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 «Мобильное приложение»»

тема

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

подпись, дата инициалы, фамилия

Студент УБ22-09Б, 432212157 \_\_\_\_\_\_\_\_ М.В. Корниенко

номер группы, зачетной книжки подпись, дата инициалы, фамилия

Красноярск 2023

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc154443562)

[1 Теоретическая часть 5](#_Toc154443563)

[1.1 HTML и CSS 5](#_Toc154443564)

[1.2 JavaScript 6](#_Toc154443565)

[1.3 PHP 8](#_Toc154443566)

[1.4 MySQL 10](#_Toc154443567)

[2 Практическая часть 12](#_Toc154443568)

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

[2.2 Общая структура 13](#_Toc154443570)

[2.3 Создание базы данных 14](#_Toc154443571)

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

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

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

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

[2.1.7 Функционал обычного пользователя 33](#_Toc154443576)

[2.2.7 Функционал администратора 41](#_Toc154443577)

[ЗАКЛЮЧЕНИЕ 46](#_Toc154443578)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 47](#_Toc154443579)

# ВВЕДЕНИЕ

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

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

База данных позволяет эффективно управлять данными, хранить информацию о посетителях, дате и времени бронирования, количестве гостей и других атрибутах заказа. Это позволяет упростить процесс поиска и обработки данных, что способствует более эффективному бронированию столиков.

Нельзя не упомянуть о хранении большого объема данных: web-приложение, связанное с бронированием столиков в ресторане, может содержать значительное количество информации о клиентах, столиках, меню, событиях и других связанных данных. База данных MySQL обеспечивает надежное хранение этих данных и позволяет эффективно управлять ими, даже при большом объеме информации.

Поддержка множества пользователей: Бронирование столиков в ресторане может быть доступно для множества пользователей, как для посетителей, так и для администраторов. MySQL обеспечивает многопользовательскую поддержку, позволяя одновременно работать с базой данных нескольким пользователям без проблем с доступом и интеграцией данных.

В целом, создание базы данных MySQL для веб-приложения по бронированию столиков в ресторане - это ключевой аспект, который способствует эффективной работе приложения, улучшает управление данными, обеспечивает безопасность и защиту данных, а также позволяет взаимодействие с другими системами для более гибкого управления процессом бронирования столиков.

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 и CSS

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

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

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

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

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

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

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

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

## JavaScript

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

Несколько важных функций JavaScript:

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

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

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

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

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

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

Некоторые достоинства JavaScript:

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

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

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

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

## 1.3 PHP

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

Важность применения PHP в веб-разработке заключается в следующем:

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

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

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

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

Функции PHP включают:

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

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

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

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

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

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

## MySQL

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

Функции, предоставляемые MySQL, включают в себя:

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

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

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

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

Применение MySQL для веб-разработки, в частности для бронирования столиков в ресторане, имеет важное значение по следующим причинам:

1. Хранение данных: MySQL позволяет сохранять информацию о клиентах, резервациях, столиках и т. д. и обеспечивает надежность хранения данных.

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

3. Масштабируемость: MySQL поддерживает горизонтальное и вертикальное масштабирование, что позволяет расширять функциональность системы по мере роста бизнеса и увеличения числа резерваций.

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

В целом, использование MySQL для бронирования столиков в ресторане позволяет эффективно управлять и обрабатывать данные, обеспечивает безопасность хранимых данных и способствует развитию функционала системы.

# 2 Практическая часть

## 2.1 Требования и описание реализации

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

**Описание реализации:**

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

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

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

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

- при регистрации пользователю присваивается статус авторизированного пользователя.

**Требования к проекту:**

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

## 2.2 Общая структура

Установка MAMP позволяет воспроизводить web-приложение локально. С помощью встроенного функционала (Рисунок 1) создалась база данных с кодировкой *utf8\_general\_ci* (Рисунок 2).

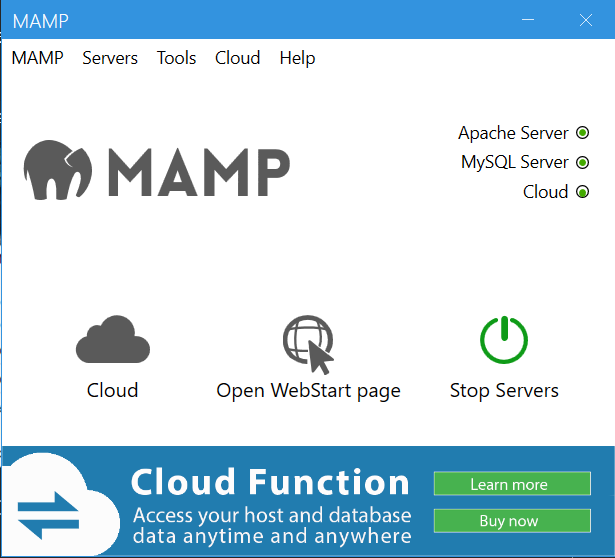


Рисунок 1 — MAMP: встроенный функционал

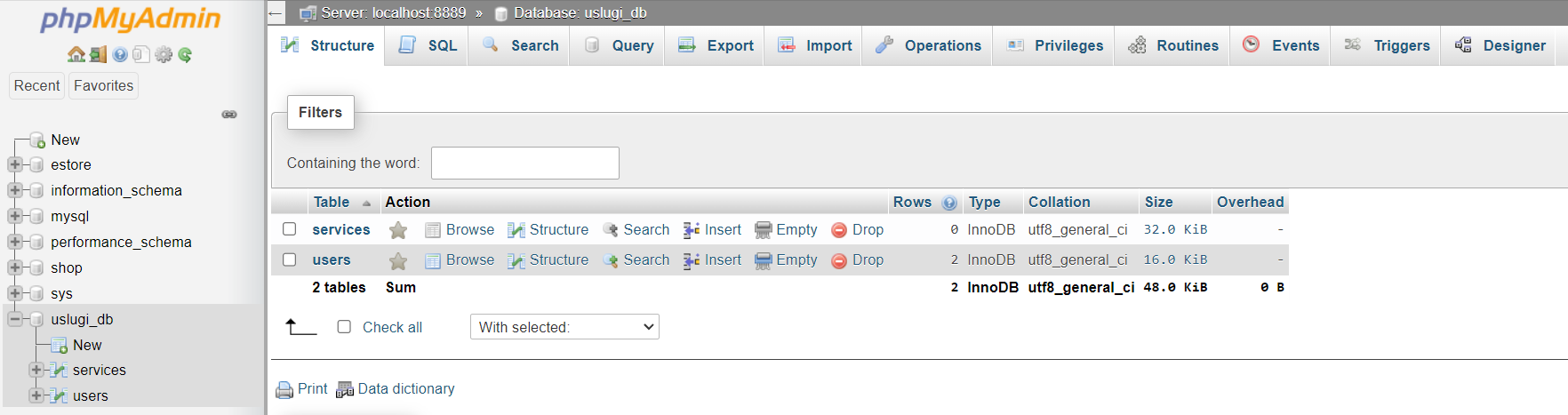


Рисунок 2 — База данных

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



Рисунок 3 — Установка MySQL

## 2.3 Создание базы данных

Листинг создания базы данных:

-- Создание базы данных

CREATE DATABASE IF NOT EXISTS uslugi\_db CHARACTER SET utf8 COLLATE utf8\_general\_ci;

