адаптивные и отзывчивые веб-страницы. Это означает, что веб-сайты могут автоматически адаптироваться к различным устройствам и экранам, таким как мобильные телефоны, планшеты и настольные компьютеры. Это важно для обеспечения удобства использования и достижения большей аудитории.

4. Разделение ответственности: HTML отвечает за структуру и содержание веб-страницы, а CSS за внешний вид. Это разделение позволяет разработчикам более эффективно работать в команде, улучшает обслуживание и повторное использование кода.

В целом, применение HTML и CSS в веб-разработке является неотъемлемой частью создания высококачественных и стильных веб-страниц. Они обеспечивают важные функции, которые улучшают доступность, пользовательский опыт и удобство использования веб-сайтов.

## JavaScript

JavaScript — это интерпретируемый язык программирования, используемый для создания динамических и интерактивных web-сайтов. Он позволяет web-разработчикам добавлять различные функции и взаимодействие на веб-страницах, что помогает создавать более интересные и привлекательные пользовательские интерфейсы.

Несколько важных функций JavaScript:

1. Манипуляция HTML-элементами: JavaScript позволяет изменять и модифицировать HTML-код страницы, добавлять, удалять или изменять элементы, классы, стили и атрибуты.

2. Обработка событий: JavaScript позволяет реагировать на различные действия пользователя на web-странице, такие как нажатие кнопки, клик мыши, наведение курсора и многое другое. Это позволяет создавать интерактивные и отзывчивые пользовательские интерфейсы.

3. Асинхронная загрузка данных: JavaScript позволяет загружать данные с сервера без перезагрузки всей страницы. Это позволяет создавать динамические приложения, которые могут обновляться без прерывания работы пользователя.

4. Работа с данными: JavaScript предоставляет множество возможностей для обработки и манипуляции данными, включая их чтение, запись, редактирование и преобразование. Это особенно полезно при создании форм, валидации данных и обработке результатов пользовательского взаимодействия.

5. Создание анимаций: JavaScript позволяет создавать анимацию на web-страницах, изменяя стили и свойства элементов, перемещая их или изменяя их размеры. Это позволяет создавать более привлекательные и интерактивные пользовательские интерфейсы.

Важность JavaScript для web-разработки состоит в его способности добавлять динамичность и интерактивность на веб-страницы. Он позволяет создавать более привлекательные визуальные эффекты, улучшать пользовательский опыт, улучшать функциональность и сокращать время загрузки страницы. Большинство современных web-сайтов и приложений используют JavaScript, поэтому знание этого языка программирования является важным навыком для веб-разработчиков.

Некоторые достоинства JavaScript:

1. Широкая поддержка: JavaScript поддерживается всеми современными web-браузерами, что позволяет создавать совместимые и доступные web-сайты для всех пользователей.

2. Простота в изучении: JavaScript относительно прост в изучении и понимании. Он имеет синтаксис, похожий на другие языки программирования, такие как Java и C++, и множество ресурсов и учебных материалов для начинающих.

3. Интеграция с HTML и CSS: JavaScript легко интегрируется с HTML и CSS, что позволяет создавать привлекательные и кросс-платформенные web-приложения.

4. Большое сообщество: JavaScript имеет очень активное и большое сообщество разработчиков и фреймворков, что позволяет быстро решать проблемы и обмениваться опытом.

## 1.3 PHP

PHP (Hypertext Preprocessor) — это язык программирования общего назначения, который широко используется для разработки динамических web-страниц и web-приложений. Он скриптовый язык, выполняемый на сервере, что означает, что код PHP обрабатывается на сервере, а результаты отображаются на клиентском устройстве.

Важность применения PHP в веб-разработке заключается в следующем:

1. Простота использования: PHP обладает простым и понятным синтаксисом, что делает его доступным для начинающих разработчиков. Большое сообщество PHP разработчиков также означает наличие большого количества обучающих материалов и поддержки.

2. Широкие возможности: PHP поддерживает множество баз данных (MySQL, PostgreSQL, Oracle и другие) и протоколов (HTTP, FTP, SMTP и другие), что позволяет разрабатывать разнообразные веб-приложения.

3. Интеграция со сторонними системами: PHP легко интегрируется с другими технологиями, такими как HTML, CSS, JavaScript, что позволяет создавать сложные веб-приложения и веб-сайты.

4. Открытый и свободный: PHP является Open Source, что означает, что разработчики могут свободно использовать и изменять его в соответствии с собственными потребностями. Это также означает наличие огромной базы бесплатных расширений и фреймворков для PHP.

Функции PHP включают:

1. Генерация динамического контента: благодаря PHP, веб-разработчики могут создавать динамические страницы, которые автоматически обновляются и адаптируются в зависимости от действий пользователя.

2. Работа с формами: PHP позволяет легко обрабатывать данные, полученные от пользователей через формы на веб-страницах. Это позволяет создавать интерактивные приложения, такие как регистрация пользователей, отправка сообщений и т. д.

