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 «Кино»»

тема

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

подпись, дата инициалы, фамилия

Студент УБ22-09Б, 432213318 \_\_\_\_\_\_\_\_ А.Е. Кузьминых

номер группы, зачетной книжки подпись, дата инициалы, фамилия

Красноярск 2023

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc156077579)

[1 Теоретическая часть 5](#_Toc156077580)

[1.1 HTML 5](#_Toc156077581)

[1.2 CSS 6](#_Toc156077582)

[1.3 JavaScript 7](#_Toc156077583)

[1.4 PHP 8](#_Toc156077584)

[1.5 MySQL 9](#_Toc156077585)

[2 Практическая часть 12](#_Toc156077586)

[2.1 Требования 12](#_Toc156077587)

[2.2 Mamp и создание базы данных 13](#_Toc156077588)

[2.3 Регистрация и авторизация в web-приложении 14](#_Toc156077589)

[2.4 Личный кабинет пользователя в web-приложении 19](#_Toc156077590)

[2.4.1 Функционал User 19](#_Toc156077591)

[2.4.2 Функционал Аdmin 23](#_Toc156077592)

[2.5 Главная страница 38](#_Toc156077593)

[ЗАКЛЮЧЕНИЕ 47](#_Toc156077594)

# ВВЕДЕНИЕ

В современном информационном обществе наблюдается неутомимый рост интернет-технологий, что приводит к повышенному спросу на разработку веб-приложений. Их создание является актуальной проблемой, поскольку веб-приложения играют ключевую роль в обеспечении пользователей доступом к различным сервисам и функционалу через интернет.

Одним из основных инструментов для разработки веб-приложений является набор языков и технологий, которые, сотрудничая вместе, позволяют разработчикам создавать динамические и интерактивные веб-страницы. Для создания структуры и содержимого веб-страниц используется язык разметки HTML (Hypertext Markup Language), который определяет структуру и визуальное представление страницы. Для придания стилей и внешнего оформления веб-страниц веб-разработчики обычно применяют язык стилей CSS (Cascade Style Sheets).

Для организации хранения и управления данными в веб-приложениях широко применяются базы данных. Одной из популярных систем управления базами данных является MySQL, которая предоставляет эффективные средства для создания, управления и манипулирования данными.

Для обработки данных и взаимодействия клиента с сервером веб-приложения активно используют язык программирования PHP (PHP: Hypertext Preprocessor). PHP позволяет создавать динамически генерируемые веб-страницы и обеспечивает взаимодействие с базами данных.

Java Script, в свою очередь, является клиентским языком программирования и позволяет создавать интерактивные элементы на веб-страницах, такие как формы, анимации и другие динамические функции.

Таким образом, комбинированное применение HTML, CSS, MySQL, PHP и Java Script позволяет разработчикам создавать мощные и удобные веб-приложения, обеспечивая им функциональность и эстетическое оформление. В данной курсовой работе будет рассмотрено использование данных средств для разработки веб-приложения, их преимущества и возможности применения.

Целью курсовой работы является разработка сайта, используя технологии HTML, CSS, Javascript, PHP, MySQL, согласно теме «Бронирование столиков в ресторане», 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

HTML (HyperText Markup Language) - это язык разметки, используемый для создания и оформления веб-страниц. HTML отвечает за структуру страницы, определяет, какие элементы на странице будут заголовками, абзацами, списками, ссылками и т. д. Он является основой для создания веб-сайтов и интернет-приложений.

Одной из основных возможностей HTML является создание гиперссылок, с помощью которых пользователь может перемещаться по разным страницам и сайтам в Интернете. Гиперссылки могут быть представлены в виде текста или кнопок и могут указывать на другой документ, раздел страницы или внешний ресурс.

HTML также предоставляет различные способы для встраивания мультимедийных элементов на веб-страницу, таких как изображения, аудио и видео файлы. Это позволяет создавать интерактивный интерфейс для удовлетворения потребностей пользователей.

Кроме того, HTML позволяет создавать формы, которые позволяют пользователям отправлять данные на сервер. Формы часто используются для сбора информации от посетителей, таких как контактные данные или результаты опросов.

Синтаксис HTML достаточно прост и понятен. Он состоит из так называемых тегов, которые обрамляют содержимое и указывают браузеру, как отображать этот контент. Каждый тег имеет открывающую и закрывающую части, а содержимое находится между ними.

HTML также обеспечивает возможность использования стилей и классов для оформления веб-страниц. Стили задаются с помощью CSS (Cascading Style Sheets) и позволяют управлять внешним видом элементов, таких как цвет фона, шрифт, размер и т. д. Добавление классов к элементам позволяет применять определенные стили только к выбранным элементам, что добавляет гибкости к разметке.

Основная цель HTML - предоставить простой и понятный способ создания структуры веб-страниц. Он является фундаментом для других языков и технологий, таких как CSS, JavaScript, PHP и другие. Благодаря своей простоте и распространенности, HTML остается языком, с которого начинается изучение разработки веб-сайтов.

## CSS

CSS, или Cascading Style Sheets, является одним из ключевых инструментов веб-разработки. Он предоставляет возможности для стилизации и форматирования веб-страниц, включая HTML-документы. CSS обеспечивает отдельное описание внешнего вида и макета элементов на странице, отделяя их от содержимого.

Одним из главных преимуществ CSS является его гибкость и мощные возможности. С помощью CSS можно изменять цвета, шрифты, размеры, расположение элементов и многое другое. Он позволяет создавать стильные, красивые и профессионально выглядящие веб-страницы.

Функциональность CSS может быть использована для создания адаптивных и отзывчивых дизайнов, которые корректно отображаются на разных устройствах и экранах. Это особенно важно в эпоху мобильных устройств, когда веб-сайты должны быть доступными и удобочитаемыми на смартфонах и планшетах.

Синтаксис CSS достаточно прост и понятен. Он основан на использовании селекторов, свойств и значений. Селекторы используются для указания, к каким элементам HTML применяются стили. Свойства определяют внешний вид элементов, а значения устанавливают конкретные параметры этих свойств.

Для подключения CSS к HTML используется тег <link> внутри раздела <head> HTML-документа. Внутри <link> указывается атрибут href, который ссылается на CSS-файл, содержащий стили. Это позволяет разделить стиль и структуру веб-страницы, облегчая ее сопровождение и избегая повторного кодирования.

CSS является эффективным инструментом для разработки и управления стилями и внешним видом веб-сайтов. Он дает разработчикам мощные и гибкие возможности для создания уникальных и привлекательных веб-интерфейсов. В сочетании с HTML и JavaScript, CSS образует основу современных веб-разработок.

CSS - это мощный язык, который позволяет разработчикам создавать привлекательные и профессионально выглядящие веб-страницы. Он предоставляет широкий спектр возможностей для стилизации и форматирования элементов на странице, обеспечивая легкость подключения к HTML и понятность синтаксиса. CSS является неотъемлемой частью веб-разработки и очень ценится за свою гибкость и мощь.

## JavaScript

JavaScript является одним из самых популярных и широко используемых языков программирования на сегодняшний день. Он был создан для придания интерактивности веб-страницам и добавления динамического функционала.

Одним из главных преимуществ JavaScript является его широкий функционал. Он позволяет разработчикам создавать сложные и интерактивные веб-приложения, обрабатывать события, взаимодействовать с сервером без перезагрузки страницы и многое другое. JavaScript также поддерживает объектно-ориентированное программирование, что делает его мощным инструментом для разработки приложений любого масштаба.

С помощью JavaScript можно создавать и использовать функции в коде. Функции позволяют упорядочивать и структурировать код, а также повторно использовать его блоки. JavaScript поддерживает как встроенные функции, так и пользовательские, что делает его более гибким и мощным инструментом.

Одним из главных преимуществ JavaScript является его способность легко подключаться к HTML. JavaScript-код может быть встроен непосредственно в HTML-страницу с помощью тега <script>, либо может быть загружен из внешнего файла. Это позволяет разработчикам легко интегрировать и использовать JavaScript в своих проектах.

Кроме того, JavaScript имеет большое сообщество разработчиков, что означает наличие множества ресурсов, форумов и библиотек для обмена опытом и решения проблем. Это делает его более доступным и помогает разработчикам быстрее и эффективнее создавать сложные веб-приложения.

В заключение, JavaScript предлагает огромный функционал, простой и понятный синтаксис, мощные функции, легкость подключения к HTML и множество других преимуществ. Он является неотъемлемой частью веб-разработки и одним из ключевых инструментов для создания современных и интерактивных веб-приложений.

## 1.4 PHP

PHP (Hypertext Preprocessor) – это широко используемый скриптовый язык программирования, который обычно используется для разработки веб-приложений и создания динамических веб-страниц. PHP предоставляет множество возможностей и преимуществ, делая его одним из самых популярных языков программирования веб-разработки.

Одной из основных возможностей PHP является интеграция с различными базами данных, включая MySQL. Благодаря этому, PHP может легко взаимодействовать с базами данных, выполнять запросы и обрабатывать полученные результаты. Это делает его полезным инструментом для создания динамических сайтов с возможностью хранения и извлечения данных.

Синтаксис PHP достаточно прост и интуитивно понятен. Он напоминает синтаксис других языков программирования, таких как C или Java, что делает его доступным и легким для изучения для новичков. В PHP также имеется множество функций и методов, которые позволяют разработчикам решать различные задачи, такие как обработка форм, валидация данных, работа с файлами и многое другое.

PHP также включает в себя простые и удобные средства для подключения к базам данных MySQL. Существует множество расширений и классов, которые облегчают работу с MySQL, позволяя выполнять запросы, получать и обрабатывать результаты, а также обеспечивать безопасность данных. PHP также поддерживает другие популярные системы управления базами данных, такие как PostgreSQL, SQLite, Oracle и др.

