Министерство науки и высшего образования РФ

Федеральное государственное автономное образовательное учреждение высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт управления бизнес-процессами

Кафедра бизнес-информатики и моделирования бизнес-процессов

**КУРСОВАЯ РАБОТА**

по дисциплине «Проектирование, разработка и оптимизация web-приложений»

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

Преподаватель

подпись, дата

Никитин А.И.

Студент УБ23-09Б, 432320749

подпись, дата

Красноярск 2024

Карташова К.А.

**СОДЕРЖАНИЕ**

[Введение 3](#_Toc156153021)

[1 Теоретическая часть 4](#_Toc156153022)

[1.1 HTML И CSS 4](#_Toc156153023)

[1.2 JavaScript 5](#_Toc156153024)

[1.3 PHP 6](#_Toc156153025)

[1.4 MySQL 7](#_Toc156153026)

[2 Практическая часть 9](#_Toc156153027)

[2.1 Требования к работе 9](#_Toc156153028)

[2.2 Создание базы данных 11](#_Toc156153029)

[2.3 Регистрация и авторизация 12](#_Toc156153030)

[2.4 Главная страница 15](#_Toc156153031)

[2.5 Личный кабинет 23](#_Toc156153032)

[Заключение 31](#_Toc156153033)

[Список использованных источников 32](#_Toc156153034)

# Введение

В современном мире веб-программирование становится всё более значимой сферой деятельности. Одним из ключевых аспектов этой области является разработка веб-приложений, которые направлены на улучшение и автоматизацию бизнес-процессов. В рамках данной курсовой работы рассматривается перспективное направление — создание веб-сайта «Музыкальный портал» с использованием MySQL для хранения данных. Сфера досуга, подобно сфере общественного питания, нуждается в инновационных решениях для автоматизации внутренних процессов. Веб-разработка предоставляет возможность создать продукт с интересным функционалом, который будет полезен как для посетителей, так и для администраторов.

Создание базы данных MySQL для веб-приложения, предназначенного для публикации музыкальных композиций, представляет собой ключевой этап в обеспечении эффективности работы приложения. Разработка базы данных MySQL для такого веб-приложения является неотъемлемой составляющей процесса обеспечения эффективности и гибкости функционирования приложения. Этот компонент способствует эффективному управлению данными, обеспечивает безопасность информации и создаёт возможности для взаимодействия с другими системами, что позволяет более гибко управлять процессом.

Целью данной курсовой работы является создание сайта с использованием HTML, CSS, Javascript, PHP, MySQL и разработка базы данных для управления пользователями с помощью СУБД MySQL. В работе используются методы анализа и синтеза информации, проектирования, а также практической реализации.

Объект исследования: Объектом исследования является само веб-приложение.

Предмет исследования: Предметом исследования является процесс создания веб-приложения и базы данных.

# 1 Теоретическая часть

# 1.1 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.

HTML является языком разметки, предназначенным для структурирования информации на веб-странице. С его помощью создаются различные элементы, такие как заголовки, параграфы, списки, изображения и гиперссылки. HTML предоставляет универсальные теги, каждый из которых определяет определенный тип контента. Например, тег <h1> используется для создания заголовков первого уровня, а тег <p> - для параграфов.

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.

CSS, с другой стороны, используется для стилизации веб-страницы. Он позволяет определять внешний вид и оформление элементов HTML, включая цвета, шрифты, расположение, отступы, рамки и другие аспекты дизайна. CSS также способствует созданию адаптивного и отзывчивого дизайна, который позволяет сайту хорошо выглядеть на различных устройствах и экранах.

Кроме того, CSS позволяет создавать анимации, переходы, градиенты и другие эффекты, делая веб-страницы более интерактивными и привлекательными для пользователей.

Сочетание HTML и CSS позволяет веб-разработчикам создавать динамичные и креативные веб-страницы. Применение семантической разметки с использованием HTML способствует лучшему восприятию контента поисковыми системами, а гибкость CSS дает возможность адаптировать внешний вид страницы под различные устройства и разрешения экранов. Эти инструменты являются основополагающими для веб-разработчика, открывая двери к созданию привлекательных, удобных в использовании и технологически современных веб-приложений.

В современной веб-разработке HTML и CSS также играют ключевую роль в обеспечении доступности и адаптивности веб-приложений. Стандарты разметки HTML5 вводят новые элементы и атрибуты, улучшая структурирование контента и обеспечивая большую гибкость для веб-разработчиков. Новые возможности, такие как мультимедийные элементы <audio> и <video>, а также API для работы с локальным хранилищем, значительно расширяют функционал веб-приложений.

CSS3, в свою очередь, вводит множество новых свойств и селекторов, что позволяет создавать более сложные и интерактивные дизайны. Технологии, такие как Flexbox и Grid Layout, предоставляют разработчикам мощные инструменты для управления расположением элементов на веб-странице.

В целом, HTML и CSS работают вместе для создания привлекательного, удобного в использовании и эстетически приятного веб-сайта, который обеспечивает хорошее визуальное впечатление и удовлетворяет потребности пользователей.

# 1.2 JavaScript

JavaScript — это высокоуровневый, интерпретируемый язык программирования, который широко используется для создания интерактивных веб-страниц. Он был создан в 1995 году компанией Netscape и изначально назывался LiveScript, но затем был переименован в JavaScript.

