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**

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

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

подпись, дата

Студент УБ22-08Б, 432218328 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.М. Алипов

подпись, дата

Красноярск 2023

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc155905984)

[1 Теоретическая часть 4](#_Toc155905985)

[1.1 HTML и CSS 4](#_Toc155905986)

[1.2 JavaScript и PHP 6](#_Toc155905987)

[1.3 MySQL 8](#_Toc155905988)

[2 Практическая часть 10](#_Toc155905989)

[2.1 Требования, базовая конфигурация и база данных 10](#_Toc155905990)

[2.2 Разработка формы регистрации 12](#_Toc155905991)

[2.3 Главная страница и личный кабинет 17](#_Toc155905992)

[2.4 Панель администратора 26](#_Toc155905993)

[2.5 Страницы с основным контентом 30](#_Toc155905994)

[ЗАКЛЮЧЕНИЕ 35](#_Toc155905995)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 36](#_Toc155905996)

# ВВЕДЕНИЕ

Актуальность курсовой работы заключается в создании современного web-приложения, специализированного на области книжной тематики. Сегодняшнее время характеризуется стремительным развитием интернет-технологий и повышенным спросом на удобные и эффективные решения в сфере чтения и доступа к книгам.

Цель данного проекта состоит в разработке web-приложения, которое предоставляет пользователям удобную и интуитивно понятную платформу для чтения книг. Главной идеей является создание системы, которая предоставляет доступ к различным литературным произведениям, является мини-рекламой более широкой платформы.

В процессе выполнения курсовой работы будут использованы современные web-технологии, такие как HTML5, CSS, JavaScript, PHP, MySQL. Благодаря использованию этих инструментов, разрабатываемое приложение будет отвечать современным требованиям производительности, масштабируемости и безопасности.

Ожидается, что результатом данной курсовой работы будет полноценное web-приложение, способное удовлетворить потребности пользователей в сфере чтения книг. Разработанное приложение будет иметь потенциал для дальнейшего развития и расширения функциональности в соответствии с потребностями пользователей.

В данном отчете будут подробно описаны этапы разработки приложения, выбранные технологии и инструменты, а также полученные результаты и возможные перспективы развития проекта.

# 1 Теоретическая часть

## **1.1 HTML и CSS**

HTML (HyperText Markup Language) - это основный язык разметки веб-страниц, который используется для создания структуры и содержания веб-страниц.

Плюсы:

1. Простота использования: HTML достаточно просто изучить и понять, поскольку он использует простую структуру тегов.

2. Универсальность: HTML поддерживается всеми веб-браузерами и является стандартом для создания веб-страниц.

3. Легкость в поддержке и изменении: Поскольку HTML используется для создания структуры веб-страниц, он легко поддерживается и изменяется без дополнительных трудностей.

Недостатки:

1. Ограниченные возможности стилизации: HTML не предоставляет обширных возможностей для стилизации веб-страниц, поэтому для создания более сложных дизайнов может потребоваться использование дополнительных языков (например, CSS).

2. Отсутствие динамического контента: HTML предназначен для создания статических веб-страниц, поэтому для добавления динамического контента (например, анимации или интерактивных элементов) может потребоваться использование JavaScript.

3. Необходимость в дополнительных языках: Для создания полноценных и интерактивных веб-страниц часто требуется комбинировать HTML с другими языками (например, CSS, JavaScript, PHP), что может усложнить процесс разработки.

CSS (Cascading Style Sheets) - это язык стилей, который используется для определения внешнего вида веб-страниц, таких как шрифты, цвета, отступы, размеры и многое другое

Плюсы:

1. Раздельное оформление: CSS позволяет разделять структуру и визуальное оформление веб-страницы, что делает код более чистым и удобным для поддержки.

2. Гибкость и масштабируемость: CSS позволяет легко изменять внешний вид веб-страницы, а также создавать масштабируемые и адаптивные дизайны для различных устройств.

3. Возможности анимации и трансформации: CSS позволяет добавлять анимацию, трансформации и другие эффекты к элементам веб-страниц, что делает их более привлекательными.

Недостатки:

1. Совместимость с браузерами: Иногда возникают проблемы с совместимостью CSS между различными веб-браузерами, что может приводить к различиям в отображении веб-страниц.