PHP является мощным и гибким инструментом для разработки веб-приложений. Он обладает широким спектром возможностей, простым синтаксисом, множеством функций и преимуществами, что делает его идеальным выбором для разработчиков, занимающихся созданием динамических веб-приложений с подключением базы данных MySQL.

## MySQL

MySQL (структурированный язык запросов) - это одна из самых популярных систем управления базами данных, которая предоставляет множество возможностей и функций для эффективного хранения и обработки данных.

Система MySQL имеет широкий функционал, который позволяет разработчикам создавать сложные базы данных и производить множество операций с данными. Вот некоторые из главных возможностей и функций MySQL:

1. Масштабируемость: MySQL может обрабатывать большие объемы данных благодаря возможности добавления множества серверов в кластер. Это позволяет распределить нагрузку и обеспечить высокую доступность данных.

2. Поддержка различных типов данных: MySQL поддерживает различные типы данных, такие как числа, строки, даты, времена, изображения и т.д. Это позволяет пользователям хранить и обрабатывать разнообразную информацию.

3. Индексы и оптимизация запросов: MySQL предоставляет инструменты для создания индексов на данных, что ускоряет процесс поиска и сортировки. Он также предоставляет возможности для оптимизации выполнения запросов, что повышает производительность системы.

4. Транзакции и целостность данных: MySQL поддерживает транзакции, что позволяет гарантировать целостность данных при выполнении нескольких операций. Это важно для поддержания согласованности базы данных, особенно в многопользовательских средах.

5. Безопасность данных: MySQL обеспечивает множество функций для защиты данных, включая возможность устанавливать различные уровни доступа к данным, шифрование информации и проверку подлинности пользователей.

6. Репликация данных: MySQL позволяет создавать репликации данных, что обеспечивает отказоустойчивость и увеличивает доступность данных. Репликация позволяет иметь несколько копий базы данных для обработки запросов и распределения нагрузки.

7. Гибкость и простота в использовании: MySQL обладает простым и интуитивно понятным пользовательским интерфейсом, что упрощает создание и управление базами данных. Он также поддерживает большое количество языков программирования, что делает его универсальным инструментом для разработчиков.

Одним из ключевых преимуществ MySQL является его открытость и бесплатность, что позволяет широко использовать его в коммерческих и не коммерческих проектах. Большое сообщество разработчиков MySQL также предоставляет постоянную поддержку и обновления системы.

Хранение данных в MySQL осуществляется в таблицах, которые состоят из строк и столбцов. Данные хранятся в определенном формате, в зависимости от типа данных. MySQL также поддерживает возможность создания связей между таблицами, что позволяет объединять и анализировать данные из нескольких источников.

Обработка запросов в MySQL осуществляется с помощью языка SQL. Пользователи могут выполнять различные операции с данными, такие как выборка, вставка, обновление, удаление и другие. Система оптимизирует выполнение запросов, чтобы обеспечить максимальную производительность.

В целом, MySQL - это мощная и надежная система управления базами данных, которая предоставляет широкие возможности и функционал для эффективной работы с данными. Ее гибкость, простота в использовании и открытость делают ее привлекательным выбором для разработчиков и организаций.

# 2 Практическая часть

## 2.1 Требования

Как уточнялось ранее, курсовой проект предполагает создание web-приложение на тему «Кино», используя технологии HTML, CSS, Javascript, PHP, MySQL. Требуется реализовать добавление, редактирование, просмотр и удаление пользователей. Также предусмотреть функции авторизации и аутентификации, с учётом ролей пользователей.

**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 Mamp и создание базы данных

Установка MAMP позволяет воспроизводиться web-приложению локально. С помощью встроенного функционала (Рисунок 1) есть возможность перейти на *WebStart page и во вкладке phpMyAdmin* (Рисунок 2) и создать таблицы *user* (Рисунок 3) и *films* (Рисунок 4) базу данных с кодировкой *utf8\_general\_ci .*