JavaScript является частью технологий, которые обеспечивают динамичное взаимодействие на веб-страницах. Он используется для создания интерактивных элементов, анимаций, обработки данных, валидации форм, обновления содержимого страницы без перезагрузки и многих других возможностей.

JavaScript был создан для добавления возможности создания динамичных элементов на стороне клиента, что позволяет веб-страницам реагировать на действия пользователя без необходимости обновления всей страницы. Этот язык поддерживает событийно-ориентированное программирование, что означает, что код может реагировать на различные события, такие как нажатия клавиш, клики мыши и изменения веб-элементов.

Основные особенности JavaScript:

1. Язык сценариев. JavaScript предназначен для выполнения на стороне клиента, в браузере пользователя. Это позволяет создавать динамичные интерфейсы и взаимодействовать с пользователем без необходимости отправлять запросы на сервер.

2. Объектно-ориентированный. JavaScript поддерживает объектно-ориентированную парадигму программирования, что позволяет создавать модульные и масштабируемые приложения.

3. Мультипарадигмальный. JavaScript поддерживает не только объектно-ориентированный подход, но также функциональное программирование и императивный стиль. Это позволяет разработчикам выбирать наиболее подходящий подход в зависимости от конкретной задачи.

4. Динамически типизированный. Переменные в JavaScript являются динамически типизированными, что означает, что тип данных переменной определяется во время выполнения программы.

5. Широкая поддержка. JavaScript поддерживается всеми современными браузерами и используется не только на веб-страницах, но также в различных фреймворках и библиотеках для разработки клиентской и серверной части приложений.

Благодаря своим возможностям, JavaScript стал одним из самых популярных языков программирования и широко используется в веб-разработке для создания динамичных и интерактивных пользовательских интерфейсов.

# 1.3 PHP

PHP (Hypertext Preprocessor) представляет собой язык программирования, специально разработанный для веб-разработки, и широко используется для создания динамических веб-приложений, работы с базами данных и обработки данных на стороне сервера.

Основные возможности PHP:

- Интеграция с HTML: PHP позволяет встраивать свой код непосредственно в HTML, что позволяет создавать динамические веб-страницы.

- Работа с базами данных: PHP обеспечивает удобное взаимодействие с различными системами управления базами данных.

- Обработка форм и данных: PHP является эффективным инструментом для обработки данных, отправляемых с веб-форм.

- Создание сессий и куки: PHP предоставляет механизмы для хранения информации о состоянии пользователя между запросами и для хранения информации на стороне клиента.

- Работа с файлами: PHP предоставляет средства для работы с файловой системой сервера, включая загрузку, скачивание и обработку файлов.

Актуальность использования PHP в веб-приложениях:

- Широкое распространение: PHP широко поддерживается хостинг-провайдерами и работает на большинстве серверов.

- Сообщество и ресурсы: PHP обладает активным сообществом разработчиков, множеством библиотек и фреймворков.

- Простота и гибкость: Язык предоставляет простой синтаксис, доступный для новичков, но при этом гибок для опытных разработчиков.

- Обширный инструментарий: PHP легко интегрируется с другими технологиями и инструментами, что делает его универсальным решением для создания различных веб-приложений.

PHP остается одним из наиболее популярных языков для серверной веб-разработки благодаря своей простоте использования и мощным возможностям. Однако разработчики должны быть в курсе его ограничений и использовать современные и безопасные методы.

# 1.4 MySQL

MySQL представляет собой открытую реляционную систему управления базами данных, разработанную и поддерживаемую компанией Oracle. Эта система предоставляет широкий спектр возможностей для хранения, обработки и извлечения данных. В данном тексте мы рассмотрим ключевые особенности и характеристики MySQL:

1. Множество типов данных: MySQL поддерживает различные типы данных, включая числовые, строковые, даты/время, JSON и другие, что позволяет эффективно хранить разнообразную информацию;

2. Индексация данных: MySQL обеспечивает возможность создания индексов для ускорения поиска и сортировки данных. Это повышает производительность при выполнении запросов к базе данных;

3. Транзакционная обработка: MySQL поддерживает транзакции, что позволяет обеспечить целостность данных при выполнении операций в рамках базы данных;

4. Расширяемость: MySQL имеет модульную архитектуру, что облегчает расширение его функциональности с помощью различных типов расширений, хранимых процедур, триггеров и других объектов базы данных;

5. Высокая производительность: MySQL обладает оптимизированным ядром и механизмами кэширования, что позволяет обеспечить высокую производительность при обработке запросов к базе данных;

6. Кластеризация и репликация: MySQL поддерживает кластеризацию и репликацию, что позволяет создавать высокодоступные и масштабируемые конфигурации баз данных;

7. Совместимость и поддержка: MySQL поддерживает стандартные языки запросов (SQL), а также имеет обширную документацию, активное сообщество пользователей и широкий набор инструментов для администрирования и разработки.

MySQL активно используется в веб-приложениях, системах управления контентом, интернет-магазинах, аналитических системах и других сферах, где необходимо эффективное управление и обработка больших объёмов информации.