2. Сложность некоторых эффектов: Для создания сложных анимаций или эффектов потребуется глубокое понимание CSS и навыки программирования.

3. Загрузка страницы: Использование большого количества CSS-стилей или изображений может замедлить загрузку веб-страницы, что может негативно повлиять на пользовательский опыт.

## 

## **1.2 JavaScript и PHP**

JavaScript - это высокоуровневый язык программирования, который применяется для создания интерактивных элементов на веб-страницах и для управления поведением веб-приложений.

Плюсы:

1. Интерактивность: JavaScript позволяет создавать интерактивные элементы на веб-страницах, такие как формы, игры, анимации, динамические меню и многое другое.

2. Взаимодействие с пользователем: JavaScript позволяет обрабатывать действия пользователя, такие как щелчки мышью, нажатия клавиш, отправку форм и другие события.

3. Возможности асинхронной загрузки данных: JavaScript используется для работы с AJAX (асинхронным JavaScript и XML), что позволяет загружать данные с сервера без перезагрузки страницы.

Недостатки:

1. Безопасность: JavaScript может быть уязвим для атак, таких как инъекции кода или вредоносные сценарии, что делает его потенциально опасным для веб-приложений.

2. Совместимость с браузерами: Некоторые функции JavaScript могут работать по-разному в различных веб-браузерах, что требует дополнительной работы по обеспечению совместимости.

3. Загруженность страницы: Использование большого количества JavaScript-кода или библиотек может привести к увеличению времени загрузки веб-страницы и негативно сказаться на пользовательском опыте.

PHP (Hypertext Preprocessor) - это скриптовый язык программирования, который обычно используется для разработки веб-приложений и динамически генерируемых веб-страниц. Он часто применяется для работы с базами данных, обработки форм, создания сессий и многое другое.

Плюсы:

1. Универсальность: PHP может быть использован для разработки различных типов веб-приложений, начиная от простых статических страниц и заканчивая сложными веб-порталами и интернет-магазинами.

2. Интеграция с базами данных: PHP имеет широкие возможности для работы с базами данных, такими как MySQL, PostgreSQL и другими, что делает его удобным для создания динамических веб-приложений.

3. Гибкость: PHP позволяет внедрять динамический контент на веб-страницы, обрабатывать формы, управлять файлами и каталогами, отправлять электронные письма и многое другое.

Недостатки:

1. Безопасность: Неправильное использование PHP может привести к уязвимостям безопасности, таким как инъекции SQL, кросс-сайтовые сценарии и другие атаки.

2. Производительность: Некоторые критики утверждают, что PHP может быть менее эффективным с точки зрения производительности по сравнению с некоторыми другими языками программирования.

3. Сложность масштабирования: Некоторые большие и сложные веб-приложения, разработанные с использованием PHP, могут быть сложными в обслуживании и масштабировании из-за возросшей сложности структуры кода.

## 

## **1.3 MySQL**

MySQL - это система управления базами данных (СУБД), которая широко используется для хранения и управления структурированными данными в веб-приложениях.

Плюсы:

1. Эффективность: MySQL обеспечивает высокую производительность при работе с большими объемами данных, и может обрабатывать тысячи запросов в секунду.

2. Поддержка различных типов данных: MySQL поддерживает широкий спектр типов данных, таких как числа, текст, дата/время, изображения и другие, что делает его удобным для хранения и обработки различных типов информации.

3. Масштабируемость: MySQL удобно масштабируется как вертикально (увеличение мощности сервера), так и горизонтально (добавление новых узлов в кластер).

Недостатки:

1. Ограниченная поддержка хранимых процедур и триггеров: Некоторые критики указывают на ограниченную поддержку хранимых процедур и триггеров в MySQL по сравнению с некоторыми другими СУБД.

2. Недостаточная надежность: Некоторые замечания относительно стабильности и надежности MySQL, особенно в сложных и масштабных проектах или при большом объеме запросов.

3. Ограниченная поддержка больших объемов данных: В некоторых сценариях MySQL может показать недостаточную эффективность при работе с очень большими объемами данных, что требует тщательного планирования и настройки.

# 2 Практическая часть

## **2.1 Требования, базовая конфигурация и база данных**

Перед тем, как приступить к разработке web-приложения, необходимо разобраться с тем, что от нас требуется. К данной работе предъявлено 8 требований:

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. Неавторизированные 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.