3. Работа с базами данных: PHP предоставляет мощные функции для работы с базами данных, что позволяет хранить, изменять и получать данные из баз данных.

4. Организация и обработка файлов: PHP предоставляет возможности для работы с файлами на сервере, включая чтение, запись и удаление данных из файлов.

5. Обработка ошибок и исключений: PHP позволяет разработчикам контролировать и обрабатывать ошибки и исключения, что помогает создавать более надежные и безопасные приложения.

В целом, PHP является мощным инструментом для разработки веб-приложений, и его популярность продолжает расти благодаря своей простоте использования и большому количеству возможностей.

## MySQL

MySQL — это система управления базами данных, которая позволяет организовывать и управлять большими объемами данных. Это одна из наиболее популярных реляционных СУБД, используемых для работы с веб-приложениями.

Функции, предоставляемые MySQL, включают в себя:

1. Создание и управление базами данных: MySQL позволяет создавать новые базы данных, а также управлять существующими базами данных, включая добавление, изменение и удаление таблиц и полей.

2. Хранение и обработка данных: MySQL позволяет хранить и обрабатывать различные типы данных, такие как числа, текст, даты и другие. Он также поддерживает операции с данными, такие как вставка, обновление и выборка данных из таблицы.

3. Обеспечение безопасности данных: MySQL предоставляет функции шифрования и аутентификации данных для обеспечения безопасности хранимых данных. Он также поддерживает ролевую модель доступа, что позволяет контролировать доступ пользователей к базе данных и ее объектам.

4. Высокая производительность: MySQL разработан с учетом высокой производительности и масштабируемости. Он способен обрабатывать большое количество запросов одновременно и обеспечивать быстрый доступ к данным.

Применение MySQL для веб-разработки, в частности для сайта по тематике «Спорт», имеет важное значение по следующим причинам:

1. Хранение данных: MySQL позволяет сохранять информацию о имени пользователя, его email и пароле и т. д. А также обеспечивает надежность хранения данных.

2. Обработка запросов: С помощью MySQL можно легко выполнять запросы на выборку, обновление и удаление данных, что позволяет эффективно управлять информацией.

3. Масштабируемость: MySQL поддерживает горизонтальное и вертикальное масштабирование, что позволяет расширять функциональность системы.

4. Безопасность данных: MySQL обеспечивает защиту данных, позволяя контролировать доступ пользователей к базе данных и предоставляя возможности шифрования для защиты конфиденциальной информации.

В целом, использование MySQL для создания web-приложения с применением современных web-технологий по тематике «Спорт» позволяет эффективно обрабатывать данные, а также обеспечивает безопасность хранимых данных и способствует развитию функционала системы.

# 2 Практическая часть

## 2.1 Требования и описание реализации

Как уточнялось ранее, курсовой проект предполагает создание web-приложение на тему «Создание web-приложения с применением современных web-технологий по тематике «Спорт»», используя технологии HTML, CSS, Javascript, PHP, MySQL. Требуется реализовать добавление, редактирование, просмотр и удаление пользователей. Также предусмотреть функции авторизации и аутентификации.

**Описание реализации:**

- неавторизированные пользователи должны иметь возможность регистрироваться. (форма добавления пользователя)

- авторизированные пользователи разделить на 2 группы: администраторы и авторизированные пользователи.

- администраторы имеют право просматривать, добавлять, редактировать и удалять пользователей.

- авторизированные пользователи имеют право просматривать и редактировать только свои данные.

- при регистрации пользователю присваивается статус авторизированного пользователя.

**Требования к проекту:**

1. Должна быть предусмотрена авторизация и регистрация пользователей.
2. Зарегистрированный пользователь имеет право редактировать свои данные и просматривать свои услуги.
3. Администраторы имеют право редактировать пользователей и добавлять услуги.
4. Неавторизированный пользователи имеют право просматривать список услуг
5. Сайт должен быть выдержан в определенном стиле и корректно открываться на любом разрешении.
6. Общий функционал может быть продуман самостоятельно.
7. Важно продемонстрировать не большой объем функционала, а минимальные требования к проекту. Проект может максимально простым, но при этом содержать все необходимые функции.
8. Отдельные бонусы предусмотрены за доп. функционал и красивый дизайн.

## 2.2 Общая структура

Установка open server позволяет воспроизводиться web-приложению локально. С помощью встроенного функционала (рис.1) создалась база данных с кодировкой *utf8\_general\_ci* (рис.2).