# 2 Практическая часть

# 2.1 Требования к работе

Необходимо разработать web-приложения с применением современных web-технологий по тематике «Музыкальный портал», используя технологии HTML, CSS, JavaScript, PHP и MySQL. Кроме того для своего проекта необходимо разработать базу данных для управления пользователями. 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:

– должна быть предусмотрена авторизация и регистрация пользователей;

– зарегистрированный пользователь имеет право редактировать свои данные и просматривать свои услуги;

– администраторы имеют право редактировать пользователей и добавлять услуги;

– неавторизированный пользователи имеют право просматривать список услуг;

– сайт должен быть выдержан в определенном стиле и корректно открываться на любом разрешении;

– общий функционал может быть продуман самостоятельно;

– важно продемонстрировать не большой объем функционала, а минимальные требования к проекту – он может быть максимально простым, но при этом содержать все необходимые функции;

– отдельные бонусы предусмотрены за доп. функционал и красивый дизайн.

В качестве инструментов в работе будет использован текстовый редактор VScode, приложение для создания локального сервера OPEN server, в котором также можно найти веб-приложение phpMyAdmin для администрирования СУБД MySQL.

# 2.2 Создание базы данных

Перед началом работы необходимо запустить приложение OPEN server и подождать, С помощью встроенного функционала (Рисунок 1) создалась база данных с кодировкой utf8\_general\_ci.

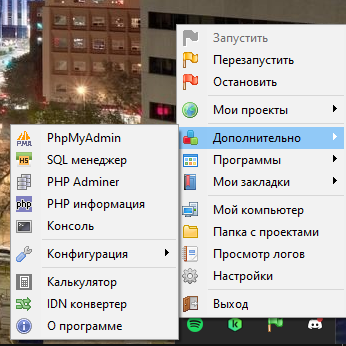


Рисунок 1 – запуск серверов через OPEN server

Далее следует открыть веб-приложение phpMyAdmin (рис.2), в котором можно создать базу данных через вспомогательные кнопки или напрямую через консоль.

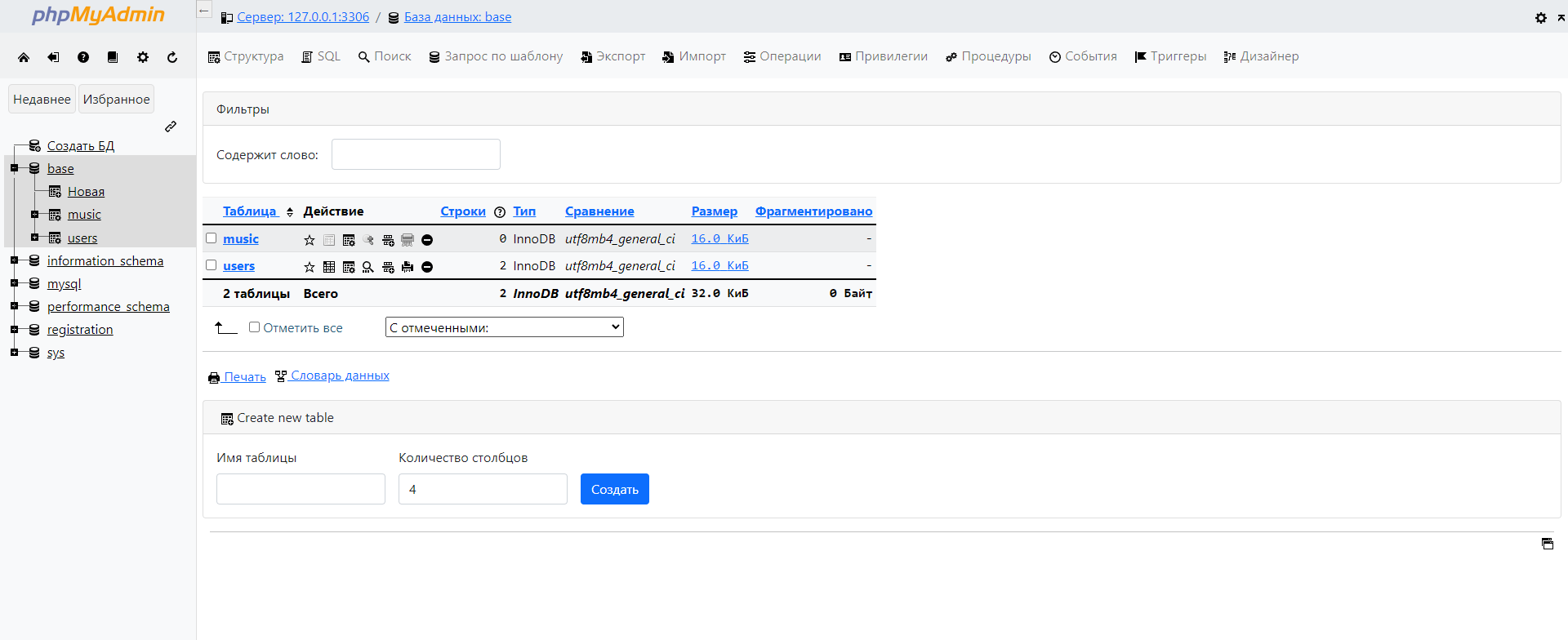


Рисунок 2 – веб-приложение phpMyAdmin

После создания базы данных следует проверить корректность ее подключения. Для этого можно использовать серверный язык php. В текстовом редакторе достаточно ввести следующий код:

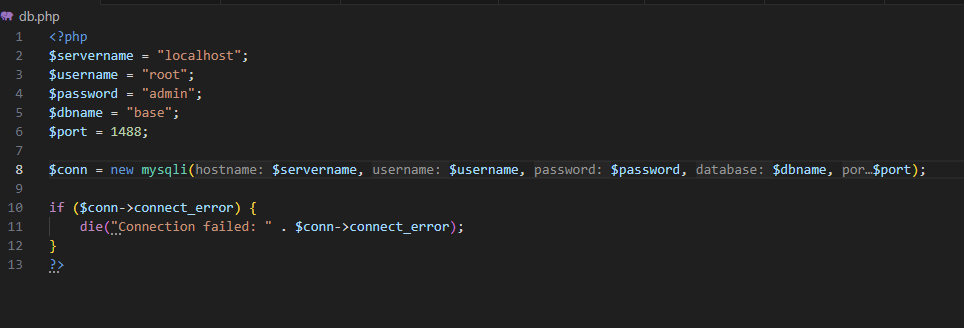


Рисунок 3 – код подключения

Теперь, при ошибке, связанной с подключением базы данных, выведется соответствующее сообщение.