CREATE TABLE `services` (

`id` int(11) NOT NULL,

`title` varchar(255) NOT NULL,

`description` text NOT NULL,

`price` text NOT NULL,

`contacts` varchar(255) NOT NULL,

`user\_id` int(11) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE `users` (

`id` int(11) NOT NULL,

`username` varchar(20) NOT NULL,

`email` varchar(30) NOT NULL,

`password` varchar(255) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

ALTER TABLE `services`

ADD PRIMARY KEY (`id`),

ADD KEY `fk\_user\_id` (`user\_id`);

ALTER TABLE `users`

ADD PRIMARY KEY (`id`);

ALTER TABLE `services`

MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=21;

ALTER TABLE `users`

MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=44;

ALTER TABLE `services`

ADD CONSTRAINT `fk\_user\_id` FOREIGN KEY (`user\_id`) REFERENCES `users` (`id`);

COMMIT;

Название базы данных *uslugi\_db* с кодировкой *utf8\_general\_ci*. Создаётся таблица пользователей, которая содержит в себе id, username, email, password. Примечание: электронная почта должна быть уникальной, зарегистрировать двух пользователей на одну и ту же почту невозможно. Далее создаётся таблица с услугами, которая содержит в себе id услуги, название, описание, цену, контакты и id пользователя, который разместил эту услугу.

## 

## 2.4 Регистрация в web-приложении

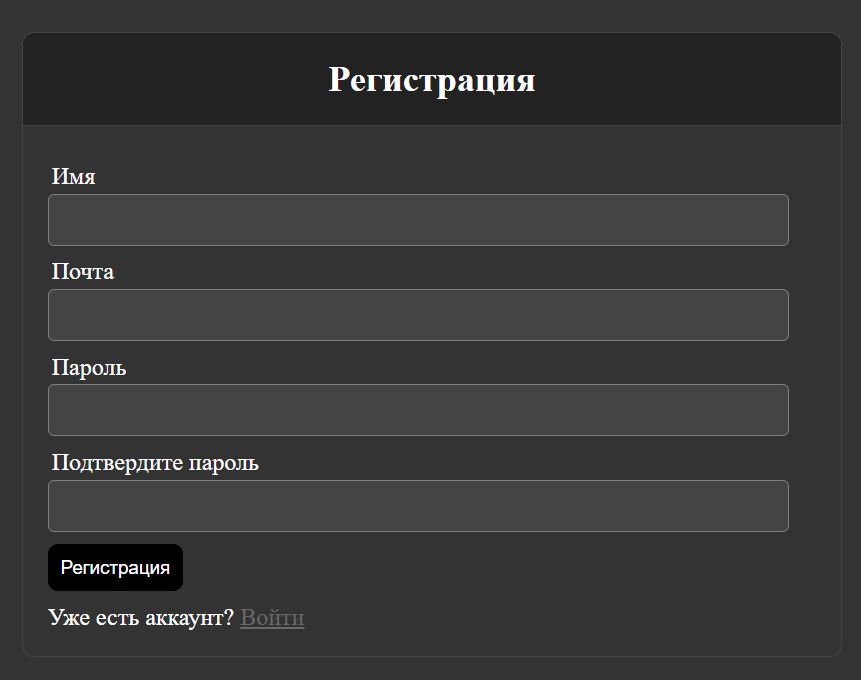


Рисунок 7 — Регистрация в web-приложении

Листинг HTML составляющей панели регистрации:

<?php include('server.php') ?>

<!DOCTYPE html>

<html>

<head>

  <title>Регистрация</title>

  <link rel="stylesheet" type="text/css" href="styles\style\_reg\_log.css">

</head>

<body>

  <div class="header">

        <h2>Регистрация</h2>

  </div>

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

        <?php include('errors.php'); ?>

        <div class="input-group">

          <label>Имя</label>

          <input type="text" name="username" value="<?php echo $username; ?>">

        </div>

        <div class="input-group">

          <label>Почта</label>

          <input type="email" name="email" value="<?php echo $email; ?>">

        </div>

        <div class="input-group">

          <label>Пароль</label>

          <input type="password" name="password\_1">

        </div>

        <div class="input-group">

          <label>Подтвердите пароль</label>

          <input type="password" name="password\_2">

        </div>

        <div class="input-group">

          <button type="submit" class="btn" name="reg\_user">Регистрация</button>

        </div>

        <p>

                Уже есть аккаунт? <a href="login.php">Войти</a>

        </p>

  </form>

</body>

</html>

Форма регистрации содержит в себе поля для ввода имени, электронной почты, пароля, строки подтверждения пароля, а также кнопки «зарегистрироваться». Отдельно хочется добавить удобное перемещение по сайту: если пользователь уже зарегистрирован, то его перебросит на страницу авторизации. «Уже есть аккаунт? Войти».

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

\* {

  margin: 0px;

  padding: 0px;

}

body {

  font-size: 120%;

  background: #333;

}

p {

  color: #ffffff;

}

p a {

  color: #696969;

}

p a:hover {

  color: #ffffff;

}

.header {

  width: 30%;

  margin: 50px auto 0px;

  color: white;

  background: #222;

  text-align: center;

  border: 1px solid #444;

  border-bottom: none;

  border-radius: 10px 10px 0px 0px;

  padding: 20px;

}

form, .content {

  width: 30%;

  margin: 0px auto;

  padding: 20px;

  border: 1px solid #444;

  background: #333;

  border-radius: 0px 0px 10px 10px;

}

.input-group {

  margin: 10px 0px 10px 0px;

}

.input-group label {

  display: block;

  text-align: left;

  margin: 3px;

  color: white;

}

.input-group input {

  height: 30px;

  width: 93%;

  padding: 5px 10px;

  font-size: 16px;

  border-radius: 5px;

  border: 1px solid gray;

  background: #444;

  color: white;

}

.btn {

  padding: 10px;

  font-size: 15px;

  color: rgb(255, 255, 255);

  background: #000000;

  border: none;

  border-radius: 8px;

  transition: color 0.5s, background 0.5s;

}

.btn:hover {

  color: #000000;

  background: rgb(255, 255, 255);

}

.error {

  width: 92%;

  margin: 0px auto;

  padding: 10px;

  border: 1px solid #3d3d3d;

  color: #3d3d3d;

  background: #3d3d3d;

  border-radius: 5px;

  text-align: left;

}

.success {

  color: #3c763d;

  background: #dff0d8;

  border: 1px solid #3c763d;

  margin-bottom: 20px;

}

При нажатии на кнопку «зарегистрироваться» активируется функция, Данная функция устанавливает правила и ограничения для полей «пароль» и «электронная почта».. Листинг функции:

if (isset($\_POST['reg\_user'])) {

    $username = mysqli\_real\_escape\_string($db, $\_POST['username']);

    $email = mysqli\_real\_escape\_string($db, $\_POST['email']);

    $password\_1 = mysqli\_real\_escape\_string($db, $\_POST['password\_1']);

    $password\_2 = mysqli\_real\_escape\_string($db, $\_POST['password\_2']);

    if (empty($username)) {

        array\_push($errors, "Вы не указали имя");

    }

    if (empty($email)) {

        array\_push($errors, "Вы не указали почту");

    }

    if (empty($password\_1)) {

        array\_push($errors, "Вы не указали пароль");