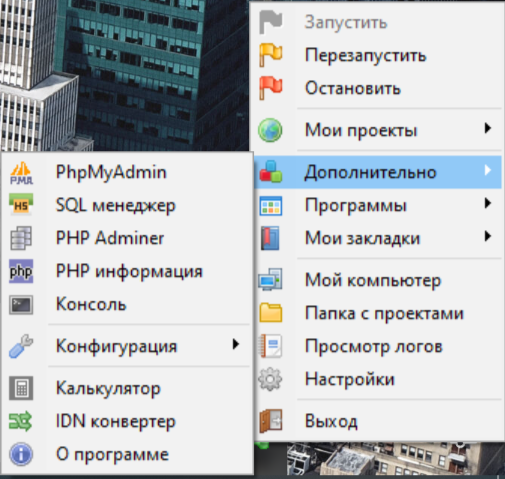


Рисунок 1 — Open server: встроенный функционал

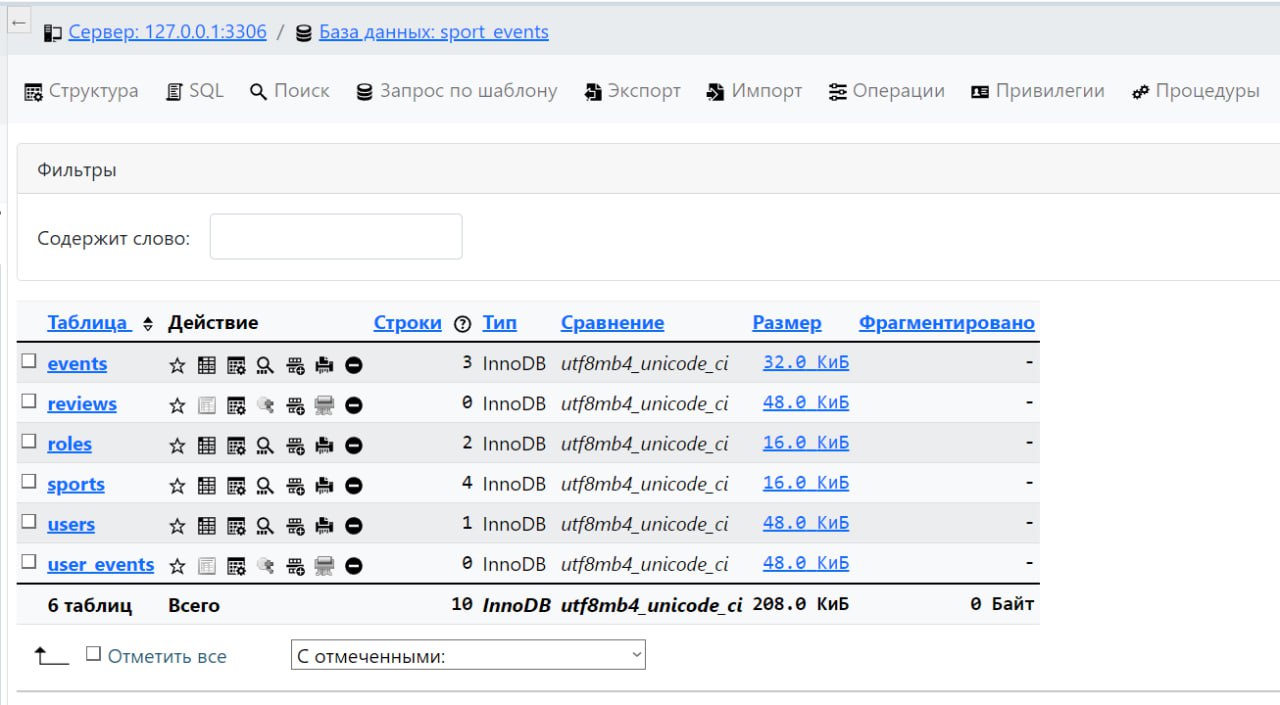


Рисунок 2 — База данных

Далее была установлена система управления базами данных MySQL, которая позволяет организовывать и управлять большими объемами данных.



Рисунок 3 — Установка MySQL

## 2.3 Создание базы данных

Листинг создания базы данных:

-- Создание базы данных

CREATE DATABASE IF NOT EXISTS sport\_events;

-- Использование созданной базы данных

USE sport\_events;

-- Создание таблицы видов спорта

CREATE TABLE IF NOT EXISTS sports (

sport\_id INT PRIMARY KEY AUTO\_INCREMENT,

sport\_name VARCHAR(50) NOT NULL

);

-- Вставка примера данных в таблицу видов спорта

INSERT INTO sports (sport\_name) VALUES

('Футбол'),

('Баскетбол'),

('Теннис');

-- Создание таблицы мероприятий

CREATE TABLE IF NOT EXISTS events (

event\_id INT PRIMARY KEY AUTO\_INCREMENT,

sport\_id INT,

event\_name VARCHAR(100) NOT NULL,

event\_date DATE,

event\_location VARCHAR(200) NOT NULL,

FOREIGN KEY (sport\_id) REFERENCES sports(sport\_id)

);

-- Создание таблицы ролей

CREATE TABLE IF NOT EXISTS roles (

role\_id INT PRIMARY KEY AUTO\_INCREMENT,

role\_name VARCHAR(50) NOT NULL

);

-- Вставка примера данных в таблицу ролей

INSERT INTO roles (role\_name) VALUES

('admin'),

('user');