# 2.3 Регистрация и авторизация

На данном этапе создается файл index.html. Он представляет собой белый фон с окном по середине, со всеми нужными функциями. Также неавторизованный пользователь может сразу перейти в веб-приложение.

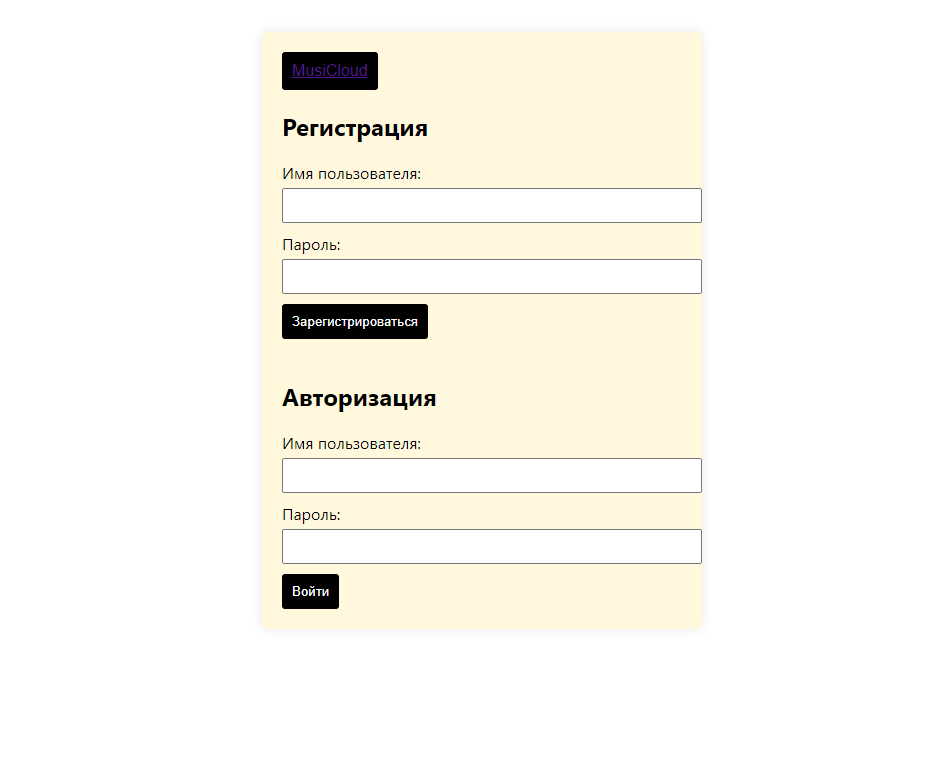


Рисунок 4 – окно регистрации и авторизации

Листинг index.html:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

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

    <title>Регистрация MusiCloud</title>

    <link rel="icon" type="imeges/x icon" href="favicon/sc.ico">

</head>

<body>

    <div class="container">

        <button type="submit"><a class="visit-music" href="nomusic.php" style="font-size: 16px;">MusiCloud</a></button>

        <br>

        <h2>Регистрация</h2>

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

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

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

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

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

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

        </form>

        <br>

        <h2>Авторизация</h2>

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

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

            <input type="text" id="login\_username" name="login\_username" required>

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

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

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

        </form>

    </div>

</body>

</html>

Также для связи функций используются следующие PHP файлы:

Листинг register.php:

<?php

require\_once('db.php');

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

    $username = $\_POST['username'];

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

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

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

        header("Location: index.html");

    } else {

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

    }

}

$conn->close();

?>

Листинг login.php:

<?php

session\_start();

require\_once('db.php');

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

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

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

    $sql = "SELECT \* FROM users WHERE username='$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 "Invalid password";

        }

    } else {

        echo "User not found";

    }

}

$conn->close();

?>

Листинг logout.php:

<?php

session\_start();

session\_destroy();

header("Location: index.html");

?>

# 2.4 Главная страница

Попасть на главную страницу могут как авторизованные, так и неавторизованные пользователи. Сам документ называется music.php и содержит весь основной контент об веб-приложении MusuCloud, используемом в качестве примера для курсовой работы.