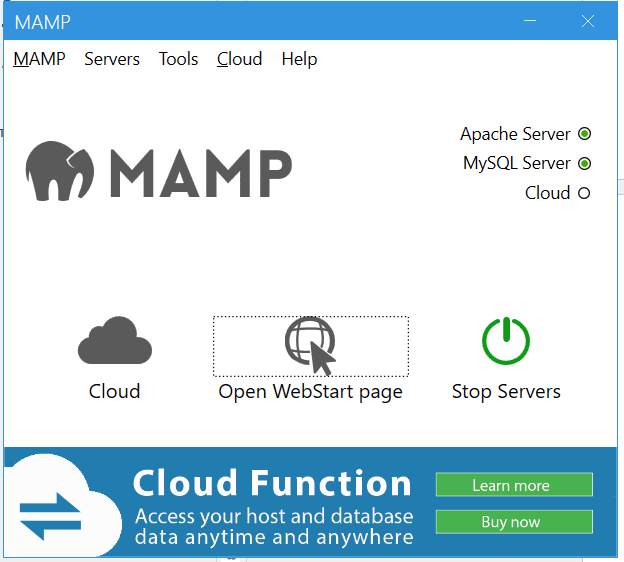


Рисунок 1 — MAMP: встроенный функционал

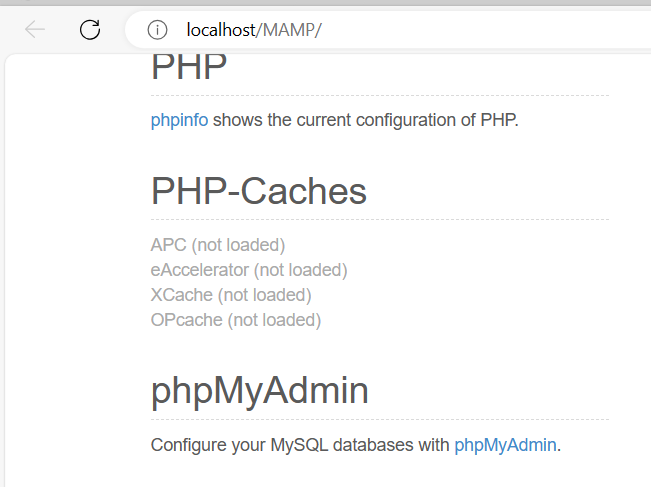


Рисунок 2 — MAMP / phpMyAdmin

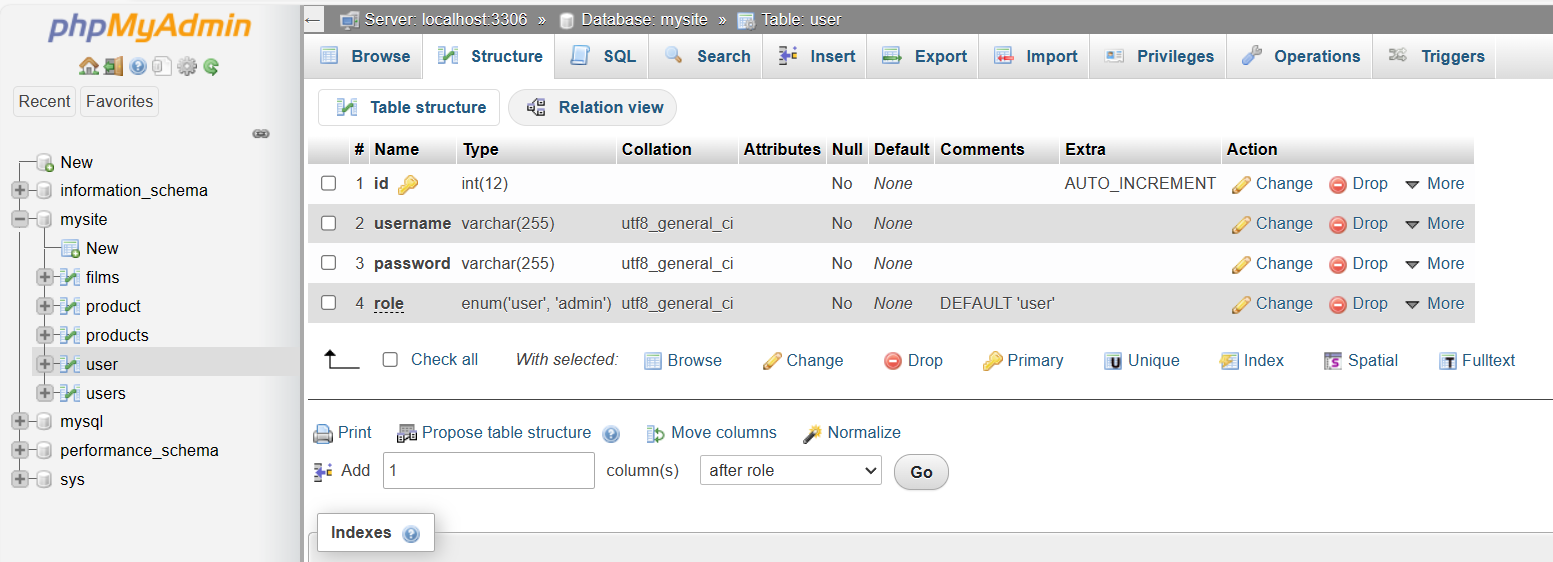


Рисунок 3 — Создание базы данных *mysite* и таблицы *user*

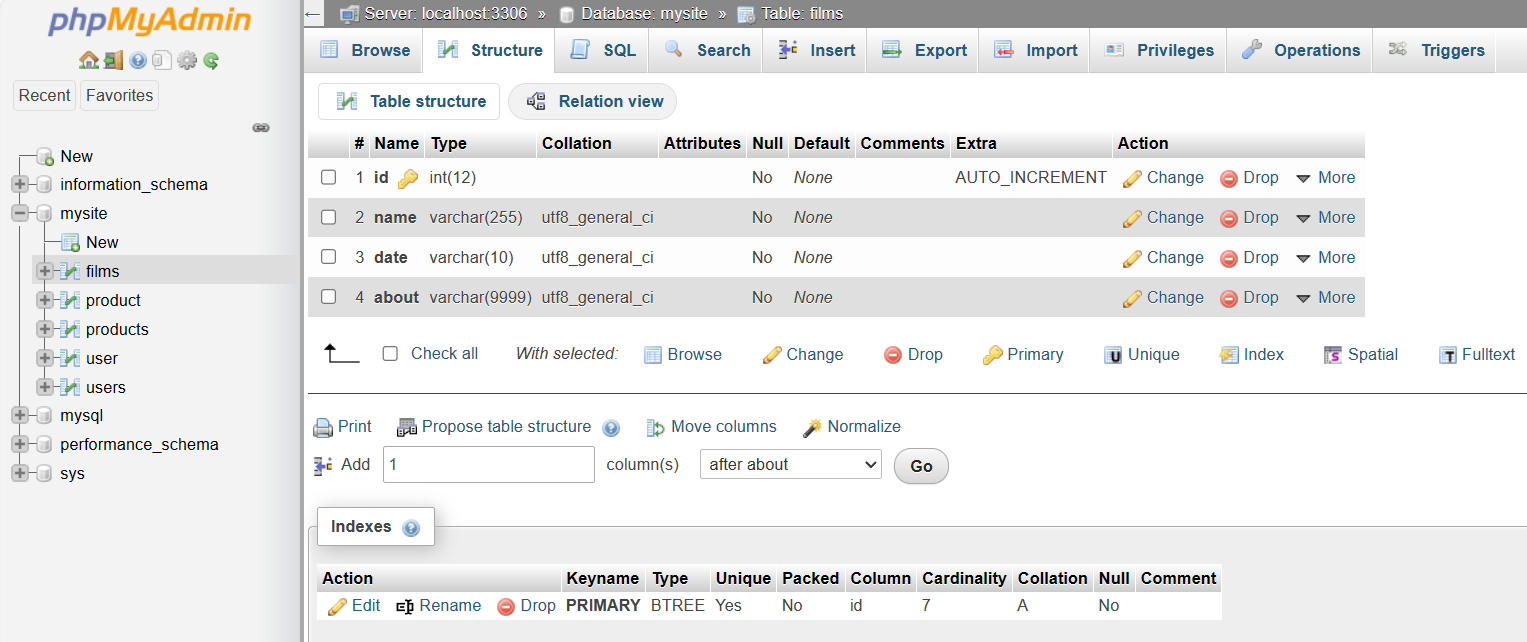


Рисунок 4 — Создание базы данных *mysite* и таблицы *films*

Создаётся таблица *user* с полями id, username, password и role. В неё попадают данные при регистрации автоматически, а поле role приминает значение user (с ограниченными правами доступа) по умолчанию. Только в базе данных можно изменить роль пользователя на admin в ручную. Далее создаётся таблица фильмов, которая содержит в себе id, name, date и about. В ней описаны название фильма, дата выхода и описание.

## 2.3 Регистрация и авторизация в web-приложении

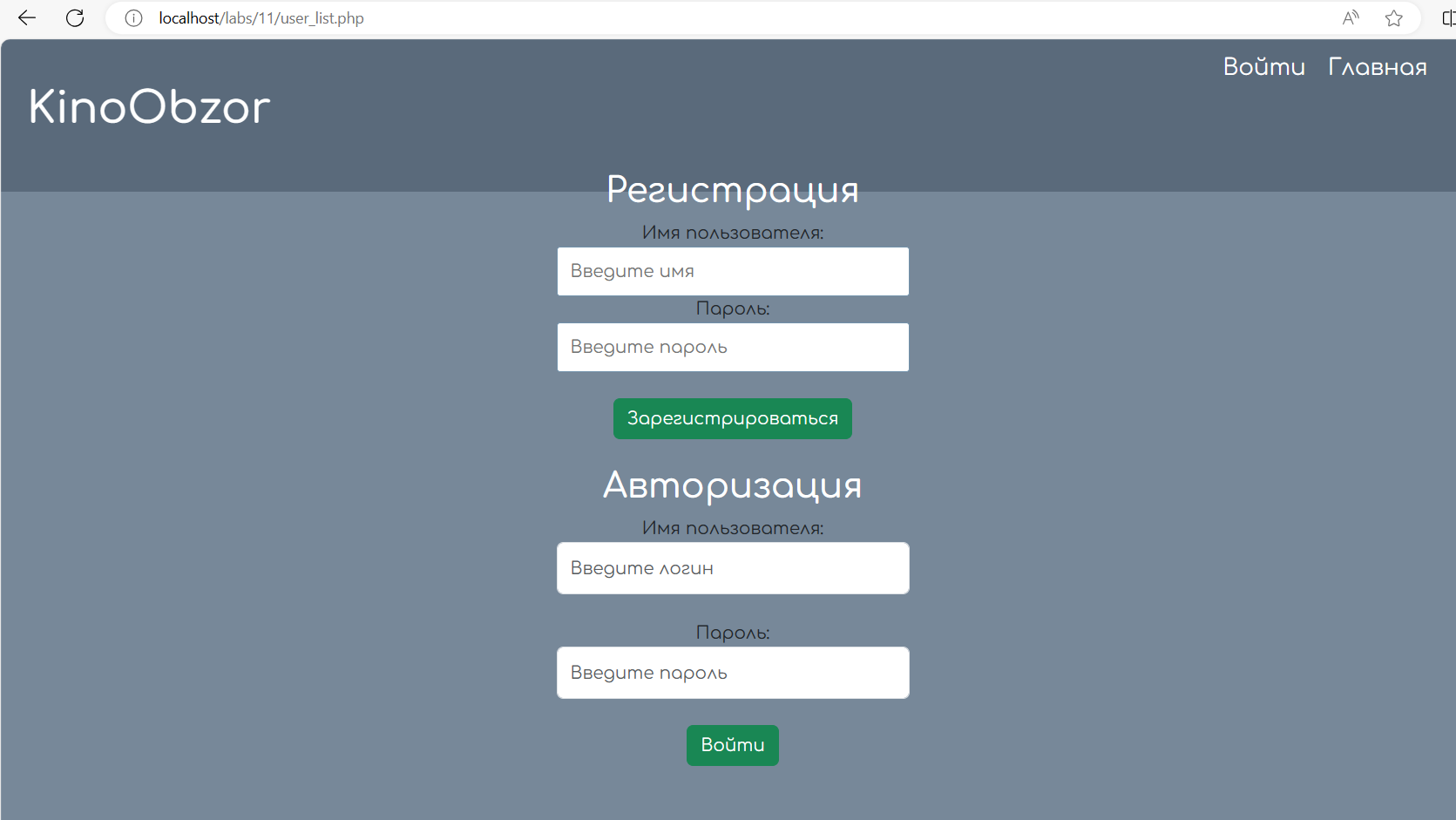


Рисунок 5 — Регистрация в web-приложении

Листинг HTML составляющей панели регистрации:

<!DOCTYPE html>

<html lang="ru">

  <head>

    <meta charset="utf-8">

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.rtl.min.css" integrity="sha384-nU14brUcp6StFntEOOEBvcJm4huWjB0OcIeQ3fltAfSmuZFrkAif0T+UtNGlKKQv" crossorigin="anonymous">

    <link rel="stylesheet" href="style.css">

    <link rel="preconnect" href="https://fonts.googleapis.com">

    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

    <link href="https://fonts.googleapis.com/css2?family=Comfortaa:wght@300;400;500;600;700&display=swap" rel="stylesheet">

    <title>KinoObzor</title>

  </head>

  <body>

    <header class="container-fluid">

<div class="container">

<div class="row">

    <div class="col-4">

    <h1> <a href="/">KinoObzor</a></h1>

    </div>

    <nav class="col-8">

        <ul>

            <li><a href="idea.php">Главная</a></li>

            <li><?php

                session\_start(); // начинаем сессию

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

                    // пользователь авторизован

                    echo '<a href="dashboard.php">Личный кабинет</a>';

                } else {

                    // пользователь не авторизован

                    echo '<a href="user\_list.php">Войти</a>';

                }?>

            </li>

        </ul>

    </nav>

</div>

<div class="container2 mt-4 col-md-auto">

    <div class="container2 mt-4 col-md-auto">

        <h2>Регистрация</h2>

        <form action="register.php" method="post">

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

            <input type="text" id="username" placeholder="Введите имя" name="username" required>

            <label for="password">Пароль:</label>

            <input type="password" id="password" placeholder="Введите пароль" name="password" required> <br>

            <button class="btn btn-success" type="submit">Зарегистрироваться</button>

        </form>

    </div>

    <div class="container2 mt-4 col-md-auto">

        <h2>Авторизация</h2>

        <form class="reg\_form" action="login.php" method="post">

            <label for="login\_username">Имя пользователя:</label>

            <input class="form-control" type="text" placeholder="Введите логин" id="login\_username" name="login\_username" required>

            <br>

            <label for="login\_password">Пароль:</label>

            <input class="form-control" type="password" placeholder="Введите пароль" id="login\_password" name="login\_password" required>

            <br>

            <button class="btn btn-success" type="submit">Войти</button>

        </form>

    </div>

</div>

</body>

</html>

Форма регистрации содержит в себе поля для ввода имени и пароля, а также кнопки «зарегистрироваться». Если пользователь уже зарегистрирован, то на этой же странице есть форма авторизации.

При нажатии на кнопку «зарегистрироваться» пользователь попадает в базу данных. Эту форму регистрации обрабатывает файл *register.php*. Листинг:

<?php

require\_once('db.php');

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

    $username = $\_POST['username'];

    $password = password\_hash($\_POST['password'], PASSWORD\_DEFAULT);

    $sql = "INSERT INTO user (username, password, role) VALUES ('$username', '$password', 'user')";

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

        header("Location: user\_list.php");

    } else {

        echo "Ошибка: " . $sql . "<br>" . $conn->error;

    }

}

$conn->close();

?>

Также, следует отметить, что пароль сразу хешируется для безопасности пользователей (Рисунок 6).