-- Создание таблицы пользователей

CREATE TABLE IF NOT EXISTS users (

user\_id INT PRIMARY KEY AUTO\_INCREMENT,

username VARCHAR(50) NOT NULL,

email VARCHAR(255) UNIQUE NOT NULL,

password VARCHAR(255) NOT NULL,

role\_id INT,

FOREIGN KEY (role\_id) REFERENCES roles(role\_id)

);

-- Создание таблицы отзывов

CREATE TABLE IF NOT EXISTS reviews (

review\_id INT PRIMARY KEY AUTO\_INCREMENT,

event\_id INT,

user\_id INT,

comment TEXT NOT NULL,

rating INT,

FOREIGN KEY (event\_id) REFERENCES events(event\_id),

FOREIGN KEY (user\_id) REFERENCES users(user\_id)

);

-- Создание таблицы регистраций пользователей на мероприятия

CREATE TABLE IF NOT EXISTS user\_events (

user\_event\_id INT PRIMARY KEY AUTO\_INCREMENT,

event\_id INT,

user\_id INT,

registration\_date DATETIME,

FOREIGN KEY (event\_id) REFERENCES events(event\_id),

FOREIGN KEY (user\_id) REFERENCES users(user\_id)

);

Название базы данных sport\_events с кодировкой *utf8\_general\_ci*. Создаётся таблица ролей, которая имеет role\_id и role\_name (роль админа или пользователя). После создаём таблицу пользователей, которая содержит в себе user\_id, username, password, email, role\_id. Примечание: электронная почта должна быть уникальной, зарегистрировать двух пользователей на одну и ту же почту невозможно. Далее создаётся таблица пользователей, которая содержит в себе id пользователя, его email и пароль. После создаётся таблица отзывов, которая связывает пользователя с его отзывом (содержит в себе сам отзыв и оценку).

Также используется разделение ролей, чтобы привилегии администратора и обычного пользователя имели различия.

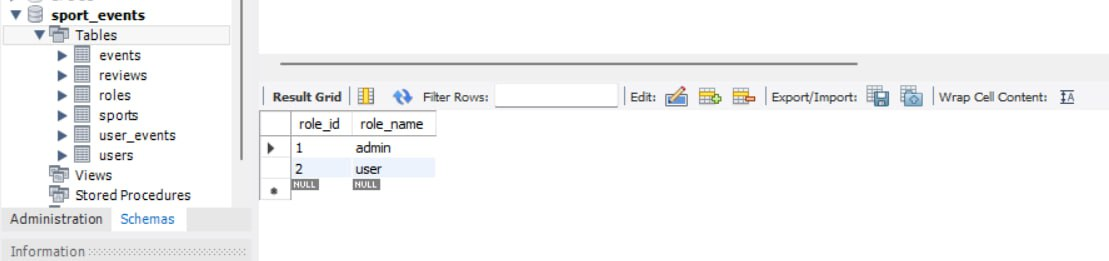


Рисунок 4 — База данных: роль пользователя и админа

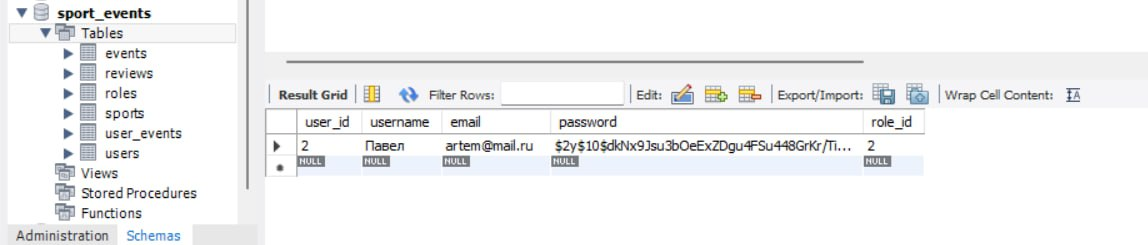


Рисунок 5 — Пример регистрации пользователя

На рисунке 5 показываются данные пользователя, такие как его user\_id, имя, email, пароль и его роль.