После ознакомления с требованиями к работе, необходимо настроить базовую конфигурацию для корректной работы приложения. Для этого мы будем использовать приложение AMPPS. После скачивания файла с официального сайта и его установки, все необходимые программы будут загружены. После установки мы открываем приложение AMPPS и нажимаем на иконку домика, которая перенаправит нас на страницу со всеми доступными функциями. Далее мы находим "Database Tools" и "MySQL Password", где мы устанавливаем необходимый пароль.

Чтобы создать файлы и далее работать с ними, необходимо в приложении AMPPS нажать на иконку папки, что автоматически перенаправит нас в папку "www", где должны находиться файлы для работы. После выполнения этих действий мы можем считать, что базовая конфигурация задана.

Далее мы перейдем к созданию базы данных с использованием phpMyAdmin. Для нашей работы нам потребуется база данных "cars" с таблицей "users", состоящей из 4 столбцов. Следует отметить, что данное приложение будет представлять собой площадку для продажи автомобилей, и список автомобилей не планируется изменять - он будет фиксированным.

Далее мы перейдем к созданию таблицы "users". В этой таблице нам нужно будет создать 4 столбца: id, login, password, email. На рисунке 2.1.1 показана данная таблица со всеми необходимыми параметрами.



Рисунок 2.1.1 – Таблица users в базе данных cars

Мы изучили требования к проекту, настроили все необходимые параметры для запуска приложения и создали базу данных. Теперь можем приступить к написанию кода.

## **2.2 Разработка формы регистрации**

Форма регистрации является обязательным компонентом нашего проекта в связи с требованиями к работе, предполагающими наличие системы авторизации и аутентификации пользователей. Для этого создадим php файл под названием logReg.php, в котором будет размещена форма регистрации.

Мы разработаем форму с двумя опциями: одна будет использоваться для регистрации, а другая - для входа. Для этого нам потребуется язык HTML для создания формы и JavaScript для обработки нажатия кнопки и изменения формы.

Листинг HTML для формы регистрации:

<div class="registration-form">

  <?php if (!isset($\_POST['loginForm'])) { ?>

    <form class="logReg\_\_form" method="POST" action="">

      <input class="logReg\_\_form-input" type="text" name="login"  placeholder="Логин" required><br><br>

      <input class="logReg\_\_form-input" type="password" name="password"  placeholder="Пароль" required><br><br>

      <input class="logReg\_\_form-input" type="email" name="email"  placeholder="Email" required><br><br>

      <button class="logReg\_\_form-button" name="register">Зарегистрироваться</button><br><br>

      <p class="logReg\_\_form-link">Уже зарегистрированы? <a href="#" onclick="showLoginForm()">Войти</a></p>

    </form>

  <?php } elseif (isset($\_POST['loginForm'])) { ?>

    <form method="POST" action="">

      <input type="text" name="login2" placeholder="Логин" class="logReg\_\_form-input" required><br><br>

      <input type="password" name="password2" placeholder="Пароль" class="logReg\_\_form-input" required><br><br>

      <button class="logReg\_\_form-button" name="loggin">Войти</button><br><br>

      <p class="logReg\_\_form-link">Нет аккаунта? <a href="#" onclick="showRegistrationForm()">Зарегистрироваться</a></p>

    </form>

  <?php } ?>

</div>

Листинг JavaScript для формы регистрации:

<script>

  function showLoginForm() {

    var registrationForm = document.querySelector(".registration-form");

    registrationForm.innerHTML = `

      <form class="logReg\_\_form" method="POST" action="">

        <input type="text" name="login" placeholder="Логин" class="logReg\_\_form-input" required><br><br>

        <input type="password" name="password" placeholder="Пароль" class="logReg\_\_form-input" required><br><br>

        <button class="logReg\_\_form-button" name="loggin">Войти</button><br><br>

        <p class="logReg\_\_form-link">Нет аккаунта? <a href="#" onclick="showRegistrationForm()">Зарегистрироваться</a></p>

      </form>

    `;

  }

  function showRegistrationForm() {

    var registrationForm = document.querySelector(".registration-form");

    registrationForm.innerHTML = `

      <form class="logReg\_\_form" method="POST" action="">

        <input type="text" name="login" placeholder="Логин" class="logReg\_\_form-input" required><br><br>