Листинг music.php:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <link rel="icon" type="imeges/x icon" href="favicon/sc.ico">

    <style>

        body {

            text-align: center;

        }

        a {

            font-size: 24px;

            text-decoration: none;

            color: #4CAF50;

        }

    </style>

</head>

<body>

    <?php

    session\_start();

    require\_once('db.php');

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

        header("Location: index.html");

        exit();

    }

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

    $sql\_role = "SELECT role FROM users WHERE id=$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') {

            echo '<a href="admin.php"><br>Перейти в карточку артиста</a>';

        }

    }

    ?>

</body>

</html>

<!DOCTYPE html>

<html>

<head>

    <meta charset="UTF-8">

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

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

    <title>MusiCloud</title>

    <link rel="icon" type="imeges/x icon" href="favicon/sc.ico">

</head>

<body>

    <div class="musiccontainer">

        <h2>MusiCloud</h2>

    </div>

    <div class="musiccontainer">

        <h3>Главные хиты</h3>

    </div>

    <div class="musiccontainer">

        <div class="music">

            <img src="Snake Eyes.jpg" width="800" height="700">

            <h3>Snake Eyes</h3>

            <p> Snake Eyes — песня Feint с вокалом CoMa. Она была выпущена 12 октября 2012 года и включена в качестве тринадцатого трека в Monstercat 010 — Conquest, третьего трека в Monstercat — Best of Drum & Bass / Drumstep Vol. 2 и девятого трека в Monstercat — Best of 2012.                </p>

                <audio  controls="controls">

                        <source src="Snake Eyes.mp3" type="audio/mpeg">

                        Установите гугл хром, ваш браузер плохой

                    </audio>

        </div>

        <div class="music">

            <img src="Talkless - Why.jpg" width="800" height="700">

            <h3>Talkless - Why (feat. jnhygs)</h3>

            <p> Talkless — это проект Банчи Теаракит (Nimp) и Втаньи Чантиван (Fon). Нимп — гитарист группы Goose, и это его сайд-проект. Они с Фоном потратили более 2 лет на создание этого мини-альбома «Dot Dot Dot». Они выпустили свой альбом на 7-м фестивале Fat на музыкальном лейбле SO::ON Dry Flower. </p>

                <audio  controls="controls">

                        <source src="Talkless - Why.mp3" type="audio/mpeg">

                        Установите гугл хром, ваш браузер плохой

                    </audio>

        </div>

        <div class="music">

            <img src="robopup.jpg" width="800" height="700">

            <h3>Robopup-program me</h3>

            <p> Robopup-певица, исполняющая гиперпоп/сценкор, которая покорила сердца многих пользователей своим прекрсным голосом, выпускает треки как на английском, так и на русском языках. Набрала большую популярность из-за релиза "yandere gf" с "Mezha". Позиционирует себя как AI, и желанная девушка для эмо-парней.</p>

            <audio  controls="controls">

                        <source src="robopup.mp3" type="audio/mpeg">

                        Установите гугл хром, ваш браузер плохой

                    </audio>

        </div>

        <div class="music">

            <img src="odetari.png" width="800" height="700">

            <h3>Odetari-GOOD LOYAL THOTS</h3>

            <p> Таха Отман Ахмад, профессионально известный как Odetari, — американский певец палестинского происхождения, рэпер, автор песен и продюсер. Он начал продюсировать и выпускать музыку в 13 лет. Его музыка стала популярной в TikTok в 2023 году, а его песни «Good Loyal Thots», «Narcissistic Personality Disorder», «Look Don't Touch», «I Love You Hoe» и «GMFU». Все они попали в топ-10 чарта Billboard Hot Dance/Electronic Songs.</p>

            <audio  controls="controls">

                        <source src="Odetari.mp3" type="audio/mpeg">

                        Установите гугл хром, ваш браузер плохой

                    </audio>

        </div>

        <div class="container">

        <h3 align="center" style="color:#000000">Новый хит!!</h3>

        </div>

         <?php

        $sql\_music = "SELECT \* FROM music";

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

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

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

                echo '<div class="music">';

                echo '<img src="' . $row['photo\_path'] . '" alt="' . $row['name'] . '" width="800" height="700">';

                echo '<h3>' . $row['name'] . '</h3>';

                echo '' . $row['opisanie'] . '<br>' ;

                echo '<audio  controls="controls"> <source src="' . $row['music\_path'] .'" type=audio/mpeg> </audio>';

                echo '</div>';

            }

        } else {

            echo '<div class="container"> <p align="center" style="color:#000000">Нет новых хитов(добавь свою неповторимую композицую, но это может сделать только подтврежденный дистрибьютор, то есть admin)</p> </div>';

        }

        $conn->close();