## 2.4 Регистрация в web-приложении

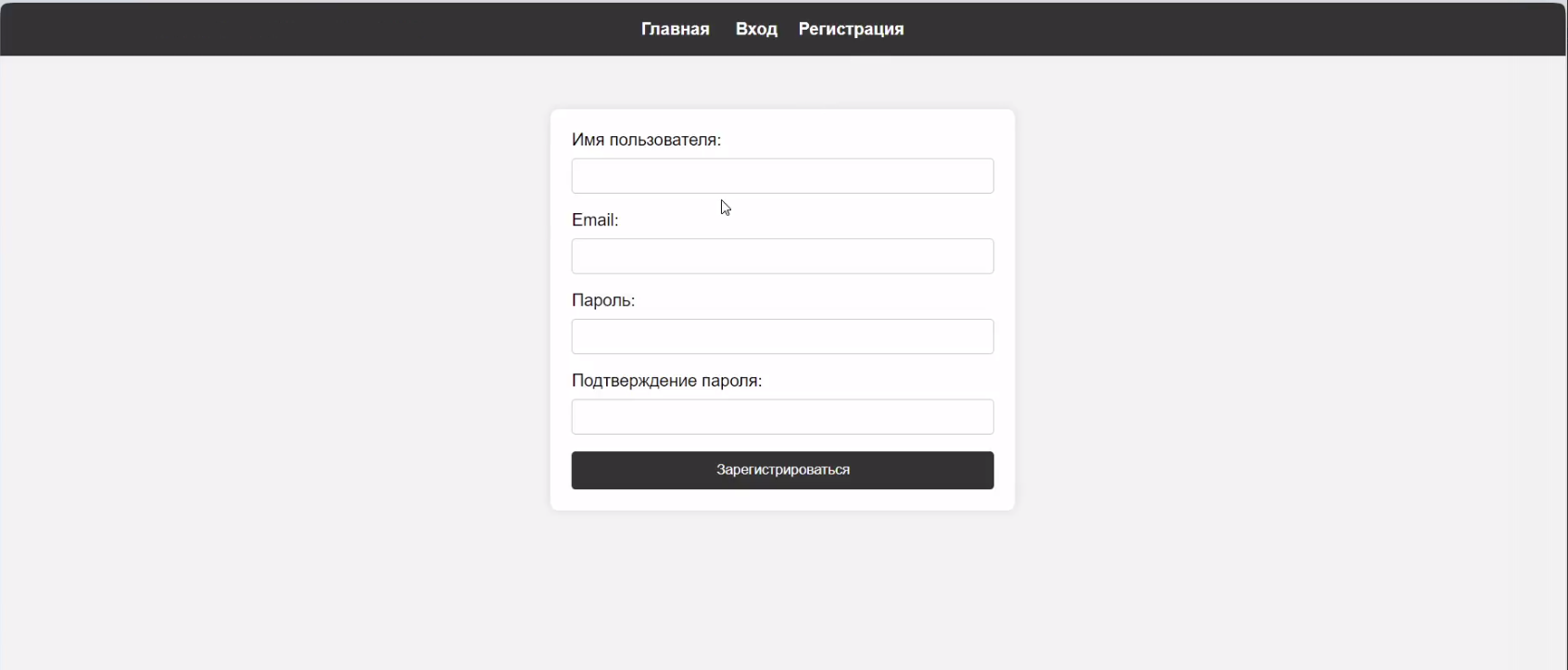


Рисунок 6 — Регистрация в web-приложении

Листинг HTML составляющей панели регистрации:

<?php

include "db.php";

session\_start();

if (isset($\_SESSION["user\_id"])) {

header("Location: profile.php");

exit();

}

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

$username = $\_POST["username"];

$password = $\_POST["password"];

$confirm\_password = $\_POST["confirm\_password"];

$email = $\_POST["email"];

if (!filter\_var($email, FILTER\_VALIDATE\_EMAIL)) {

$error = "Некорректный формат email";

} elseif (!preg\_match("/^[a-zA-Zа-яА-Я0-9\_]+$/u", $username)) {

$error = "Имя пользователя может содержать только буквы, цифры и подчеркивания";

} elseif (mb\_strlen($password) < 8) {

$error = "Пароль должен содержать не менее 8 символов";

} elseif ($password !== $confirm\_password) {

$error = "Пароли не совпадают";

} else {

$email\_check\_sql = "SELECT \* FROM users WHERE email = '$email'";

$email\_check\_result = $conn->query($email\_check\_sql);

if ($email\_check\_result->num\_rows > 0) {

$error = "Пользователь с таким email уже зарегистрирован";

} else {

$hashed\_password = password\_hash($password, PASSWORD\_DEFAULT);

$role\_id = 2;

$sql = "INSERT INTO users (username, password, email, role\_id) VALUES ('$username', '$hashed\_password', '$email', '$role\_id')";

if ($conn->query($sql) === TRUE) {

header("Location: login.php");

exit();

} else {

$error = "Ошибка при регистрации: " . $db\_connection->error;

}

}

}

}

include "header.php";

?>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Регистрация</title>

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

</head>

<body>

<section>

<form id="registrationForm" action="" method="post">

<label for="username">Имя пользователя:</label>

<input type="text" id="username" name="username" required>

<label for="email">Email:</label>

<input type="email" id="email" name="email" required>

<label for="password">Пароль:</label>

<input type="password" id="password" name="password" required>

<label for="confirm\_password">Подтверждение пароля:</label>

<input type="password" id="confirm\_password" name="confirm\_password" required>

<button type="submit">Зарегистрироваться</button>

</form>

<?php

if (isset($error)) {

echo "<p class='error'>$error</p>";

}

?>

</section>

<script>

document.getElementById("registrationForm").addEventListener("submit", function(event) {

var emailInput = document.getElementById("email");

var emailPattern = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;

if (!emailPattern.test(emailInput.value)) {

alert("Некорректный формат email");

event.preventDefault();

return;

}

var passwordInput = document.getElementById("password");

if (passwordInput.value.length < 8) {

alert("Пароль должен содержать не менее 8 символов");

event.preventDefault();

return;

}

var confirmPasswordInput = document.getElementById("confirm\_password");

if (passwordInput.value !== confirmPasswordInput.value) {

alert("Пароли не совпадают");

event.preventDefault();

return;

}

});

</script>

</body>

</html>

Форма регистрации содержит в себе поля для ввода имени пользователя, электронной почты, пароля, строки подтверждения пароля, а также кнопки «зарегистрироваться».