    }

    if ($password\_1 != $password\_2) {

        array\_push($errors, "Пароли не совпадают");

    }

    $user\_check\_query = "SELECT \* FROM users WHERE username='$username' OR email='$email' LIMIT 1";

    $result = mysqli\_query($db, $user\_check\_query);

    $user = mysqli\_fetch\_assoc($result);

    if ($user) {

        if ($user['username'] === $username) {

            array\_push($errors, "Пользователь с таким именем уже существует");

        }

        if ($user['email'] === $email) {

            array\_push($errors, "Пользователь с такой почтой уже существует");

        }

    }

    if (count($errors) == 0) {

        $password = password\_hash($password\_1, PASSWORD\_DEFAULT);

        $query = "INSERT INTO users (username, email, password)

                          VALUES('$username', '$email', '$password')";

        mysqli\_query($db, $query);

        $user\_query = "SELECT id FROM users WHERE username='$username'";

        $user\_result = mysqli\_query($db, $user\_query);

        $user\_row = mysqli\_fetch\_assoc($user\_result);

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

        $\_SESSION['success'] = "You are now logged in";

        header('location: index.html');

    }

}

При проверке пароля, если он неверный, система оповестит о том, что пароли не совпадают. Если же пароли совпали, запись отобразится в базе данных и пароль хешируется (Делается это для безопасности пользователей). Листинг:

// Проверка пароля

if ($password !== $confirm\_password) {

die("Пароли не совпадают");

}

// Хеширование пароля

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

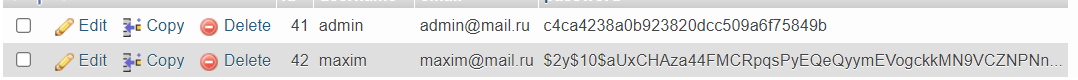


Рисунок 8 — Сохранение регистрации пользователя в базе данных, хеширование пароля

Далее реализована проверка уникальности логина для того, чтобы код не сохранил один и тот же логин дважды. Если почта уже занята, система выдаст оповещение. Листинг:

// Проверка уникальности адреса электронной почты

if (isset($\_POST['reg\_user'])) {

    $username = mysqli\_real\_escape\_string($db, $\_POST['username']);

    $email = mysqli\_real\_escape\_string($db, $\_POST['email']);

    $password\_1 = mysqli\_real\_escape\_string($db, $\_POST['password\_1']);

    $password\_2 = mysqli\_real\_escape\_string($db, $\_POST['password\_2']);

    if (empty($username)) {

        array\_push($errors, "Вы не указали имя");

    }

    if (empty($email)) {

        array\_push($errors, "Вы не указали почту");

    }

    if (empty($password\_1)) {

        array\_push($errors, "Вы не указали пароль");

    }

    if ($password\_1 != $password\_2) {

        array\_push($errors, "Пароли не совпадают");

    }

    $user\_check\_query = "SELECT \* FROM users WHERE username='$username' OR email='$email' LIMIT 1";

    $result = mysqli\_query($db, $user\_check\_query);

    $user = mysqli\_fetch\_assoc($result);

    if ($user) {

        if ($user['username'] === $username) {

            array\_push($errors, "Пользователь с таким именем уже существует");

        }

        if ($user['email'] === $email) {

            array\_push($errors, "Пользователь с такой почтой уже существует");

        }

    }

    if (count($errors) == 0) {

        $password = password\_hash($password\_1, PASSWORD\_DEFAULT);

        $query = "INSERT INTO users (username, email, password)

                          VALUES('$username', '$email', '$password')";

        mysqli\_query($db, $query);

        $user\_query = "SELECT id FROM users WHERE username='$username'";

        $user\_result = mysqli\_query($db, $user\_query);

        $user\_row = mysqli\_fetch\_assoc($user\_result);

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

        $\_SESSION['success'] = "You are now logged in";

        header('location: index.html');

    }

}

## 2.5 Авторизация в web-приложении



Рисунок 9 — Авторизация в web-приложении

Листинг HTML составляющей панели авторизации:

<?php include('server.php') ?>

<!DOCTYPE html>

<html>

<head>

  <title>Вход</title>

  <link rel="stylesheet" type="text/css" href="styles\style\_reg\_log.css">

</head>

<body>

  <div class="header">

        <h2>Вход</h2>

  </div>

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

        <?php include('errors.php'); ?>

        <div class="input-group">

                <label>Имя</label>

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

        </div>

        <div class="input-group">

                <label>Пароль</label>

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

        </div>

        <div class="input-group">

                <button type="submit" class="btn" name="login\_user">Вход</button>

        </div>

        <p>

                Еще нет аккаунта? <a href="register.php">Зарегистрироваться</a>

        </p>

  </form>

</body>

</html>

Авторизация включает в себя форму ввода логина и пароля и кнопку «войти».

Установка соединения с базой данных. Листинг:

<?php

if(session\_status() == PHP\_SESSION\_NONE){

    session\_start();

}

$username = "";

$email = "";

$errors = array();

$db = mysqli\_connect('localhost', 'root', 'root', 'uslugi\_db', '8889');

Далее с помощью PHP кода обрабатывается форма авторизации, проверяется существование пользователя с соответствующими введёнными данными. После успешной регистрации пользователь будет перенаправлен на главную страницу сайта. Листинг:

if (isset($\_POST['login\_user'])) {

    $username = mysqli\_real\_escape\_string($db, $\_POST['username']);

    $password = mysqli\_real\_escape\_string($db, $\_POST['password']);

    if (empty($username)) {

        array\_push($errors, "Вы не указали имя");

    }

    if (empty($password)) {

        array\_push($errors, "Вы не указали пароль");

    }

    if (count($errors) == 0) {

        $query = "SELECT \* FROM users WHERE username='$username'";

        $results = mysqli\_query($db, $query);

        if (mysqli\_num\_rows($results) == 1) {

            $user = mysqli\_fetch\_assoc($results);

            if (password\_verify($password, $user['password'])) {

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

                $\_SESSION['username'] = $username;

                $\_SESSION['success'] = "Вход успешно выполнен";

                header('location: index.html');

            } else {

                array\_push($errors, "Неправильный логин/пароль");

            }

        } else {

            array\_push($errors, "Неправильный логин/пароль");

        }

    }

}

Листинг стилей для панели авторизации:

\* {

  margin: 0px;

  padding: 0px;

}

body {

  font-size: 120%;

  background: #333;

}

p {

  color: #ffffff;

}

p a {

  color: #696969;

}

p a:hover {

  color: #ffffff;

}

.header {

  width: 30%;

  margin: 50px auto 0px;

  color: white;

  background: #222;

  text-align: center;

  border: 1px solid #444;

  border-bottom: none;

  border-radius: 10px 10px 0px 0px;

  padding: 20px;

}

form, .content {

  width: 30%;

  margin: 0px auto;

  padding: 20px;

  border: 1px solid #444;

  background: #333;

  border-radius: 0px 0px 10px 10px;

}

.input-group {

  margin: 10px 0px 10px 0px;

}

.input-group label {

  display: block;

  text-align: left;

  margin: 3px;

  color: white;

}

.input-group input {

  height: 30px;

  width: 93%;

  padding: 5px 10px;

  font-size: 16px;

  border-radius: 5px;

  border: 1px solid gray;

  background: #444;

  color: white;