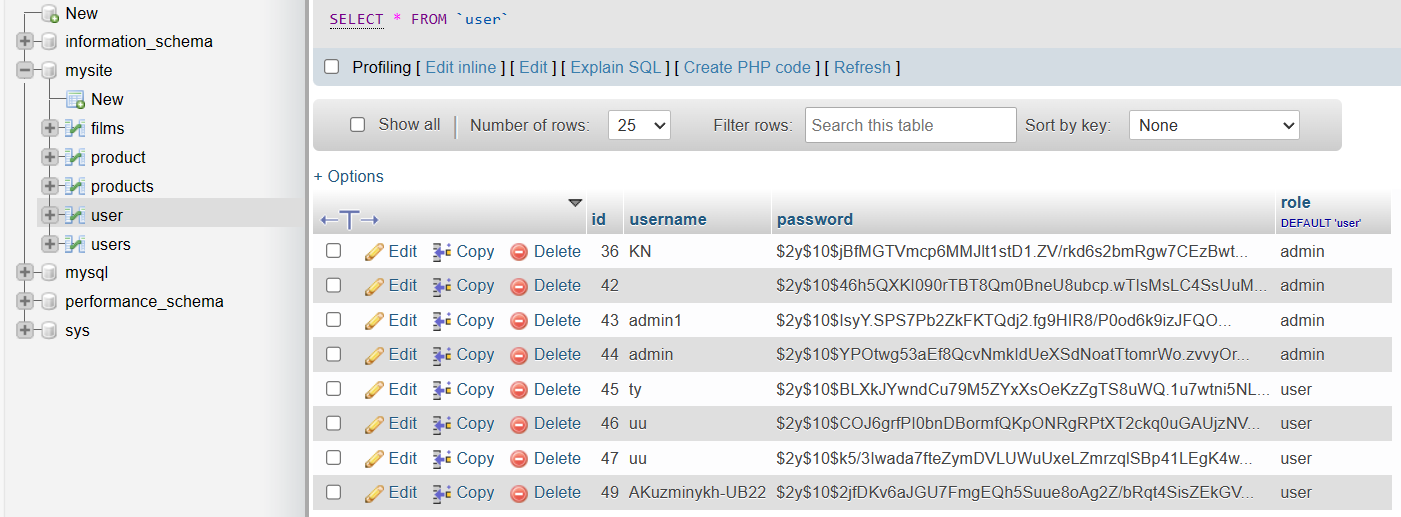


Рисунок 6 — Сохранение регистрации пользователя в базе данных, хеширование пароля

После обновления страницы он может ввести те же логин и пароль в форме авторизации и попасть на станицу *Личный кабинет.* Листинг:

<?php

session\_start();

require\_once('db.php');

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

    $login\_username = $\_POST['login\_username'];

    $login\_password = $\_POST['login\_password'];

    $sql = "SELECT \* FROM user WHERE username='$login\_username'";

    $result = $conn->query($sql);

    if ($result->num\_rows > 0) {

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

        if (password\_verify($login\_password, $row['password'])) {

            $\_SESSION['user\_id'] = $row['id'];

            header("Location: dashboard.php");

        } else {

            echo "Неверный пароль";

        }

    } else {

        echo "Пользователь не найден";

    }

}

$conn->close();

?>

Если пароль введён неверно, то на экране появляется соответствующее сообщение (Рисунок 7).

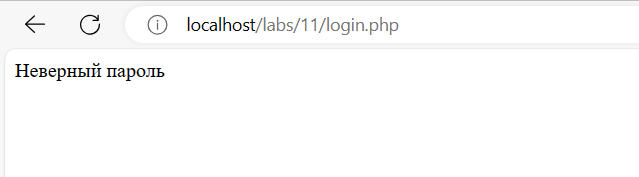


Рисунок 7 – Сообщение об ошибке

Подключение к базе данных вынесено в отдельный файл bd.php. Листинг:

<?php

$servername = "localhost";

$username = "root";

$password = "root";

$dbname = "mysite";

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

if ($conn->connect\_error) {

    die("Ошибка соединения: " . $conn->connect\_error);

}

?>

## 2.4 Личный кабинет пользователя в web-приложении

## 2.4.1 Функционал User

После авторизации user попадает на станицу *Личный кабинет* (Рисунок 8), где он может изменить свои личные данные.

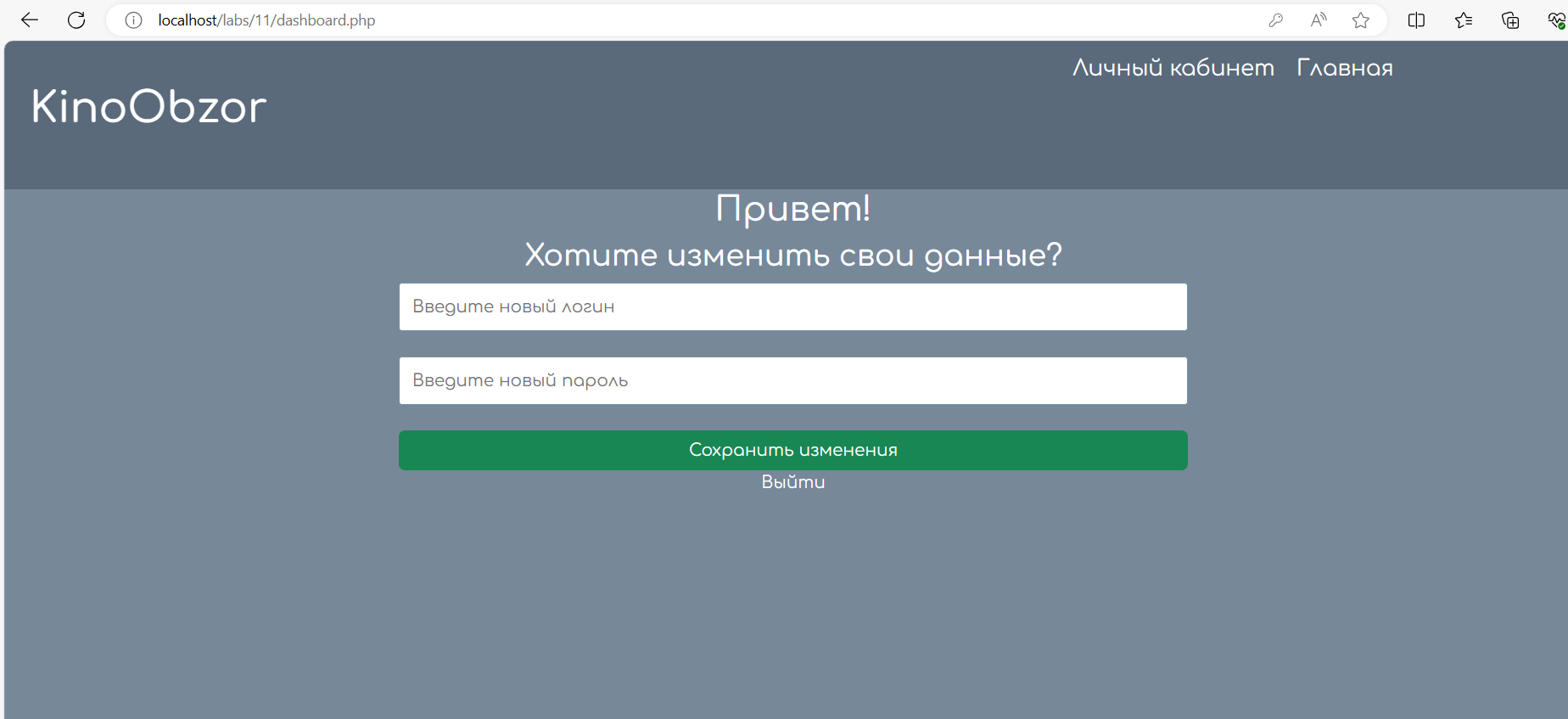


Рисунок 8 — Личный кабинет пользователя в web-приложении

Листинг html-страницы личного кабинета User:

<!doctype html>

<html lang="ru">

  <head>

    <meta charset="utf-8">

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.rtl.min.css" integrity="sha384-nU14brUcp6StFntEOOEBvcJm4huWjB0OcIeQ3fltAfSmuZFrkAif0T+UtNGlKKQv" crossorigin="anonymous">

    <link rel="stylesheet" href="style.css">

    <link rel="preconnect" href="https://fonts.googleapis.com">

    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

    <link href="https://fonts.googleapis.com/css2?family=Comfortaa:wght@300;400;500;600;700&display=swap" rel="stylesheet">

    <title>KinoObzor</title>

  </head>

  <body >

    <header class="container-fluid">

<div class="container">

    <div class="row">

        <div class="col-4">

        <h1> <a href="/">KinoObzor</a></h1>

        </div>

        <nav class="col-8">

            <ul>

                <li><a href="idea.php">Главная</a></li>

                <li> <?php

                    session\_start(); // начинаем сессию

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

                        // пользователь авторизован

                        echo '<a href="dashboard.php">Личный кабинет</a>';

                    } else {

                        // пользователь не авторизован

                        echo '<a href="user\_list.php">Войти</a>';

                    }

                    ?>

                </li>

            </ul>

        </nav>

    </div>

</div>

</header>

    <div class="container2 mt-9 col-md-auto">

        <h2>Привет!</h2>

        <h3>Хотите изменить свои данные?</h3>

        <form action="edit\_user.php" method="post">

            <input type="text" id="newUsername" placeholder="Введите новый логин" name="newUsername"><br>

            <input type="password" id="newPassword" placeholder="Введите новый пароль" name="newPassword"><br>