        ?>

    </div>

    <div class="container">

        <div class="contacts">

        <div class="contact-block">

            <div class="contact-block">

                <h4>Твой профиль</h4>

                <a href="dashboard.php">Вернуться в редактор профиля</a>

            </div>

        </div>

    </div>

</body>

</html>

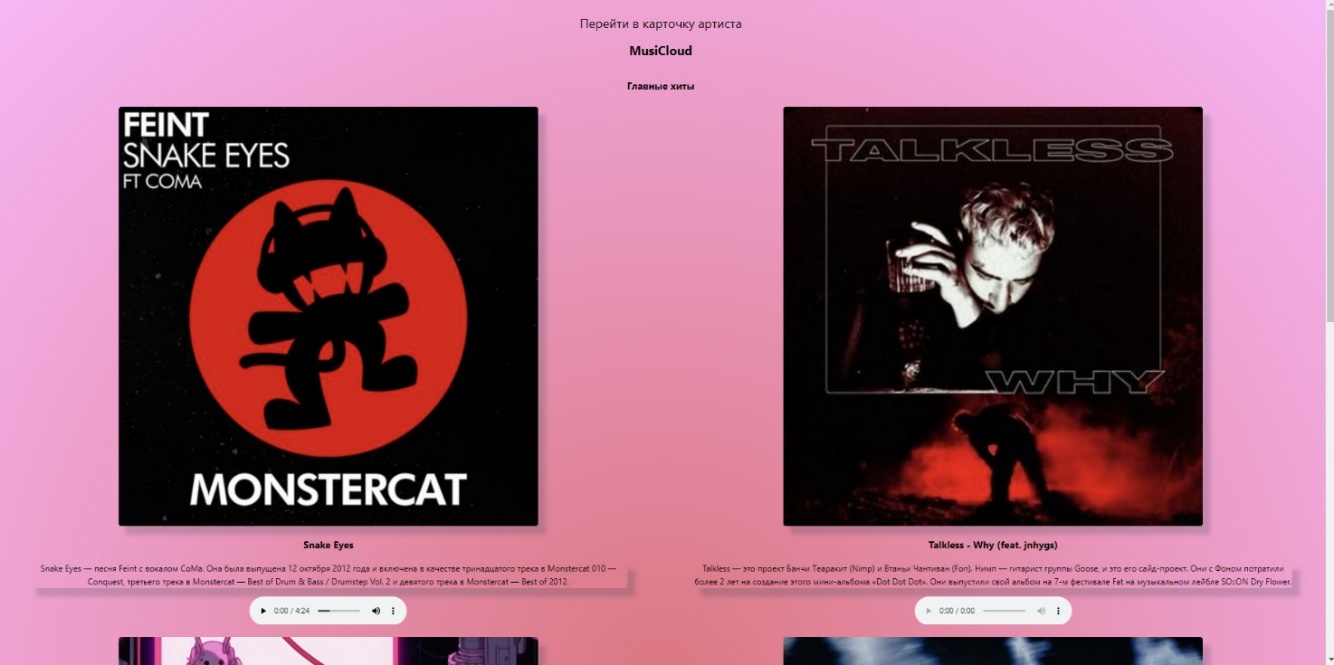


Рисунок 5 – главная страница веб-приложения



Рисунок 6 – отображение выложенной музыки дистрибьютором

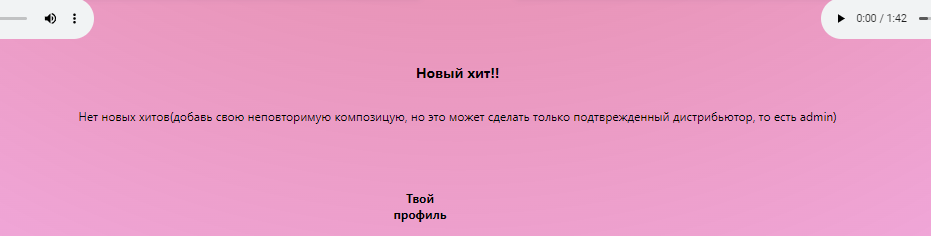


Рисунок 7 – отображение без выложенной музыки дистрибьютором



Рисунок 8 – переход в карточку артиста, которая доступна только дистрибьютору

Листинг страницы «Перейти в карточку артиста» (admin.php):

<?php

session\_start();

require\_once('db.php');

require\_once('add\_music.php');

if ($\_SERVER['REQUEST\_METHOD'] === 'POST' && isset($\_POST['add\_music'])) {

    $name = $\_POST['name'];

    $opisanie = $\_POST['opisanie'];

    $uploadDir = 'uploads/';

    $uploadFile = $uploadDir . basename($\_FILES['photo']['name']);

    if (move\_uploaded\_file($\_FILES['photo']['tmp\_name'], $uploadFile)) {

        $photoPath = $uploadFile;

    } else {

        $photoPath = '';

    }

    $uploadDir = 'uploads/';

        $uploadFile = $uploadDir . basename($\_FILES['music']['name']);

        if (move\_uploaded\_file($\_FILES['music']['tmp\_name'], $uploadFile)) {

            $musicPath = $uploadFile;

        } else {

            $musicPath = '';

        }

    addMusic($name, $opisanie, $photoPath, $musicPath);

}

$conn->close();

?>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

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

    <title>Карточка артиста</title>

    <link rel="icon" type="imeges/x icon" href="favicon/sc.ico">

</head>

<body>

    <div class="container">

        <h2>Карточка артиста</h2>

        <form action="admin.php" method="post" enctype="multipart/form-data">

            <label for="name">Напишите ваш никнейм и название вашего трека:</label>

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

            <br>

            <label for="opisanie">Рассказите что-то о себе и своем релизе:</label>

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

            <br>

            <label for="photo">Обложка(рекомендуемый размер 2000х2000):</label>

            <input type="file" id="photo" name="photo" accept="image/\*" required>

            <br>

            <label for="photo">Загрузите свой музыкальный фаил:</label>

            <input type="file" id="music" name="music" accept=".mp3" required>

            <br>

            <br>

            <button type="submit" name="add\_music">Добавить релиз</button>

        </form>

        <br>

        <br>

        <a href="music.php">MusiCloud</a>

        <br>

        <br>

        <h2>Пример заполнения карточки артиста</h2>

        <img src="primer.jpeg" width="500" height="210">

    </div>

</body>

</html>

Также используется файл add\_music.php:

<?php

function addMusic($name,$opisanie, $photoPath, $musicPath ) {

    global $conn;

    $name = mysqli\_real\_escape\_string($conn, $name);

    $sql = "INSERT INTO music (name, opisanie, photo\_path, music\_path)

            VALUES ('$name','$opisanie','$photoPath','$musicPath')";