}

.btn {

  padding: 10px;

  font-size: 15px;

  color: rgb(255, 255, 255);

  background: #000000;

  border: none;

  border-radius: 8px;

  transition: color 0.5s, background 0.5s;

}

.btn:hover {

  color: #000000;

  background: rgb(255, 255, 255);

}

.error {

  width: 92%;

  margin: 0px auto;

  padding: 10px;

  border: 1px solid #3d3d3d;

  color: #3d3d3d;

  background: #3d3d3d;

  border-radius: 5px;

  text-align: left;

}

.success {

  color: #3c763d;

  background: #dff0d8;

  border: 1px solid #3c763d;

  margin-bottom: 20px;

}

## 2.6 Создание услуги в web-приложении

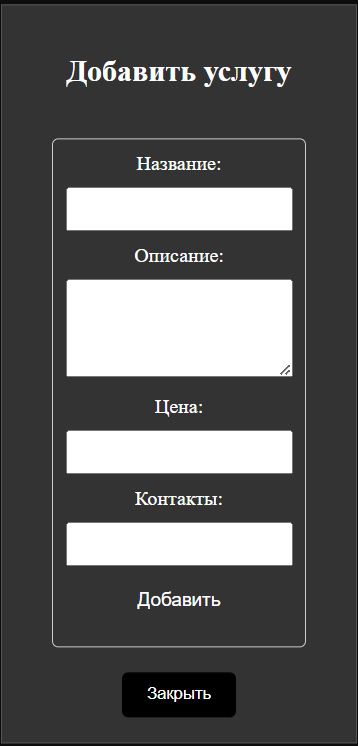


Рисунок 10 — Создание услуги в web-приложении

Листинг PHP страницы с отображением и созданием услуг

<?php

session\_start();

include('server.php');

if (isset($\_POST['add\_service'])) {

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

        $title = mysqli\_real\_escape\_string($db, $\_POST['title']);

        $description = mysqli\_real\_escape\_string($db, $\_POST['description']);

        $price = mysqli\_real\_escape\_string($db, $\_POST['price']);

        $contacts = mysqli\_real\_escape\_string($db, $\_POST['contacts']);

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

        if (empty($title) || empty($description) || empty($price) || empty($contacts)) {

            echo "Please fill in all the fields.";

        } else {

            $query = "INSERT INTO services (title, description, price, contacts, user\_id)

                      VALUES ('$title', '$description', '$price', '$contacts', '$user\_id')";

            $result = mysqli\_query($db, $query);

            if (!$result) {

                die("Query failed: " . mysqli\_error($db));

            }

        }

    } else {

        echo "User not logged in. Please log in and try again.";

    }

}

function displayUserServices($db, $user\_id) {

    $query = "SELECT services.\*, users.username FROM services JOIN users ON services.user\_id = users.id WHERE user\_id = '$user\_id'";

    $result = mysqli\_query($db, $query);

    while ($row = mysqli\_fetch\_assoc($result)) {

        echo "<div class='service-card'>";

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

        echo "<p><strong>Описание:</strong> " . $row['description'] . "</p>";

        echo "<p><strong>Цена:</strong> $" . $row['price'] . "</p>";

        echo "<p><strong>Пользователь:</strong> " . $row['username'] . "</p>";

        echo "<p><strong>Связаться:</strong> <a href='" . $row['contacts'] . "' target='\_blank'>Ссылка</a></p>";

        echo "</div>";

    }

}

function displayServices($db) {

    $query = "SELECT services.\*, users.username FROM services JOIN users ON services.user\_id = users.id";

    $result = mysqli\_query($db, $query);

    while ($row = mysqli\_fetch\_assoc($result)) {

        echo "<div class='service-card'>";

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

        echo "<p><strong>Описание:</strong> " . $row['description'] . "</p>";

        echo "<p><strong>Цена:</strong> $" . $row['price'] . "</p>";

        echo "<p><strong>Пользователь:</strong> " . $row['username'] . "</p>";

        echo "<p><strong>Связаться:</strong> <a href='" . $row['contacts'] . "' target='\_blank'>Ссылка</a></p>";

        echo "</div>";

    }

}

?>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <link rel="stylesheet" type="text/css" href="styles\style\_index.css">

    <header>

    <h1>Услуги</h1>

    <nav>

        <ul>

            <li><a href="index.html">Главная</a></li>

            <li><a href="uslugi.php">Услуги</a></li>

            <li><a href="profile.php">Личный кабинет</a></li>

        </ul>

    </nav>

</header>

    <title>Услуги</title>

</head>

<body>

   <form action="" method="post">

   <label for="filter">

    <input type="radio" name="filter" value="all" <?php if (!isset($\_POST['filter']) || $\_POST['filter'] === 'all') echo 'checked'; ?>>

    Все услуги

</label>

<label for="filter">

    <input type="radio" name="filter" value="user" <?php if (isset($\_POST['filter']) && $\_POST['filter'] === 'user') echo 'checked'; ?>>

    Мои услуги

</label>

        <input type="submit" value="Применить">

    </form>

   <div>

   <button onclick="openModal()">Добавить услугу</button>

   </div>

    <?php

    if (isset($\_POST['filter'])) {

        $filter = $\_POST['filter'];

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

            displayUserServices($db, $\_SESSION['user\_id']);

        } else {

            displayServices($db);

        }

    } else {

        displayServices($db);

    }

    ?>

    <div class="modal" id="myModal">

        <h2>Добавить услугу</h2>

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

            <label for="title">Название:</label>

            <input type="text" name="title" required><br>

            <label for="description">Описание:</label>

            <textarea name="description" rows="4" required></textarea><br>

            <label for="price">Цена:</label>

            <input type="number" name="price" step="0.01" required><br>

            <label for="contacts">Контакты:</label>

            <input type="text" name="contacts" required><br>

            <input type="submit" name="add\_service" value="Добавить" class="button">

        </form>

        <div>

        <button onclick="closeModal()">Закрыть</button>

        </div>

    </div>

    <div class="modal-overlay" id="overlay" onclick="closeModal()"></div>

    <script src="scripts\script.js"></script>

</body>

</html>

После создания услуги все данные отправляются на сервер. Мы получаем ID пользователя, название услуги, описание услуги, цену и контакты. (Рисунок 11). Есди все поля была заполнены, услуга отображается на сайте, так же есть фильтрация – можно отобразить все услуги или услуги созданные вами. Листинг:

<?php

session\_start();

include('server.php');

if (isset($\_POST['add\_service'])) {

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

        $title = mysqli\_real\_escape\_string($db, $\_POST['title']);

        $description = mysqli\_real\_escape\_string($db, $\_POST['description']);

        $price = mysqli\_real\_escape\_string($db, $\_POST['price']);

        $contacts = mysqli\_real\_escape\_string($db, $\_POST['contacts']);

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

        if (empty($title) || empty($description) || empty($price) || empty($contacts)) {

            echo "Please fill in all the fields.";

        } else {

            $query = "INSERT INTO services (title, description, price, contacts, user\_id)

                      VALUES ('$title', '$description', '$price', '$contacts', '$user\_id')";

            $result = mysqli\_query($db, $query);

            if (!$result) {

                die("Query failed: " . mysqli\_error($db));

            }

        }

    } else {

        echo "User not logged in. Please log in and try again.";