            <input type="submit" class="btn btn-success" value="Сохранить изменения">

            <a class="ah" href="logout.php">Выйти</a>

        </form>

    </div>

    <div>

    <script>

////////////////////////////////////

    </script>

     </div>

 </div>

</body>

</html>

С помощью PHP кода и сессии на проверку роли пользователя отображается только форма для изменения его данных. На web-странице есть кнопка «сохранить», при нажатии на которую с помощью файла edit\_user.php новые логин и пароль попадают в базу данных user. Листинг edit\_user.php:

<?php

// Проверка авторизации пользователя

session\_start();

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

  header('Location: dashboard.php'); // Перенаправление на страницу авторизации, если пользователь не авторизован

  exit();

}

// Подключение к базе данных

$servername = "localhost";

$username = "root";

$password = "root";

$dbname = "mysite";

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

if ($conn->connect\_error) {

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

}

// Обработка формы изменения данных

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

  $newUsername = $\_POST['newUsername'];

  $newPassword = $\_POST['newPassword'];

  // Защита от SQL-инъекций

  $newPassword = password\_hash(mysqli\_real\_escape\_string($conn, $newPassword), PASSWORD\_DEFAULT);

  // Обновление данных в базе данных

  $sql = "UPDATE user SET username='$newUsername', password='$newPassword' WHERE id=" . $\_SESSION['user\_id'];

  if ($conn->query($sql) === TRUE) {

    echo "Данные успешно обновлены";

  } else {

    echo "Ошибка при обновлении данных: " . $conn->error;

  }

}

$conn->close();

?>

При нажатии на кнопку «Сохранить изменения» на странице выводится сообщение об успешной смене данных или об ошибке (Рисунок 9).

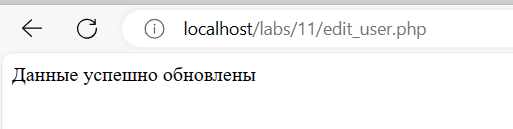


Рисунок 9 - Сообщение об успешной смене данных

На сайте также представлена возможность выйти из личного кабинета, которая осуществляется с помощью кнопки “Выйти” и файла logout.php. Листинг logout.php :

<?php

session\_start();

session\_destroy();

header("Location: user\_list.php");

?>

Функционал обычного пользователя закончен, далее будет представлен функционал администратора.

## 2.4.2 Функционал Аdmin

С помощью PHP кода проверяется, соответствует ли роль зарегистрированного пользователя роли администратора. При выполнении данного условия на панель администратора выводится его личный кабинет с дополнительными ссылками на базы данных (Рисунок 10).

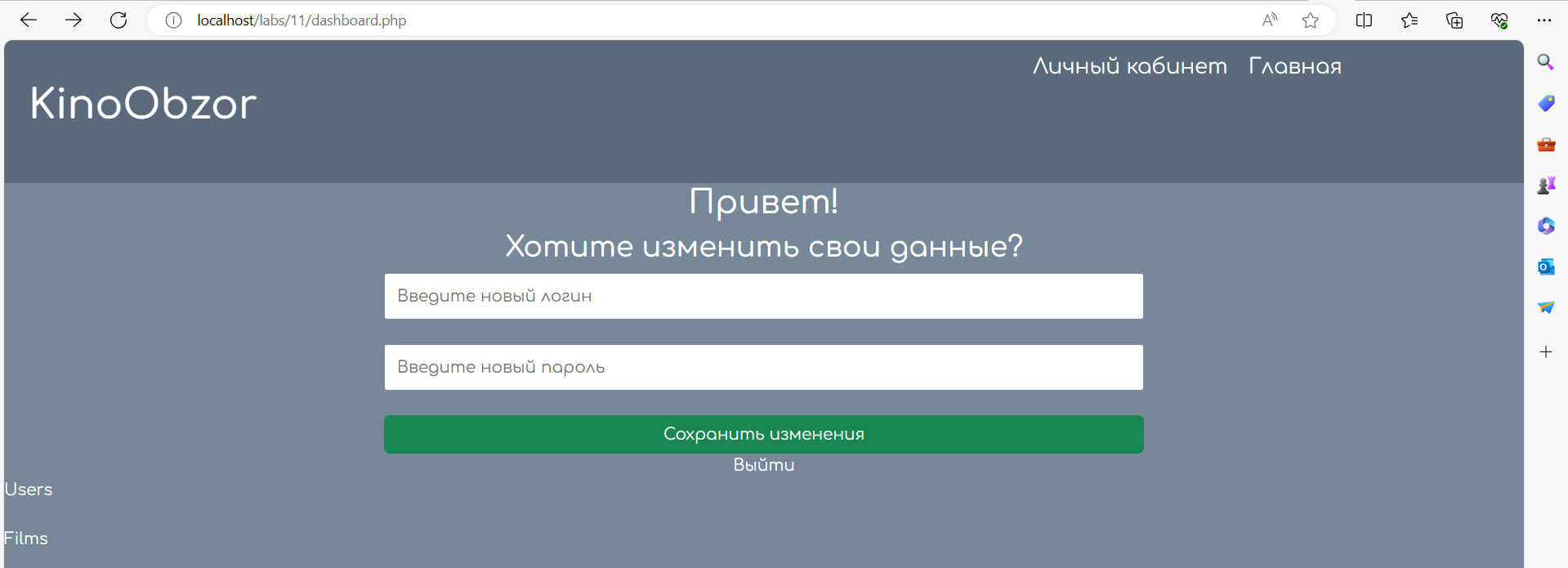


Рисунок 10 - Личный кабинет администратора в web-приложении

Появились ссылки на два раздела, в которых администратор может править информацию о пользователях и фильмах.

Страница users.php имеет следующий вид (Рисунок 11).

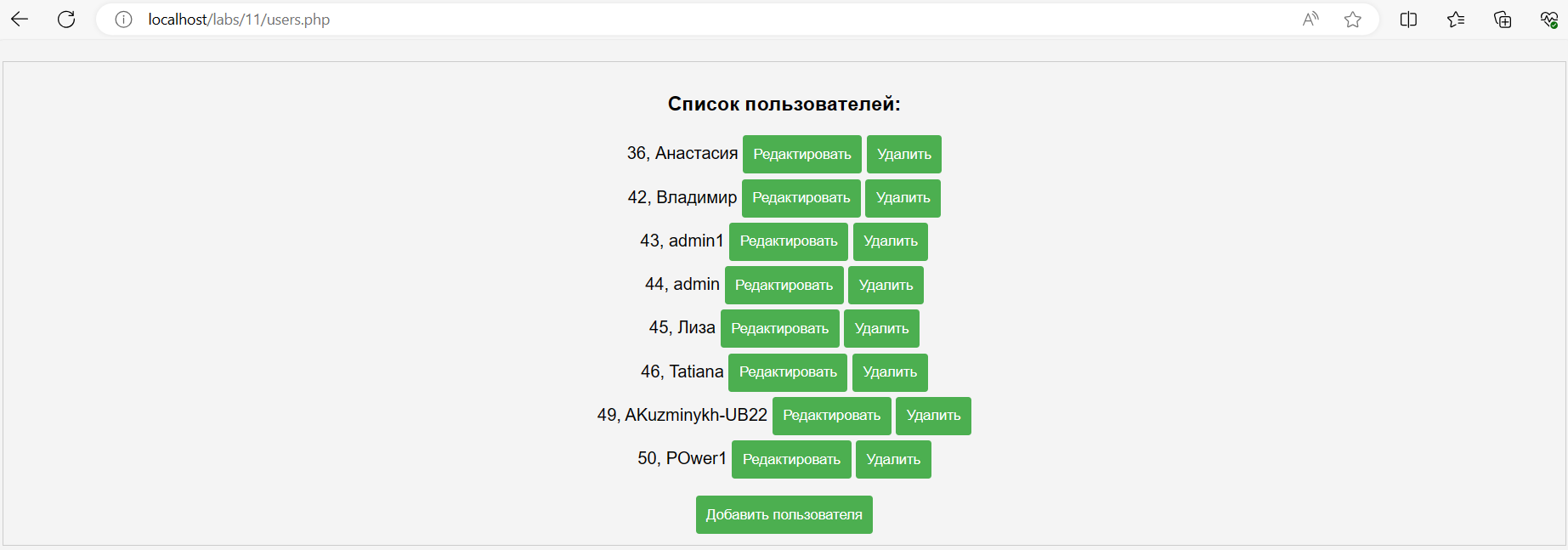


Рисунок 11 - users.php

Редактирование пользователей реализовано через функцию editUser на языке Java Script

Листинг:

 <script>

        function editUser(userId) {

            document.getElementById("edit\_user\_id").value = userId;

            document.getElementById("new\_username").value = prompt("Введите новый логин:", "");

            document.getElementById("new\_password").value = prompt("Введите новый пароль:", "");

            document.forms[0].submit();

        }

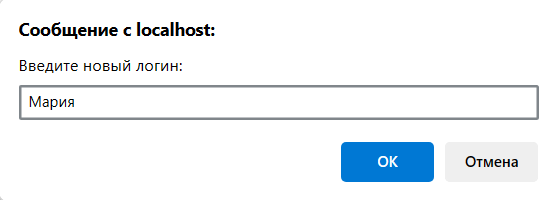


Рисунок 12 - Изменение имени пользователя с id=43

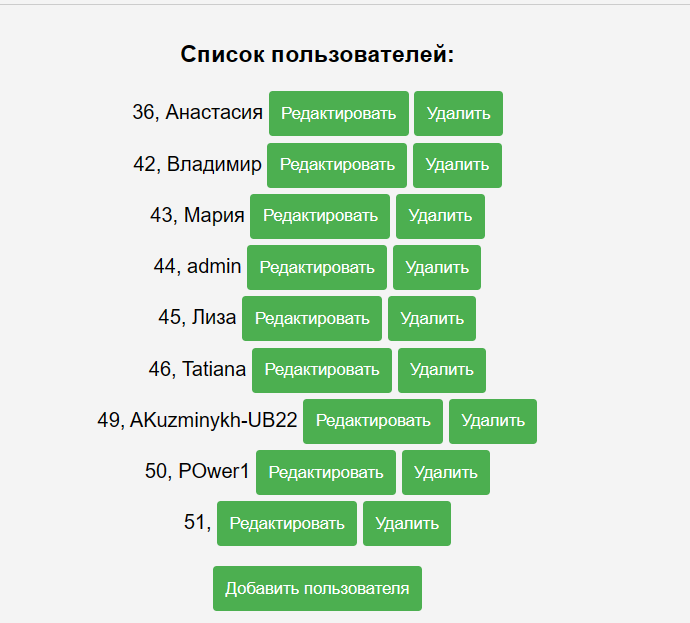


Рисунок 13 - Вывод изменённых данных пользователя с id=43

Чтобы удалить пользователя, на сайте необходимо нажать кнопку “Удалить”, так сработает function confirmDelete(userId). Метод получает в переменную userId его ID и запустится файл delet\_user.php для удаления пользователя.