С помощью тега <link rel="stylesheet" href="registration.css"> подключаем стили. Листинг стилей для панели регистрации:

body {

font-family: 'Arial', sans-serif;

margin: 0;

padding: 0;

background-color: #f4f4f4;

}

header {

background-color: #333;

color: white;

text-align: center;

padding: 1em;

}

section {

max-width: 400px;

margin: 50px auto;

background-color: white;

padding: 20px;

border-radius: 8px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

}

form {

display: flex;

flex-direction: column;

}

label {

margin-bottom: 8px;

}

input {

padding: 8px;

margin-bottom: 16px;

border: 1px solid #ccc;

border-radius: 4px;

}

button {

background-color: #333;

color: white;

padding: 10px;

border: none;

border-radius: 4px;

cursor: pointer;

}

.error {

color: #c0392b;

margin-top: 10px;

}

Далее при отправке формы активируется PHP код, устанавливается подключение к базе данных (подключение к локальному хосту). Указывается имя базы данных, имя пользователя, имеющего доступ и пароль.

Листинг:

<?php

$servername = "localhost";

$username = "root";

$password = "";

$dbname = "sport\_events";

$conn = new mysqli($servername, $username, $password, $dbname);

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

function getUserInfoFromDatabase($userId) {

global $conn;

$sql = "SELECT \* FROM users WHERE user\_id = $userId";

$result = $conn->query($sql);

if ($result->num\_rows > 0) {

return $result->fetch\_assoc();

} else {

return false;

}

}

?>

## 2.5 Авторизация в web-приложении

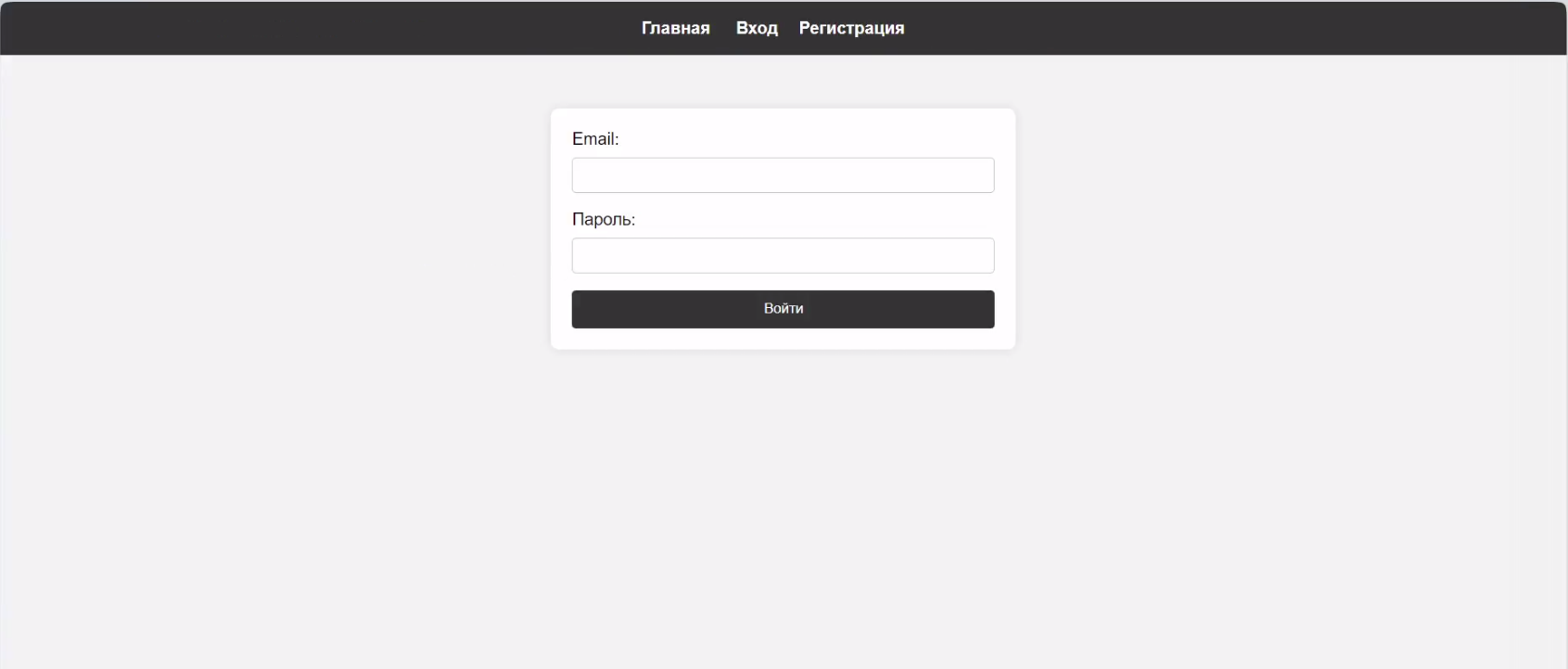


Рисунок 7 — Авторизация в web-приложении

Листинг HTML составляющей панели авторизации:

<?php

include "db.php";

session\_start();

if (isset($\_SESSION["user\_id"])) {

header("Location: profile.php");

exit();

}

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

$email = $\_POST["email"];

$password = $\_POST["password"];

$sql = "SELECT \* FROM users WHERE email = '$email'";

$result = $conn->query($sql);

if ($result->num\_rows > 0) {

$row = $result->fetch\_assoc();

if (password\_verify($password, $row["password"])) {

session\_start();

$\_SESSION["user\_id"] = $row["user\_id"];

$\_SESSION["username"] = $row["username"];

$\_SESSION["role\_id"] = $row["role\_id"];

header("Location: profile.php");

exit();

} else {

$error = "Неверный пароль";

}

} else {

$error = "Пользователь не найден";

}

}

include "header.php";

?>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Вход</title>

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

</head>

<body>

<section>

<form id="loginForm" action="" method="post">

<label for="email">Email:</label>

<input type="email" id="email" name="email" required>

<label for="password">Пароль:</label>

<input type="password" id="password" name="password" required>

<button type="submit">Войти</button>

</form>

<?php

if (isset($error)) {

echo "<p class='error'>$error</p>";

}

?>

</section>

</body>

</html>

Авторизация включает в себя форму ввода электронной почты, пароль и кнопку «войти».

Листинг стилей для панели авторизации:

body {

font-family: 'Arial', sans-serif;

margin: 0;

padding: 0;

background-color: #f4f4f4;

}

header {

background-color: #333;

color: white;

text-align: center;

padding: 1em;

}

section {

max-width: 400px;

margin: 50px auto;

background-color: white;

padding: 20px;

border-radius: 8px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

}

form {

display: flex;

flex-direction: column;

}

label {

margin-bottom: 8px;

}

input {

padding: 8px;

margin-bottom: 16px;

border: 1px solid #ccc;

border-radius: 4px;

}

button {

background-color: #333;

color: white;

padding: 10px;

border: none;

border-radius: 4px;

cursor: pointer;

}

.error {

color: #c0392b;

margin-top: 10px;