    }

}

function displayUserServices($db, $user\_id) {

    $query = "SELECT services.\*, users.username FROM services JOIN users ON services.user\_id = users.id WHERE user\_id = '$user\_id'";

    $result = mysqli\_query($db, $query);

    while ($row = mysqli\_fetch\_assoc($result)) {

        echo "<div class='service-card'>";

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

        echo "<p><strong>Описание:</strong> " . $row['description'] . "</p>";

        echo "<p><strong>Цена:</strong> $" . $row['price'] . "</p>";

        echo "<p><strong>Пользователь:</strong> " . $row['username'] . "</p>";

        echo "<p><strong>Связаться:</strong> <a href='" . $row['contacts'] . "' target='\_blank'>Ссылка</a></p>";

        echo "</div>";

    }

}

function displayServices($db) {

    $query = "SELECT services.\*, users.username FROM services JOIN users ON services.user\_id = users.id";

    $result = mysqli\_query($db, $query);

    while ($row = mysqli\_fetch\_assoc($result)) {

        echo "<div class='service-card'>";

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

        echo "<p><strong>Описание:</strong> " . $row['description'] . "</p>";

        echo "<p><strong>Цена:</strong> $" . $row['price'] . "</p>";

        echo "<p><strong>Пользователь:</strong> " . $row['username'] . "</p>";

        echo "<p><strong>Связаться:</strong> <a href='" . $row['contacts'] . "' target='\_blank'>Ссылка</a></p>";

        echo "</div>";

    }

}

?>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <link rel="stylesheet" type="text/css" href="styles\style\_index.css">

    <header>

    <h1>Услуги</h1>

    <nav>

        <ul>

            <li><a href="index.html">Главная</a></li>

            <li><a href="uslugi.php">Услуги</a></li>

            <li><a href="profile.php">Личный кабинет</a></li>

        </ul>

    </nav>

</header>

    <title>Услуги</title>

</head>

<body>

   <form action="" method="post">

   <label for="filter">

    <input type="radio" name="filter" value="all" <?php if (!isset($\_POST['filter']) || $\_POST['filter'] === 'all') echo 'checked'; ?>>

    Все услуги

</label>

<label for="filter">

    <input type="radio" name="filter" value="user" <?php if (isset($\_POST['filter']) && $\_POST['filter'] === 'user') echo 'checked'; ?>>

    Мои услуги

</label>

        <input type="submit" value="Применить">

    </form>

   <div>

   <button onclick="openModal()">Добавить услугу</button>

   </div>

    <?php

    if (isset($\_POST['filter'])) {

        $filter = $\_POST['filter'];

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

            displayUserServices($db, $\_SESSION['user\_id']);

        } else {

            displayServices($db);

        }

    } else {

        displayServices($db);

    }

    ?>

    <div class="modal" id="myModal">

        <h2>Добавить услугу</h2>

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

            <label for="title">Название:</label>

            <input type="text" name="title" required><br>

            <label for="description">Описание:</label>

            <textarea name="description" rows="4" required></textarea><br>

            <label for="price">Цена:</label>

            <input type="number" name="price" step="0.01" required><br>

            <label for="contacts">Контакты:</label>

            <input type="text" name="contacts" required><br>

            <input type="submit" name="add\_service" value="Добавить" class="button">

        </form>

        <div>

        <button onclick="closeModal()">Закрыть</button>

        </div>

    </div>

    <div class="modal-overlay" id="overlay" onclick="closeModal()"></div>

    <script src="scripts\script.js"></script>

</body>

</html>

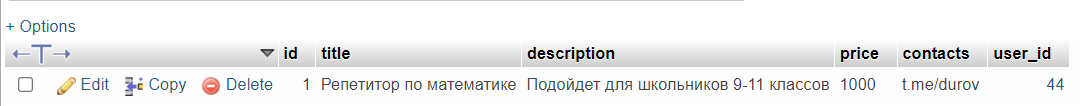


Рисунок 11 — Данные о услугах на сервере

Листинг стилей для страницы с услугами.

body {

    margin: 0;

    background-color: #1a1a1a;

    color: #fff;

    text-align: center;

}

header {

    background-color: #333;

    color: white;

    padding: 20px;

    display: flex;

    justify-content: space-between;

    align-items: center;

}

nav {

    display: flex;

    justify-content: flex-end;

    align-items: center;

    margin-left: auto;

}

ul {

    list-style-type: none;

    margin: 0;

    padding: 0;

    display: flex;

}

li {

    margin-left: 20px;

}

a {

    text-decoration: none;

    color: white;

    font-size: 20px;

}

a:hover {

    text-decoration: underline;

}

h1 {

    margin: 0;

    color: #ffffff;

}

.service-card {

    border: 1px solid #333;

    padding: 20px;

    margin: 20px;

    background-color: #333;

    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

    border-radius: 8px;

    text-align: left;

}

.service-card h3 {

    color: #fff;

    margin-bottom: 10px;

}

p {

    margin: 10px 0;

}

strong {

    font-weight: bold;

}

input.button {

    padding: 10px;

    font-size: 15px;

    color: black;

    background: #fff;

    border: none;

    border-radius: 8px;

    transition: color 0.5s, background 0.5s;

}

input.button:hover {

    color: #fff;

    background: black;

}

button {

    background-color: #000000;

    color: #fff;

    padding: 10px 20px;

    border: none;

    border-radius: 5px;

    cursor: pointer;

    margin-bottom: auto;

    transition: background-color 0.3s;

}

button:hover {

    background-color: #fff;

    color: #000;

}

.modal {

    display: none;

    position: fixed;

    top: 50%;

    left: 50%;

    transform: translate(-50%, -50%);

    background-color: #333;

    padding: 20px;

    border: 1px solid #555;

    z-index: 1;

    color: #fff;

}

.modal-overlay {

    display: none;

    position: fixed;

    top: 0;

    left: 0;

    width: 100%;

    height: 100%;

    background-color: rgba(0, 0, 0, 0.5);

    z-index: 0;

}

.modal label {

    display: block;

    margin-bottom: 10px;

}

.modal input,

.modal textarea {

    width: 100%;

    padding: 8px;

    margin-bottom: 10px;

    box-sizing: border-box;

}

.modal button {

    background-color: #000;

    color: #fff;

    padding: 10px 20px;

    border: none;

    border-radius: 5px;

    cursor: pointer;

    transition: background-color 0.3s;

}

.modal button:hover {

    background-color: #fff;

    color: #000;

}

.container {

    position: absolute;

    left: 0;

    top: 0;

    width: 100%;

}

form {

    margin: 20px;

    padding: 10px;

    border: 1px solid #ccc;

    border-radius: 5px;

    display: inline-block;

}

label {

    display: block;

    margin-bottom: 10px;

}

input[type="radio"] {

    margin-right: 5px;

}

input[type="submit"] {

    background-color: #333;

    color: white;

    padding: 8px 15px;

    border: none;

    border-radius: 3px;

    cursor: pointer;