Листинг script в личном кабинете users.php:

function confirmDelete(userId) {

            var confirmDelete = confirm("Удалить пользователя?");

            if (confirmDelete) {

                var deleteForm = document.createElement("form");

                deleteForm.method = "post";

                deleteForm.action = "delete\_user.php";

                var input = document.createElement("input");

                input.type = "hidden";

                input.name = "user\_id";

                input.value = userId;

                deleteForm.appendChild(input);

                document.body.appendChild(deleteForm);

                deleteForm.submit();

            }

        }

Листинг delete\_user.php:

<?php

require\_once('db.php');

if ($\_SERVER["REQUEST\_METHOD"] === "POST" && isset($\_POST['user\_id'])) {

    $userToDelete = $\_POST['user\_id'];

    $sql = "DELETE FROM user WHERE id = $userToDelete";

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

        header("Location: dashboard.php");

        exit();

    } else {

        echo "Ошибка удаления пользователя: " . $conn->error;

    }

} else {

    echo "Ошибка";

}

$conn->close();

?>

Также было реализовано добавление пользователя с помощью function showAddUserForm() через script в users.php:

Листинг script в личном кабинете users.php:

  function showAddUserForm() {

            var username = prompt("Введите новый логин:", "");

            var password = prompt("Введите новый пароль:", "");

            var addUserForm = document.createElement("form");

            addUserForm.method = "post";

            addUserForm.action = "add\_user.php";

            var usernameInput = document.createElement("input");

            usernameInput.type = "text";

            usernameInput.name = "username";

            usernameInput.value = username;

            var passwordInput = document.createElement("input");

            passwordInput.type = "password";

            passwordInput.name = "new\_password";

            passwordInput.value = password;

            var addButton = document.createElement("button");

            addButton.type = "submit";

            addButton.textContent = "Add User";

            addUserForm.appendChild(usernameInput);

            addUserForm.appendChild(passwordInput);

            addUserForm.appendChild(addButton);

            document.body.appendChild(addUserForm);

            addUserForm.submit();

        }

    </script>

Листинг add\_user.php:

<?php

require\_once('db.php');

if ($\_SERVER["REQUEST\_METHOD"] === "POST" && isset($\_POST['username']) && isset($\_POST['new\_password'])) {

    $newUsername = $\_POST['username'];

    $newPassword = password\_hash($\_POST['new\_password'], PASSWORD\_DEFAULT);

    $sql = "INSERT INTO user (username, password) VALUES ('$newUsername', '$newPassword')";

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

        header("Location: dashboard.php");

        exit();

    } else {

        echo "Не удалось добавить нового пользователя: " . $conn->error;

    }

} else {

    echo "Ошибка";

}

$conn->close();

?>

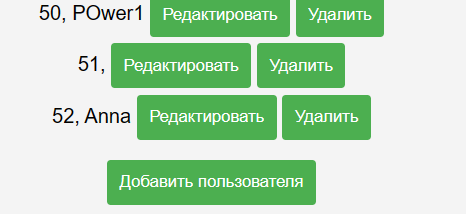


Рисунок 14 – Добавлен пользователь Anna, id=52

Полный листинг страницы users.php:

<?php

session\_start();

require\_once('db.php');

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

    header("Location: user\_list.html");

    exit();

}

$user\_id = $\_SESSION['user\_id'];

$sql\_role = "SELECT role FROM user WHERE id=$user\_id";

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

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

    $user\_role = $result\_role->fetch\_assoc()['role'];

    if ($user\_role === 'admin') {

        $sql\_users = "SELECT id, username FROM user";

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

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

            echo "<div class='admin-panel'>";

            echo "<h3>Список пользователей:</h3>";

            echo "<ul>";

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

                echo "<li> " . $row['id'] . ",  " . $row['username'] . "

                      <button onclick='editUser(" . $row['id'] . ")'>Редактировать</button>

                      <button onclick='confirmDelete(" . $row['id'] . ")'>Удалить</button></li>";

            }

            echo "</ul>";

            echo "<button onclick='showAddUserForm()'>Добавить пользователя</button>";

            echo "</div>";

        } else {

            echo "Нет зарегистрированных пользователей.";

        }

    }

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

    $result = $conn->query($sql);

    if ($result->num\_rows > 0) {

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

        echo "<h2> </h2>";

    } else {

        echo "Нет пользователя";

    }

} else {

    echo "Не удалось определить роль пользователя";

}

$conn->close();

?>

<!DOCTYPE html>

<html lang="ru">

<head>

    <meta charset="UTF-8">

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

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

    <title>Пользователи</title>

</head>

<body>

    <script>

        function editUser(userId) {

            document.getElementById("edit\_user\_id").value = userId;

            document.getElementById("new\_username").value = prompt("Введите новый логин:", "");

            document.getElementById("new\_password").value = prompt("Введите новый пароль:", "");

            document.forms[0].submit();

        }

        function confirmDelete(userId) {

            var confirmDelete = confirm("Удалить пользователя?");

            if (confirmDelete) {

                var deleteForm = document.createElement("form");

                deleteForm.method = "post";

                deleteForm.action = "delete\_user.php";

                var input = document.createElement("input");

                input.type = "hidden";

                input.name = "user\_id";

                input.value = userId;

                deleteForm.appendChild(input);

                document.body.appendChild(deleteForm);

                deleteForm.submit();

            }

        }

        function showAddUserForm() {

            var username = prompt("Введите новый логин:", "");

            var password = prompt("Введите новый пароль:", "");

            var addUserForm = document.createElement("form");

            addUserForm.method = "post";

            addUserForm.action = "add\_user.php";

            var usernameInput = document.createElement("input");

            usernameInput.type = "text";

            usernameInput.name = "username";

            usernameInput.value = username;

            var passwordInput = document.createElement("input");

            passwordInput.type = "password";

            passwordInput.name = "new\_password";

            passwordInput.value = password;

            var addButton = document.createElement("button");

            addButton.type = "submit";

            addButton.textContent = "Add User";

            addUserForm.appendChild(usernameInput);

            addUserForm.appendChild(passwordInput);

            addUserForm.appendChild(addButton);

            document.body.appendChild(addUserForm);

            addUserForm.submit();

        }

    </script>

</body>

</html>

По такому же принципу устроена работа добавления фильмов, редактирования и удаления.

Страница films.php имеет следующий вид (Рисунок 15).

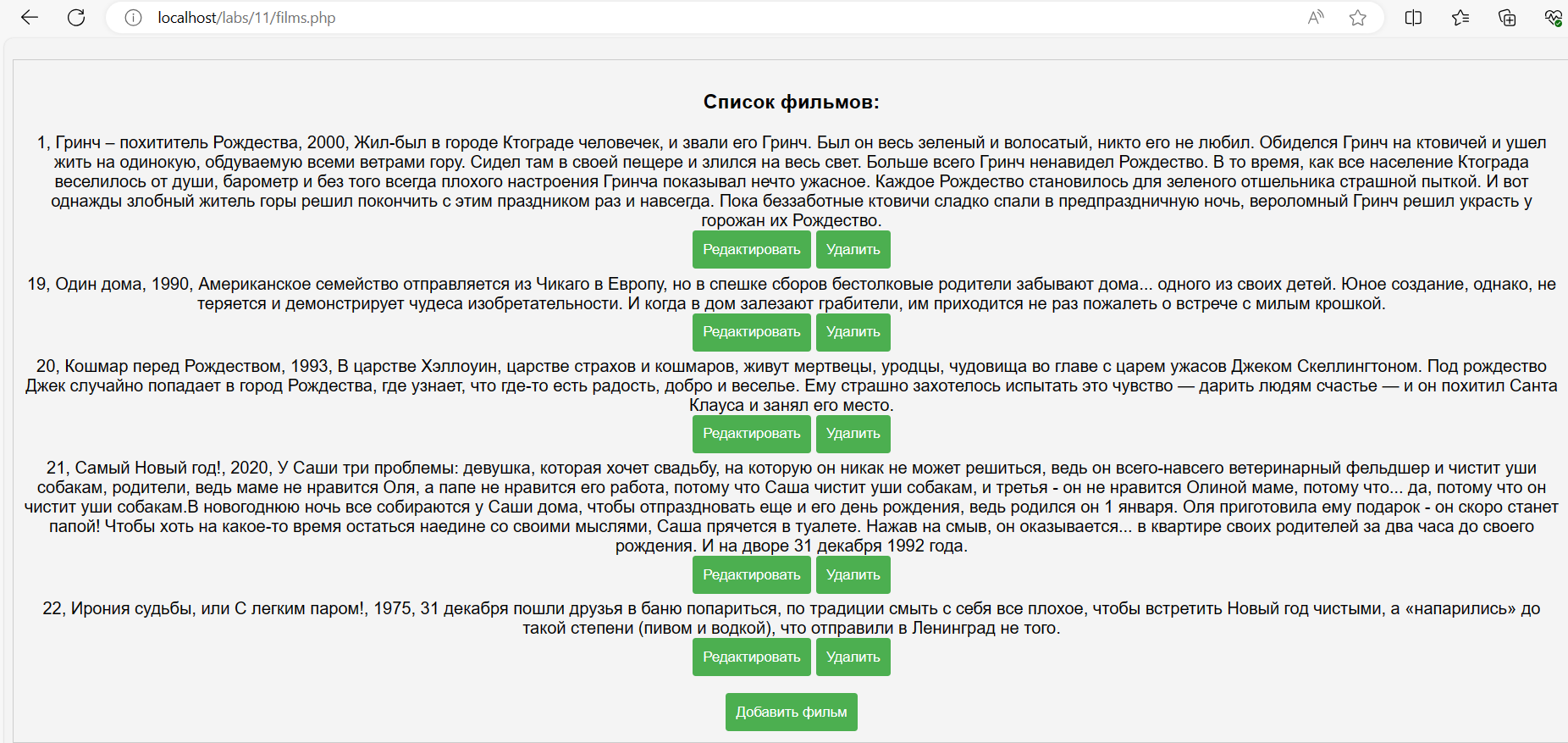


Рисунок 15 - films.php

Редактирование фильмов реализовано через функцию editFilms на языке Java Script

Листинг:

   <script>

        function editFilms(filmsId) {

            document.getElementById("edit\_films\_id").value = filmsId;

            document.getElementById("new\_name").value = prompt("Введите новое название:", "");

            document.getElementById("new\_date").value = prompt("Введите новую дату", "");

            document.getElementById("new\_about").value = prompt("Введите новое описание:", "");

            document.forms[0].submit();

        }

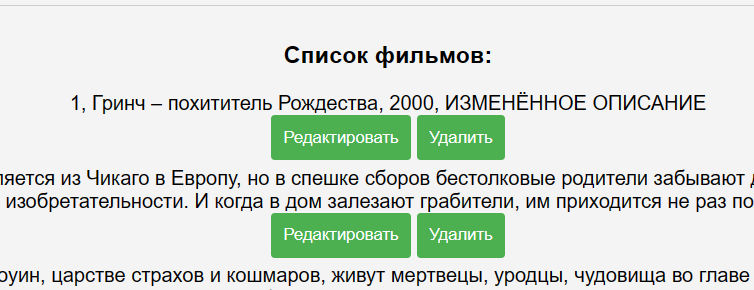


Рисунок 16 - Вывод изменённых данных фильма с id=1

Чтобы удалить фильм, на сайте необходимо нажать кнопку “Удалить”, так сработает function confirmDelete(filmsId). Метод получает в переменную filmsId его ID и запустится файл delet\_film.php для удаления фильма.

Листинг script в films.php:

function confirmDelete(filmsId) {

            var confirmDelete = confirm("Удалить фильм?");

            if (confirmDelete) {

                var deleteForm = document.createElement("form");

                deleteForm.method = "post";

                deleteForm.action = "delete\_film.php";

                var input = document.createElement("input");

                input.type = "hidden";

                input.name = "films\_id";

                input.value = filmsId;

                deleteForm.appendChild(input);

                document.body.appendChild(deleteForm);

                deleteForm.submit();

            }

        }

Листинг delete\_film.php:

<?php

require\_once('db.php');

if ($\_SERVER["REQUEST\_METHOD"] === "POST" && isset($\_POST['films\_id'])) {

    $filmsToDelete = $\_POST['films\_id'];

    $sql = "DELETE FROM films WHERE id = $filmsToDelete";

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

        header("Location: films.php");

        exit();

    } else {

        echo "Ошибка удаления фильма: " . $conn->error;

    }

} else {

    echo "Ошибка";

}

$conn->close();

?>

Также было реализовано добавление фильма с помощью function showAddFilmsForm() через script в films.php:

Листинг script в films.php:

  function showAddFilmsForm() {

           var name = prompt("Введите новое название:", "");

           var date = prompt("Введите новую дату:", "");

           var about = prompt("Введите новое описание:", "");

           var addFilmsForm = document.createElement("form");

           addFilmsForm.method = "post";

           addFilmsForm.action = "add\_film.php";

           var nameInput = document.createElement("input");

           nameInput.type = "text";

           nameInput.name = "new\_name";

           nameInput.value = name;

           var dateInput = document.createElement("input");

           dateInput.type = "text";

           dateInput.name = "new\_date";

           dateInput.value = date;

           var aboutInput = document.createElement("input");

           aboutInput.type = "text";

           aboutInput.name = "new\_about";

           aboutInput.value = about;

           var addButton = document.createElement("button");

           addButton.type = "submit";

           addButton.textContent = "Add Films";

           addFilmsForm.appendChild(nameInput);

           addFilmsForm.appendChild(dateInput);

           addFilmsForm.appendChild(aboutInput);

           addFilmsForm.appendChild(addButton);

           document.body.appendChild(addFilmsForm);

           addFilmsForm.submit();

        }

    </script>

Листинг add\_film.php:

<?php

require\_once('db.php');

if ($\_SERVER["REQUEST\_METHOD"] === "POST" && isset($\_POST['new\_name']) && isset($\_POST['new\_date']) && isset($\_POST['new\_about'])) {

    $name = $\_POST['new\_name'];

    $date = $\_POST['new\_date'];

    $about = $\_POST['new\_about'];

    $sql = "INSERT INTO films (name, date, about) VALUES ('$name', '$date', '$about')";

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

        header("Location: films.php");

        exit();

    } else {

        echo "Не удалось добавить новый фильм: " . $conn->error;

    }

} else {

    echo "Ошибка";

}

$conn->close();

?>

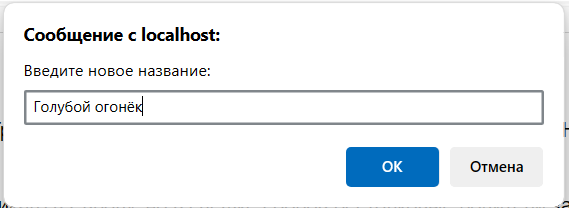


Рисунок 17 – Добавление нового фильма

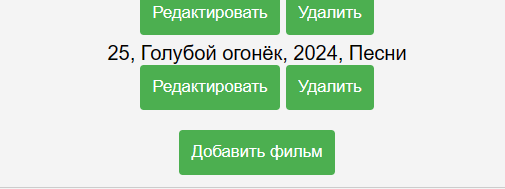


Рисунок 18 – Добавлен новый фильм с id=25

Полный листинг films.php:

<?php

session\_start();

require\_once('db.php');

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

    header("Location: user\_list.html");

    exit();

}

$user\_id = $\_SESSION['user\_id'];

$sql\_role = "SELECT role FROM user WHERE id=$user\_id";

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

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

    $user\_role = $result\_role->fetch\_assoc()['role'];

    if ($user\_role === 'admin') {

        $sql\_films = "SELECT id, name, date, about FROM films";

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

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

            echo "<div class='admin-panel'>";

            echo "<h3>Список фильмов:</h3>";

            echo "<ul>";

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

                echo "<li> " . $row['id'] . ",  " . $row['name'] . ",  " . $row['date'] . ",  " . $row['about'] . "   <br>

                      <button onclick='editFilms(" . $row['id'] . ")'>Редактировать</button>

                      <button onclick='confirmDelete(" . $row['id'] . ")'>Удалить</button></li>";

            }

            echo "</ul>";

            echo "<button onclick='showAddFilmsForm()'>Добавить фильм</button>";

            echo "</div>";

        } else {

            echo "Нет фильмов.";

        }

    }

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

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

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

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

                echo "<h2> </h2>";

            } else {

                echo "Нет пользователя";

            }

        } else {

            echo "Не удалось определить роль пользователя";

        }

        $conn->close();

?>

<!DOCTYPE html>

<html lang="ru">

<head>

    <meta charset="UTF-8">

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

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

    <title>Фильмы</title>

</head>

<body>

<body>

    <script>

        function editFilms(filmsId) {

            document.getElementById("edit\_films\_id").value = filmsId;

            document.getElementById("new\_name").value = prompt("Введите новое название:", "");

            document.getElementById("new\_date").value = prompt("Введите новую дату", "");

            document.getElementById("new\_about").value = prompt("Введите новое описание:", "");

            document.forms[0].submit();

        }

        function confirmDelete(filmsId) {

            var confirmDelete = confirm("Удалить фильм?");

            if (confirmDelete) {

                var deleteForm = document.createElement("form");

                deleteForm.method = "post";

                deleteForm.action = "delete\_film.php";

                var input = document.createElement("input");

                input.type = "hidden";

                input.name = "films\_id";

                input.value = filmsId;

                deleteForm.appendChild(input);

                document.body.appendChild(deleteForm);

                deleteForm.submit();

            }

        }

        function showAddFilmsForm() {

           var name = prompt("Введите новое название:", "");

           var date = prompt("Введите новую дату:", "");

           var about = prompt("Введите новое описание:", "");

           var addFilmsForm = document.createElement("form");

           addFilmsForm.method = "post";

           addFilmsForm.action = "add\_film.php";

           var nameInput = document.createElement("input");

           nameInput.type = "text";

           nameInput.name = "new\_name";

           nameInput.value = name;

           var dateInput = document.createElement("input");

           dateInput.type = "text";

           dateInput.name = "new\_date";

           dateInput.value = date;

           var aboutInput = document.createElement("input");

           aboutInput.type = "text";

           aboutInput.name = "new\_about";

           aboutInput.value = about;

           var addButton = document.createElement("button");

           addButton.type = "submit";

           addButton.textContent = "Add Films";

           addFilmsForm.appendChild(nameInput);

           addFilmsForm.appendChild(dateInput);

           addFilmsForm.appendChild(aboutInput);

           addFilmsForm.appendChild(addButton);

           document.body.appendChild(addFilmsForm);

           addFilmsForm.submit();

        }

    </script>

</body>

</html>

Выше был представлен функционал администратора.