}

## 2.6 Добавление вида спорта в web-приложении

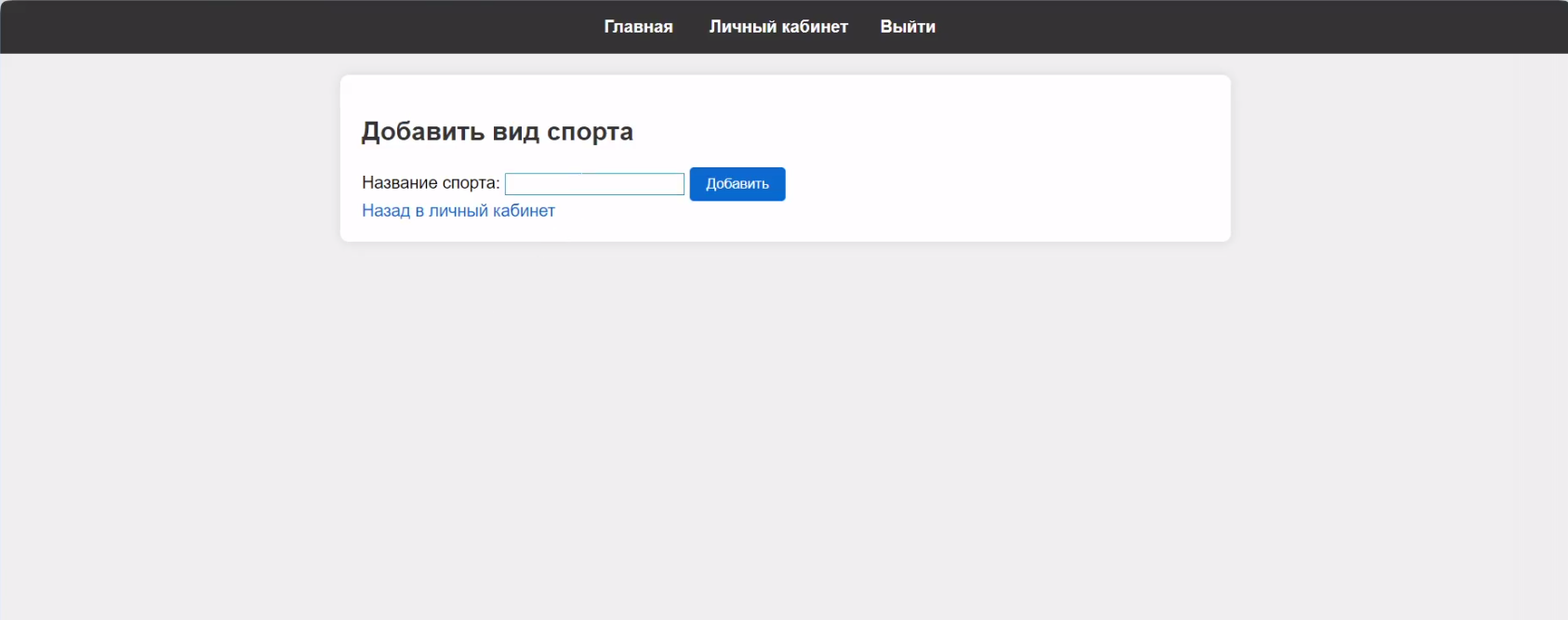


Рисунок 8 — Добавление вида спорта в web-приложении

Листинг HTML составляющей панели бронирования:

<?php

session\_start();

if (!isset($\_SESSION["user\_id"]) && $\_SESSION["role\_id"] != 1) {

header("Location: login.php");

exit();

}

include "db.php";

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

$sportName = $\_POST["sport\_name"];

if (empty($sportName)) {

$error = "Введите название спорта";

} else {

$insertSql = "INSERT INTO sports (sport\_name) VALUES ('$sportName')";

if ($conn->query($insertSql) === TRUE) {

$successMessage = "Новый вид спорта успешно добавлен";

} else {

$error = "Ошибка при добавлении вида спорта: " . $conn->error;

}

}

}

include "header.php";

?>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Добавить вид спорта</title>

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

</head>

<body>

<section>

<h2>Добавить вид спорта</h2>

<form id="addSportForm" action="" method="post">

<label for="sport\_name">Название спорта:</label>

<input type="text" id="sport\_name" name="sport\_name" required>

<button type="submit">Добавить</button>

</form>

<?php

if (isset($error)) {

echo "<p class='error'>$error</p>";

} elseif (isset($successMessage)) {

echo "<p class='success'>$successMessage</p>";

}

?>

<a href="profile.php">Назад в личный кабинет</a>

</section>

</body>

</html>

Также имеется кнопка «выход». Реализованный код для разрыва пользователя с сайтом (кнопка «выход»).

Листинг:

<?php

session\_start();

session\_destroy();

header("Location: login.php");

exit();

?>

После создания мероприятия все данные отправляются на сервер.

# ЗАКЛЮЧЕНИЕ

В заключении хочется сказать, что создание web-приложений играет важную роль в современном обществе, поскольку они обладают большим потенциалом для улучшения работы и удовлетворения потребностей в различных сферах.

Web-приложение в тематике спорта имеет важную роль в улучшении жизни и автоматизации web-процессов.

Одна из основных функций такого приложения - обеспечение удобства и доступности информации для пользователей. Пользователи могут удобно выбирать спортивное мероприятие и посмотреть время и дату. Они могут также оставить отзыв и оценку мероприятию.

Для владельцев спортивных клубов web-приложение предоставляет мощный инструмент для объявления информации о предстоящих спортивных мероприятиях. В целом, web-приложение в сфере спорта играет роль мощного инструмента для улучшения опыта клиентов, упрощения и автоматизации процесса организации спортивных мероприятий.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Дуванов А. Web-конструирование. HTML / А. Дуванов. - СПб.: БХВ-Петербург, 2013. - 321 с.
2. Белунцов В. Новейший самоучитель по созданию Web-страниц / В. Белунцов. - М.: NT Press, 2012. - 185 с.
3. Браун Брэдли Oracle Database. Создание Web-приложений: учебник / Брэдли Браун, Ричард Ниемик, Джозеф С. Треззо. - СПб.: Лори, 2011. - 722 с.
4. Алексеев Ю.М. Быстро и легко создаем, программируем, шлифуем и раскручиваем web-сайт: учеб. пособие / Ю.М. Алексеев. - М.: Лучшие книги, 2011. - 189 с.
5. Гончаров А.Ю. Web-дизайн: HTML, JavaScript и CSS. Карманный справочник / А.Ю. Гончаров. - М.: «КУДИЦ-ПРЕСС», 2010. - 320 с.
6. Дунаев В. JavaScript: учеб. пособие / В. Дунаев. - СПб.: Питер, 2010. - 394 с.
7. Кожемякин А. HTML и CSS в примерах. Создание Web-страниц: учебник / А. Кожемякин. - М.: Альтекс-А, 2011. - 415 с.
8. Яргер, Р.Дж. MySQL и mSQL: Базы данных для небольших предприятий и Интернета / Р.Дж. Яргер, Дж. Риз, Т. Кинг. - М.: СПб: Символ-Плюс, **2014**. - 560 c.