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

        $conn->close();

        if (file\_exists("music.php")) {

            header("Location: music.php");

            exit();

        } else {

            echo "Error: music.php not found!";

        }

    } else {

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

    }

}

?>

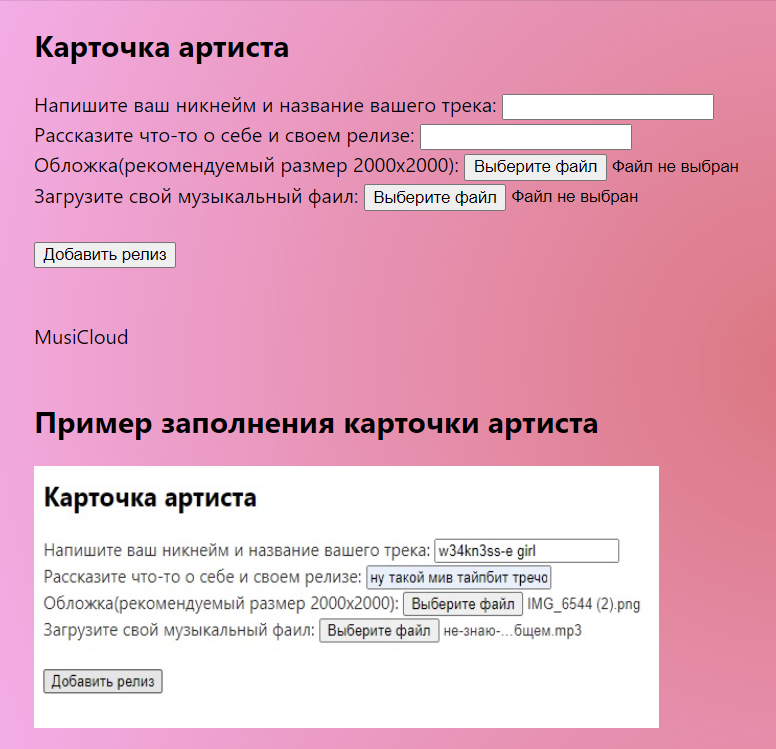


Рисунок 9 –карточка артиста

# 2.5 Личный кабинет

При переходе в личный кабинет (файл dashbroad.php) перед пользователем представлена возможность изменить все необходимые данные (логин, пароль). Также как и для регистрации, здесь используется функция, помогающая определить корректность ввода пароля. Имеется возможность выхода из аккаунта. Сама страница представлена на рисунке 10.

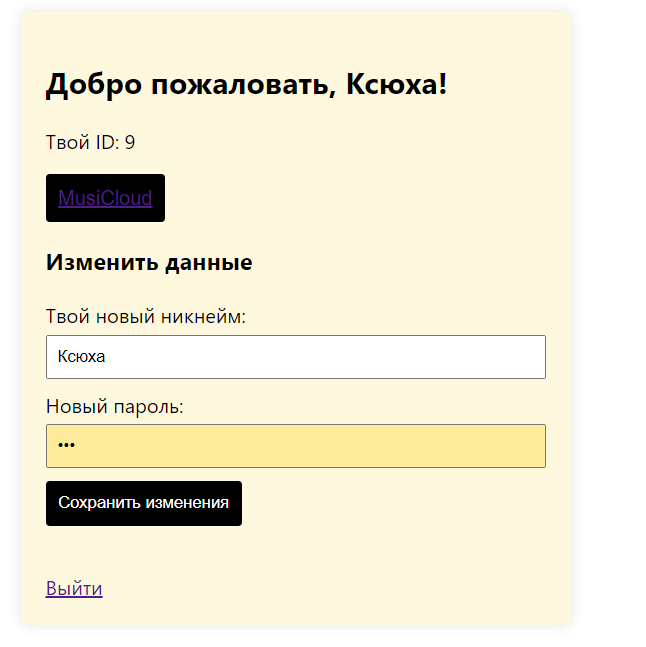


Рисунок 10 – личный кабинет

Для администраторов имеется дополнительная функция, позволяющая работать с панелью администратора (рис.11).

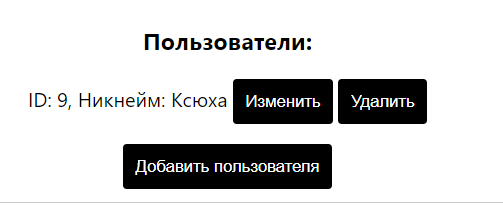


Рисунок 11 – панель администратора

Для объяснения принципа работы всей страницы стоит ознакомиться с кодом:

Листинг dashboard.php:

<?php

session\_start();

require\_once('db.php');

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

    header("Location: index.html");

    exit();

}

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

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

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

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

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

            echo "<h3>Пользователи:</h3>";

            echo "<ul>";

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

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

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

        }

    }

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

    $result = $conn->query($sql);

    if ($result->num\_rows > 0) {

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

        echo "<h2> </h2>";

    } else {

        echo "User not found";

    }

} else {

    echo "Error fetching user role.";

}

$conn->close();

?>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <link rel="icon" type="imeges/x icon" href="favicon/sc.ico">

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

    <title>Твой профиль</title>

    <link rel="icon" type="imeges/x icon" href="favicon/sc.ico">

</head>

<body>

    <div class="container">

        <h2>Добро пожаловать, <?php echo $user['username']; ?>!</h2>

        <p>Твой ID: <?php echo $user['id']; ?></p>

        <button type="submit"><a class="visit-music" href="music.php" style="font-size: 16px;">MusiCloud</a></button>