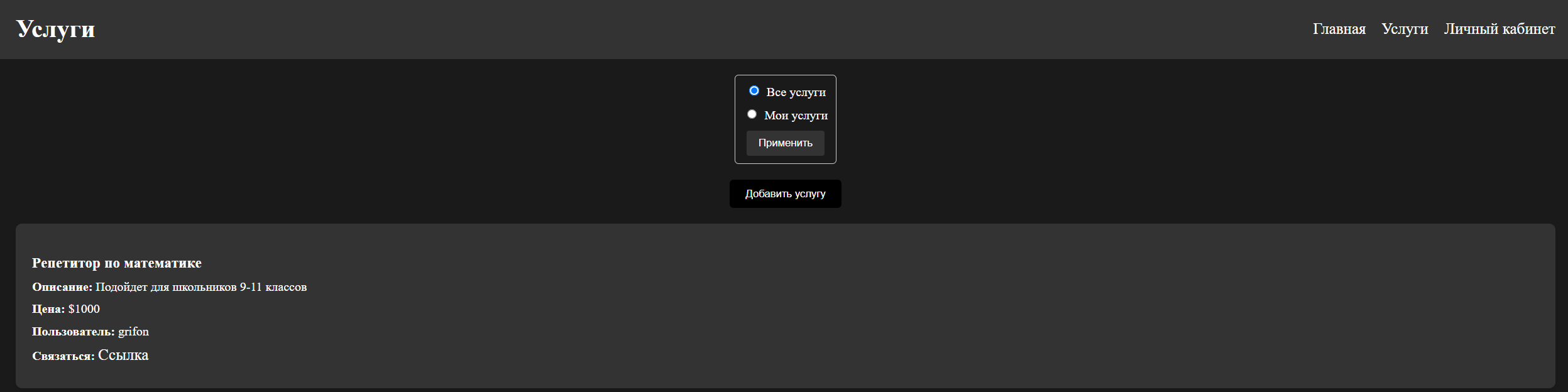
}

input[type="submit"]:hover {

    background-color: #555;

}

## 2.7 Страница с услугами в web-приложении



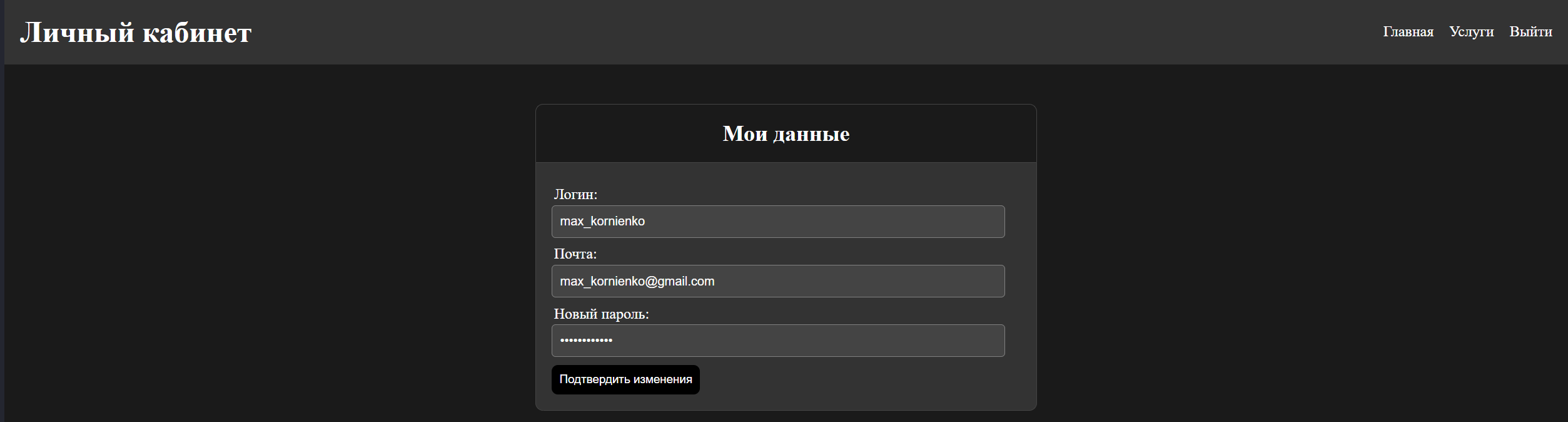


Рисунок 12 — Личный кабинет пользователя в web-приложении

Листинг стилей для личного кабинета пользователя:

\* {

    margin: 0;

    padding: 0;

}

body {

    font-size: 120%;

    background: #1a1a1a;

}

.header1 {

    width: 30%;

    margin: 50px auto 0;

    color: white;

    background: #1a1a1a;

    text-align: center;

    border: 1px solid #444;

    border-bottom: none;

    border-radius: 10px 10px 0 0;

    padding: 20px;

}

.header2 {

    background-color: #333;

    color: white;

    padding: 20px;

    display: flex;

    justify-content: space-between;

    align-items: center;

}

nav {

    display: flex;

    justify-content: flex-end;

    align-items: center;

    margin-left: auto;

}

ul {

    list-style-type: none;

    margin: 0;

    padding: 0;

    display: flex;

}

li {

    margin-left: 20px;

}

a {

    text-decoration: none;

    color: white;

}

a:hover {

    text-decoration: underline;

}

h1 {

    margin: 0;

    color: #ffffff;

    margin-right: auto;

}

form, .content {

    width: 30%;

    margin: 0 auto;

    padding: 20px;

    border: 1px solid #444;

    background: #333;

    border-radius: 0 0 10px 10px;

}

.input-group {

    margin: 10px 0;

}

.input-group label {

    display: block;

    text-align: left;

    margin: 3px;

    color: white;

}

.input-group input {

    height: 30px;

    width: 93%;

    padding: 5px 10px;

    font-size: 16px;

    border-radius: 5px;

    border: 1px solid gray;

    background: #444;

    color: white;

}

.btn {

    padding: 10px;

    font-size: 15px;

    color: rgb(255, 255, 255);

    background: #000000;

    border: none;

    border-radius: 8px;

    transition: color 0.5s, background 0.5s;

}

.btn:hover {

    color: #000000;

    background: rgb(255, 255, 255);

}

.error, .success {

    width: 92%;

    margin: 0 auto;

    padding: 10px;

    border-radius: 5px;

    text-align: left;

}

.error {

    border: 1px solid #a94442;

    color: #a94442;

    background: #f2dede;

}

.success {

    border: 1px solid #3c763d;

    color: #3c763d;

    background: #dff0d8;

    margin-bottom: 20px;

}

С помощью PHP кода отображаем информацию о пользователе. На web-странице возле имени есть кнопка «изменить», при нажатии на которую можно изменить логин, почту или пароль. Листинг формы обновления имени пользователя:

<!-- Добавляем кнопку "Изменить" и форму для изменения имени -->

    <button onclick="showUpdateUsernameForm()">Изменить</button>

<form id="updateUsernameForm" style="display: none;">

    <label for="newUsername">Новое имя пользователя:</label>

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

    <button type="button" onclick="updateUsername()">Сохранить</button>

</form>

</p>

<p>Роль: <?php echo ucfirst($roleName); ?></p>

Функционал обычного пользователя закончен, далее будет представлен функционал администратора.