## 2.5 Главная страница



Рисунок 19 — Главная страница, 1



Рисунок 20 — Главная страница, 2

На главной странице выводятся все фильмы с таблицы films базы данных mysite и представлены рекомендуемые фильмы, кликнув на которые откроется станица фильма (Рисунок 21) с возможностью просмотра кино.

Листинг главной страницы:

<?php

session\_start();

$servername = "localhost";

$username = "root";

$password = "root";

$dbname = "mysite";

// Создание соединения

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

?>

<!doctype html>

<html lang="ru">

  <head>

    <meta charset="utf-8">

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.rtl.min.css" integrity="sha384-nU14brUcp6StFntEOOEBvcJm4huWjB0OcIeQ3fltAfSmuZFrkAif0T+UtNGlKKQv" crossorigin="anonymous">

    <link rel="stylesheet" href="style.css">

    <link rel="preconnect" href="https://fonts.googleapis.com">

    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

    <link href="https://fonts.googleapis.com/css2?family=Comfortaa:wght@300;400;500;600;700&display=swap" rel="stylesheet">

    <title>KinoObzor</title>

  </head>

  <body>

    <header class="container-fluid">

<div class="container">

<div class="row">

    <div class="col-4">

    <h1> <a href="/">KinoObzor</a></h1>

    </div>

    <nav class="col-8">

        <ul>

            <li><a href="idea.php">Главная</a></li>

                <li> <?php

                session\_start(); // начинаем сессию

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

                    // пользователь авторизован

                    echo '<a href="dashboard.php">Личный кабинет</a>';

                } else {

                    // пользователь не авторизован

                    echo '<a href="user\_list.php">Войти</a>';

                }?>

            </li>

        </ul>

    </nav>

</div>

</div>

    </header>

<div class="container"></div>

<div class="row"><h2 class="slider-title">Новогоднее кино</h2></div>

</div>

    <div class="films">

            <div class="main-content col-12">

                <h2>Все фильмы</h2>

                <img src="гринч1.jpg" class="d-block w-100" >

                <ul>

            <?php

                $sql = "SELECT \* FROM films";

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

                ?>

            </ul>

            <table>

            <tr>

                <!-- <th>Pic</th> -->

                <th>Название</th>

                <th>Дата выхода</th>

                <th>О фильме</th>

            </tr>

            <?php foreach($result as $row): ?>

                <tr>

                    <!-- <td><?php echo $row['pic']; ?></td> -->

                    <td><?php echo $row['name']; ?></td>

                    <td><?php echo $row['date']; ?></td>

                    <td><?php echo $row['about']; ?></td>

                </tr>

            <?php endforeach; ?>

        </table></div>

    </div>

    <div class="container">

        <div class="content row">

            <div class="main-content col-md-9 col-12">

                <h2>Рекомендуемые фильмы</h2>

                <div class="post row">

                    <div class="img col-12 col-md-4">

                        <img src="irsud.jpg" class="img-thumbnail">

                    </div>

                    <div class="post\_text col-12 col-md-8">

                        <h3>

                            <a href="single.php">Ирония судьбы</a>

                        </h3>

                        <i > 1975</i>

                        <p class="prewiew\_text "> 31 декабря пошли друзья в баню попариться,

по традиции смыть с себя все плохое, чтобы встретить Новый год чистыми,  а «напарились» до такой степени

                            (пивом и водкой), что отправили в Ленинград не того.</p>

                    </div>

                </div>

                <div class="post row">

                    <div class="img col-12 col-md-4">

                        <img src="tarng.jpg" class="img-thumbnail">

                    </div>

                    <div class="post\_text col-12 col-md-8">

                        <h3>

                            <a href="single2.php">Тариф новогодний</a>

                        </h3>

                        <i> 2008</i>

                        <p class="prewiew\_text "> Андрей (Максим Матвеев), молодой парень,

                            работающий переводчиком, в канун новогоднего праздника приезжает из

                            Австрии домой и случайно — благодаря волшебному тарифу для сотового

                            телефона — знакомится с девушкой по имени Алёна (Валерия Ланская),

                            скрипачкой. Они пытаются встретиться, но неожиданно выясняют, что

                            находятся в Москве… с разницей в год! Более того, Андрей вскоре узнаёт,

                             что Алёне суждено погибнуть в дорожной аварии. </p>

                    </div>

                </div>

            </div>

            <div class="main-content col-md-3">

            </div>

        </div>

    </div>

  </body>

</html>

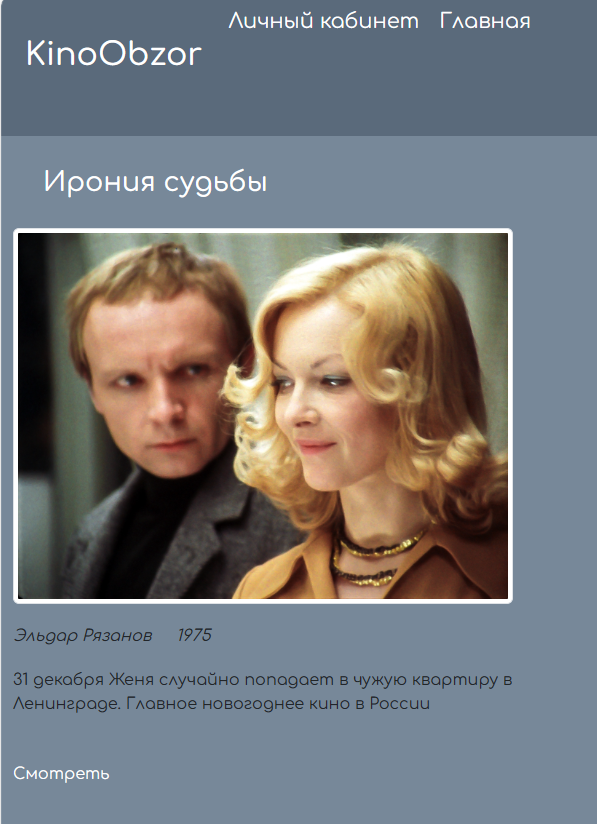


Рисунок 21 — Страница фильма

Листинг стилей для сайта:

\* {

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

}

h1, h2, h3, h4,h5,h6, html { color:white;}

body {background-color: lightslategray;}

a{

    text-decoration: none;

    color: white;

}

input{

   padding: 10px 12px ;

   display: block;

   width: 100%;

   border-radius: 3px;

   border: 1px solid rgb(114, 146, 171);

   line-height: 1.5rem;

   color: rgb(65, 82, 96) ;

}

header{

    background-color: rgb(90, 106, 123) ;

height: 140px;

}

header h1 {

    margin-top: 40px;

}

header h1 :hover{

    color: black;

}

header ul {

    margin: 0;

    padding: 0;

    list-style-type: none;

}

header ul li {

   float: right;

   position: relative;

}

header ul li a{

   display: block;

   font-size: 1.3rem;

   text-decoration: none;

   padding: 10px;

   float: right;

 }

 header ul li :hover{

    color: black;

    background-color: aliceblue;

  }

  .slider-title {

    text-align: center;

    padding: 30px;

  }

  .carousel{

    height: 400px;

  }

  .carousel-inner, .carousel-item{

    height: 400px;

  }

  .carousel h5{

    font-size:2rem ;

    font-weight: 500;

    background: black;

    max-width: 100%;

    padding: 0.2em;

    opacity: 0.5;

    bottom: 0.05rem;

  }

  .carousel-caption :hover{

   text-decoration: underline;

   color: antiquewhite;

  }

  .container{

    float: left;

  }

  .main-content h2{

    margin: 30px;

  }

  .main-content .post{

    margin-bottom: 30px;

  }

  .main-content .post\_text{

    padding:7px ;

    background-color: rgb(76, 92, 120);

    color: aliceblue;

    border-radius: 3px;

  }

  .main-content .post\_text h3 {

    margin: 13px 0px 13px 0px;

 }

 .main-content .post\_text i{

    margin: 13px 0px 13px 0px;

 }

  .main-content .post\_text h3 a:hover{

     text-decoration: underline;

  }

  .single\_post {

    margin-bottom: 30px;

}

  .single\_post .info{

    margin: 20px 0px;

}

  .single\_post .info i{

    display: inline-block;

    margin-right: 20px;

}

.container2{

    width: 50%;

    margin:0 25% 0 25%;

    text-align: center;

}

.container2 .reg\_form a{

    text-decoration: none;

    margin-left: 10px;

}

.container2 .reg\_form a :hover{

    text-decoration: underline;

    color: rgb(120, 120, 120);

 }

 .reg\_form input{

    height: 3rem;

 }

 .container2 .form .ah :hover{

  text-decoration: underline;

  color: rgb(120, 120, 120);

 }

 .films{

  margin-left: 100px;

  padding:40px ;

  background-color: rgb(163, 170, 183);

  color: aliceblue;

  border-radius: 3px;

 }

 .s{

  margin-left: 100px;

  padding:40px ;

  background-color: rgb(163, 170, 183);

  color: aliceblue;

  border-radius: 3px;

 }

# ЗАКЛЮЧЕНИЕ

В заключении хочется сказать, Web-приложение по теме "кино" представляет собой актуальное и востребованное решение в современном мире развлечений. С учетом постоянно растущей популярности кинематографа, такое приложение предоставляет пользователям удобный способ находить информацию о фильмах.

Разделение ролей пользователей в таком приложении может включать в себя обычных пользователей, которые могут просматривать информацию о фильмах, и зарегистрированных пользователей.

Расширенный функционал администраторов включает возможность добавления, изменения и удаления фильмов и пользователей. Такой функционал позволяет администраторам эффективно управлять контентом и обеспечивать качество обслуживания пользователей.

Также следует отметить, что данное web-приложение имеет адаптивный макет, и весь контент меняет своё расположение в зависимости от формата экрана.