        <h3>Изменить данные</h3>

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

            <input type="hidden" name="edit\_user\_id" id="edit\_user\_id" value="">

            <label for="new\_username">Твой новый никнейм:</label>

            <input type="text" id="new\_username" name="new\_username" value="<?php echo $user['username']; ?>" required>

            <label for="new\_password">Новый пароль:</label>

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

            <button type="submit">Сохранить изменения</button>

        </form>

        <br>

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

    </div>

    <script>

        function editUser(userId) {

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

            document.getElementById("new\_username").value = prompt("Enter new username:", "");

            document.getElementById("new\_password").value = prompt("Enter new password:", "");

            document.forms[0].submit();

        }

        function confirmDelete(userId) {

            var confirmDelete = confirm("Are you sure you want to delete this user?");

            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("Enter new username:", "");

            var password = prompt("Enter new password:", "");

            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>

Также используются следующие PHP файлы:

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);

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

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

        header("Location: dashboard.php");

        exit();

    } else {

        echo "Error adding new user: " . $conn->error;

    }

} else {

    echo "Invalid request";

}

$conn->close();

?>

Листинг edit\_user.php:

<?php

require\_once('db.php');

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

    $editUserId = $\_POST['edit\_user\_id'];

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

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

    $sql = "UPDATE users SET username=?, password=? WHERE id=?";

    $stmt = $conn->prepare($sql);

    $stmt->bind\_param("ssi", $newUsername, $newPassword, $editUserId);

    if ($stmt->execute()) {

        header("Location: dashboard.php");

        exit();

    } else {

        echo "Error updating user data: " . $conn->error;

    }

    $stmt->close();

} else {

    echo "Invalid request";

}

$conn->close();

?>

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 users WHERE id = $userToDelete";

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

        header("Location: dashboard.php");

        exit();

    } else {

        echo "Error deleting user: " . $conn->error;

    }

} else {

    echo "Invalid request";

}

$conn->close();

?>

Далее рассмотрим созданную базу данных.

Созданная база данных имеет следующие таблицы:



Рисунок 12 – база данных

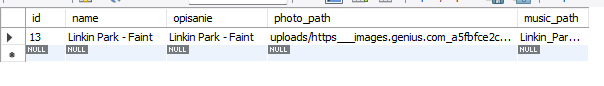


Рисунок 13 – Таблица выложенных релизов на сайт

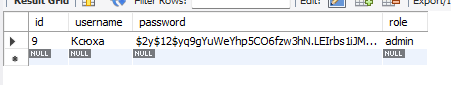


Рисунок 14 – Таблица пользователей

# Заключение

В рамках данного курсового проекта было разработано веб-приложение, использующее передовые веб-технологии, такие как HTML, CSS, JavaScript, PHP и MySQL. Также была создана база данных, предназначенная для управления пользователями и внутренними сервисами.

Дизайн проекта отличается минималистичностью, но при этом обладает всеми необходимыми функциональными возможностями.

В приложении представлены формы регистрации и авторизации, где были использованы простые алгоритмы безопасности, такие как хеширование паролей и проверка их минимальной длины. Также были разработаны основные и вспомогательные страницы интернет-магазина.

Был создан личный кабинет, доступный только для авторизованных пользователей, где они могут изменить все необходимые данные (логин, пароль), выйти из аккаунта или полностью его удалить. При наличии роли администратора пользователю отображается специальная панель, с помощью которой можно изменять данные пользователей, удалять и добавлять аккаунты, а также добавлять новые музыкальные композиции на главной странице интернет-магазина.

# Список использованных источников

1. Основы CSS: mdn web docs [Электронный источник]. – URL: https://developer.mozilla.org/ru/docs/Learn/Getting\_started\_with\_the\_web/CSS\_basics [дата доступа 12.01.2024].
2. Основы HTML: mdn web docs [Электронный источник]. – URL: https://developer.mozilla.org/ru/docs/Learn/Getting\_started\_with\_the\_web/HTML\_basics [дата доступа 12.01.2024].
3. Основы JavaScript: mdn web docs [Электронный источник]. – URL: https://developer.mozilla.org/ru/docs/Learn/Getting\_started\_with\_the\_web/JavaScript\_basics [дата доступа 12.01.2024].
4. Подробнее о PHP [Электронный источник]. – URL:
5. https://ru.hexlet.io/courses/php-setup-environment/lessons/what-is-php/theory\_unit (дата обращения: 12.01.2024).
6. Что такое MySQL: Макхост [Электронный источник]. – URL: https://mchost.ru/articles/chto-takoe-mysql/ [дата доступа 12.01.2024].
7. Что такое - PHP? [Электронный источник]. – URL: https://archive-ipq-co.narod.ru/ (дата обращения: 12.01.2023).
8. Что такое HTML и CSS [Электронный источник]. – URL: https://shneider-host.ru/blog/chto-takoe-html-i-css.html (дата обращения: 12.01.2024).
9. Язык разметки HTML, таблицы стилей CSS и язык программирования JavaScript [Электронный источник]. – URL: https://web-creator.ru/articles/about\_client\_side (дата обращения: 12.01.2024).
10. PHP: википедия [Электронный источник]. – URL: https://ru.wikipedia.org/wiki/PHP [дата доступа 12.01.2024].