## 2.2.7 Функционал администратора

С помощью PHP кода проверяется, соответствует ли название пользователя роли администратора. При выполнении данного условия администратор получает список всех пользователей и всех услу. Листинг:

$is\_admin = ($username === 'admin');

    </ul>

Администратор может добавлять/удалять пользователей или услуги. Листинг:

<?php

session\_start();

include('server.php');

function displayServices($db) {

    $query = "SELECT services.\*, users.id as user\_id, users.username FROM services JOIN users ON services.user\_id = users.id";

    $result = mysqli\_query($db, $query);

    while ($row = mysqli\_fetch\_assoc($result)) {

        echo "<div class='service-card'>";

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

        echo "<p><strong>ID услуги:</strong> " . $row['id'] . "</p>";

        echo "<p><strong>Описание:</strong> " . $row['description'] . "</p>";

        echo "<p><strong>Цена:</strong> $" . $row['price'] . "</p>";

        echo "<p><strong>Пользователь:</strong> " . $row['username'] . " (ID: " . $row['user\_id'] . ")</p>";

        echo "<p><strong>Связаться:</strong> <a href='" . $row['contacts'] . "' target='\_blank'>Ссылка</a></p>";

        echo "</div>";

    }

}

function displayAllUsers($db) {

    $query = "SELECT \* FROM users";

    $result = mysqli\_query($db, $query);

    while ($row = mysqli\_fetch\_assoc($result)) {

        echo "<div class='service-card'>";

        echo "<p><strong>ID пользователя:</strong> " . $row['id'] . "</p>";

        echo "<p><strong>Имя пользователя:</strong> " . $row['username'] . "</p>";

        echo "<p><strong>Email:</strong> " . $row['email'] . "</p>";

        echo "</div>";

    }

}

if (isset($\_POST['add\_service'])) {

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

        $title = mysqli\_real\_escape\_string($db, $\_POST['title']);

        $description = mysqli\_real\_escape\_string($db, $\_POST['description']);

        $price = mysqli\_real\_escape\_string($db, $\_POST['price']);

        $contacts = mysqli\_real\_escape\_string($db, $\_POST['contacts']);

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

        if (empty($title) || empty($description) || empty($price) || empty($contacts)) {

            echo "Пожалуйста, заполните все поля.";

        } else {

            $query = "INSERT INTO services (title, description, price, contacts, user\_id)

                      VALUES ('$title', '$description', '$price', '$contacts', '$user\_id')";

            $result = mysqli\_query($db, $query);

        }

    }

}

if (isset($\_POST['add\_user'])) {

    $login = mysqli\_real\_escape\_string($db, $\_POST['login']);

    $email = mysqli\_real\_escape\_string($db, $\_POST['mail']);

    $password = mysqli\_real\_escape\_string($db, $\_POST['password']);

    if (empty($login) || empty($email) || empty($password)) {

        echo "Пожалуйста, заполните все поля.";

    } else {

        $query = "INSERT INTO users (username, email, password)

                  VALUES ('$login', '$email', '$password')";

        $result = mysqli\_query($db, $query);

    }

}

if (isset($\_POST['delete\_service'])) {

    $serviceNumber = mysqli\_real\_escape\_string($db, $\_POST['number']);

    if (empty($serviceNumber)) {

        echo "Пожалуйста, введите номер услуги.";

    } else {

        $query = "DELETE FROM services WHERE id = '$serviceNumber'";

        $result = mysqli\_query($db, $query);

    }

}

if (isset($\_POST['delete\_user'])) {

    $userNumber = mysqli\_real\_escape\_string($db, $\_POST['number']);

    if (empty($userNumber)) {

        echo "Пожалуйста, введите номер пользователя.";

    } else {

        $query = "DELETE FROM users WHERE id = '$userNumber'";

        $result = mysqli\_query($db, $query);

    }

}

?>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Список услуг</title>

    <link rel="stylesheet" type="text/css" href="styles\style\_admin.css">

</head>

<header>

    <h1>Панель администратора</h1>

    <nav>

        <ul>

        <li><a href="index.html">Главная</a></li>

            <li><a href="uslugi.php">Услуги</a></li>

            <li><a href="profile.php">Личный кабинет</a></li>

        </ul>

    </nav>

</header>

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

        <label for="filter">

            <input type="radio" name="filter" value="all" <?php if (!isset($\_POST['filter']) || $\_POST['filter'] === 'all') echo 'checked'; ?>>

            Услуги

        </label>

        <label for="filter">

            <input type="radio" name="filter" value="user" <?php if (isset($\_POST['filter']) && $\_POST['filter'] === 'user') echo 'checked'; ?>>

            Пользователи

        </label>

        <input type="submit" value="Применить">

    </form>

    <?php

    if (isset($\_POST['filter'])) {

        $filter = $\_POST['filter'];

        if ($filter === 'user') {

            ?>

            <div>

                <button onclick="openModaluser()">Добавить</button>

                <button onclick="openModaldeleteuser()">Удалить</button>

            </div>

            <?php

        } else {

            ?>

            <div>

                <button onclick="openModaluslugi()">Добавить</button>

                <button onclick="openModaldeleteuslugi()">Удалить</button>

            </div>

            <?php

        }

    }

    ?>

<?php

if (isset($\_POST['filter'])) {

    $filter = $\_POST['filter'];

    if ($filter === 'user') {

        displayAllUsers($db);

    } else {

        displayServices($db);

    }

}

?>

<div class="modal" id="uslugi">

        <h2>Добавить услугу</h2>

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

            <label for="title">Название:</label>

            <input type="text" name="title" required><br>

            <label for="description">Описание:</label>

            <textarea name="description" rows="4" required></textarea><br>

            <label for="price">Цена:</label>

            <input type="number" name="price" step="0.01" required><br>

            <label for="contacts">Контакты:</label>

            <input type="text" name="contacts" required><br>

            <input type="submit" name="add\_service" value="Добавить">

        </form>

        <div>

            <button onclick="closeModaluslugi()">Закрыть</button>

        </div>

    </div>

    <div class="modal" id="user">

        <h2>Добавить пользователя</h2>

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

            <label for="login">Логин:</label>

            <input type="text" name="login" required><br>

            <label for="mail">Почта:</label>

            <textarea name="mail" rows="1" required></textarea><br>

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

            <textarea name="password" rows="1" required></textarea><br>

            <input type="submit" name="add\_user" value="Добавить">

        </form>

        <div>

            <button onclick="closeModaluser()">Закрыть</button>

        </div>

    </div>

    <div class="modal" id="deleteuslugi">

        <h2>Удалить услугу</h2>

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

            <label for="number">ID Услуги:</label>

            <input type="text" name="number" required><br>

            <input type="submit" name="delete\_service" value="Удалить">

        </form>

        <div>

            <button onclick="closeModaldeleteuslugi()">Закрыть</button>

        </div>

    </div>

    <div class="modal" id="deleteuser">

        <h2>Удалить пользователя</h2>

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

            <label for="number">ID Пользователя:</label>

            <input type="text" name="number" required><br>

            <input type="submit" name="delete\_user" value="Удалить">

        </form>

        <div>

            <button onclick="closeModaldeleteuser()">Закрыть</button>

        </div>

    </div>

    <div class="modal-overlay" id="overlay" onclick="closeModaluslugi()"></div>

    <div class="modal-overlay" id="overlay" onclick="closeModaluser()"></div>

    <div class="modal-overlay" id="overlay" onclick="closeModaldeleteuslugi()"></div>

    <div class="modal-overlay" id="overlay" onclick="closeModaldeleteuser()"></div>

    <script src="scripts\admin.js"></script>

</body>

</html>

Также было реализовано отображение информации о пользователе в отдельном окне.

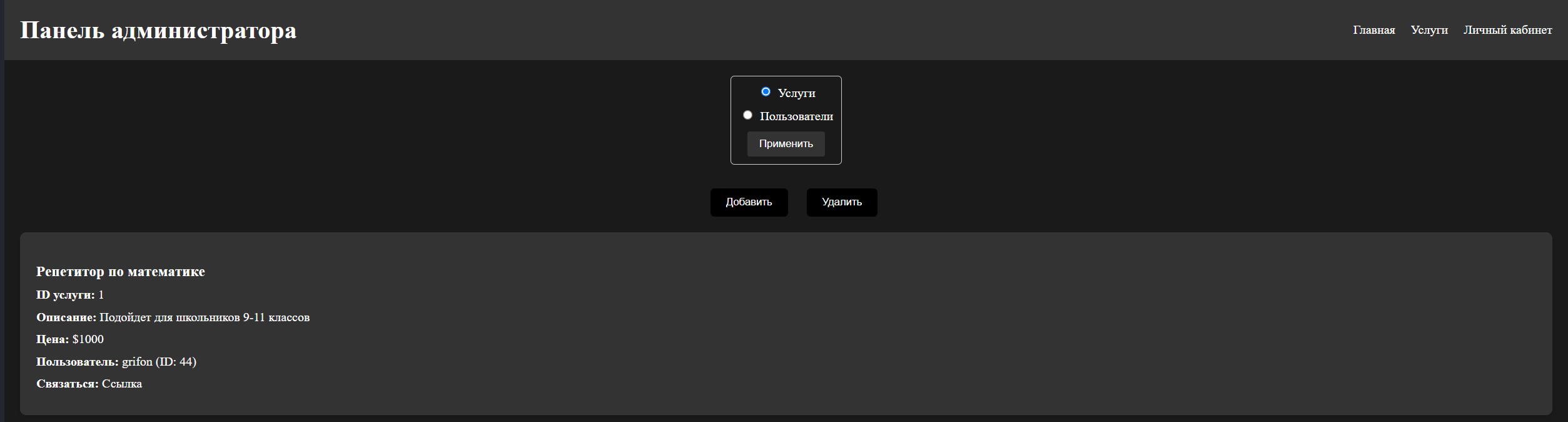
****

Рисунок 13 — отображение информации о услугах

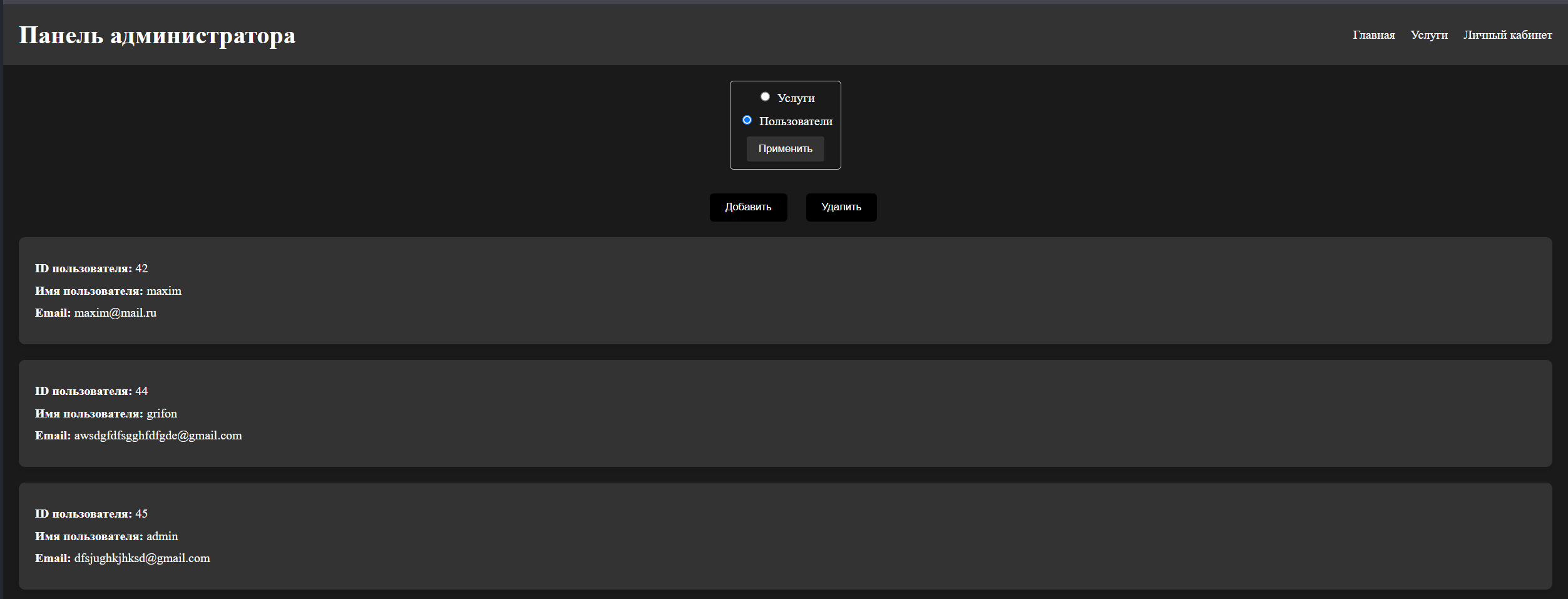


Рисунок 14 – отображение информации о пользователях

У нас есть такой html код модального окна, для добавления пользователей и услуг.. Листинг:

<div class="modal" id="uslugi">

        <h2>Добавить услугу</h2>

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

            <label for="title">Название:</label>

            <input type="text" name="title" required><br>

            <label for="description">Описание:</label>

            <textarea name="description" rows="4" required></textarea><br>

            <label for="price">Цена:</label>

            <input type="number" name="price" step="0.01" required><br>

            <label for="contacts">Контакты:</label>

            <input type="text" name="contacts" required><br>

            <input type="submit" name="add\_service" value="Добавить">

        </form>

        <div>

            <button onclick="closeModaluslugi()">Закрыть</button>

        </div>

    </div>

    <div class="modal" id="user">

        <h2>Добавить пользователя</h2>

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

            <label for="login">Логин:</label>

            <input type="text" name="login" required><br>

            <label for="mail">Почта:</label>

            <textarea name="mail" rows="1" required></textarea><br>

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

            <textarea name="password" rows="1" required></textarea><br>

            <input type="submit" name="add\_user" value="Добавить">

        </form>

        <div>

            <button onclick="closeModaluser()">Закрыть</button>

        </div>

    </div>

# ЗАКЛЮЧЕНИЕ

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

Web-приложение в сфере заказов одежды с различных стран онлайн имеет важную роль в улучшении жизни людей, т.к. из-за границы одежда выходит дешевле и выбор больше.

Одна из основных функций такого приложения - обеспечение удобства и доступности оформления заказа одежды для клиентов. Пользователи могут удобно выбирать модель и размер, которые им необходимы. Кроме того, клиенты могут узнать о цене, и осуществить заявку для заказа в режиме реального времени.

Для владельцев онлайн магазинов web-приложение предоставляет мощный инструмент управления заказами. Они могут легко контролировать наличие вещей в режиме реального времени, управлять заказами и их оформлениями. Приложение также облегчает взаимодействие с клиентами, позволяя удобно управлять своими заказами.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

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