        <input type="password" name="password" placeholder="Пароль" class="logReg\_\_form-input" required><br><br>

        <input type="mail" name="email" placeholder="Email" class="logReg\_\_form-input" required><br><br>

        <button class="logReg\_\_form-button" name="register">Зарегистрироваться</button><br><br>

        <p class="logReg\_\_form-link">Уже зарегистрированы? <a href="#" onclick="showLoginForm()">Войти</a></p>

      </form>

    `;

  }

</script>

Помимо размещения формы, мы разместим на странице кнопки "Главная" и "Машины", для выполнения требования, согласно которому неавторизованные пользователи могут просматривать предоставляемые услуги. После создания основной структуры нашей страницы, необходимо создать стили для обеспечения ее эстетичного вида и корректного отображения на любом разрешении экрана.

Листинг CSS для страницы с формой регистрации:

.registration-form{

    margin-top: 70px;

    margin-bottom: 150px;

}

.logReg\_\_form{

    width: 300px;

    margin: 0 auto;

}

.logReg\_\_form-input{

    width: 100%;

    font-size: 18px;

    background-color: #353535;

    color: #fff;

    padding: 10px 20px;

}

.logReg\_\_form-button{

    background-color: #EE8313;

    padding: 10px 20px;

    color: #fff;

    width: 100%;

    font-size: 18px;

    font-weight: 600;

    transition: all 0.3s;

}

.logReg\_\_form-button:hover{

    background-color: #8f4d06;

}

.logReg\_\_form-link{

    text-align: center;

}

.logReg\_\_form-link>a{

    color: #EE8313;

    transition: all 0.3s;

}

.logReg\_\_form-link>a:hover{

    color: #bf6506;

}

.registration-header {

    font-size: 36px;

    font-weight: bold;

    margin-top: 20px;

}

.reg\_\_title {

    font-size: 48px;

    margin-bottom: 40px;

    text-align: center;

}

Излишним будет представлять листинг кода стилей, поскольку каждая страница сайта имеет схожий стиль.

После успешного выполнения основных задач по созданию формы регистрации, необходимо установить связь между нашей базой данных и страницей. Мы стремимся обеспечить такое поведение, при котором при регистрации пользователь заносится в базу данных, а если он уже существует в этой базе данных, происходит перенаправление на главную страницу сайта; в противном случае должно отображаться сообщение «Неверный логин или пароль».

Листинг PHP составляющей db.php:

<?php

$servername = "localhost";

$username = "root";

$password = "root";

$dbname = "cars";

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

if(!$conn){

    die("- connect" . mysqli\_connect\_error());

}

Листинг PHP связи базы данных с формой регистрации:

<?php

session\_start();

require\_once \_\_DIR\_\_ . "/config/db.php";

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

  if (isset($\_POST['register'])) {

      $login = $\_POST['login'];

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

      $email = $\_POST['email'];

      $checkUserQuery = "SELECT \* FROM users WHERE login='$login' OR email='$email'";

      $checkUserResult = $conn->query($checkUserQuery);

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

          echo '<script>alert("Такой пользователь уже существует")</script>';

      } else {

          $insertQuery = "INSERT INTO users (login, password, email) VALUES ('$login', '$password', '$email')";

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

            $\_SESSION['authorized'] = true;

              echo '<script>alert("Регистрация прошла успешно");</script>';

          } else {

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

          }

      }

  }

  if (isset($\_POST['loggin'])) {

      $login = $\_POST['login'];

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

      $sql = "SELECT \* FROM users WHERE login = '$login' AND password = '$password'";

      $result = $conn->query($sql);

    if ($result->num\_rows > 0) {

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

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

        $\_SESSION['email'] = $email;

        session\_start();

        $\_SESSION['authorized'] = true;

        header('Location: person.php');

        exit;

    } else {

        $errorMessage = "Неверный логин или пароль";

    }

}

}

$conn->close();

?>

Последним действием на данной странице необходимо сделать вывод слов «Неверный логин или пароль».

Листинг PHP ошибки «Неверный логин или пароль»:

<?php

  if (isset($errorMessage)) {

      echo '<script>alert("Неверный логин или пароль");</script>';

  }

?>

<?php require\_once \_\_DIR\_\_ . "/incs/footer.php"; ?>

</body>

</html>

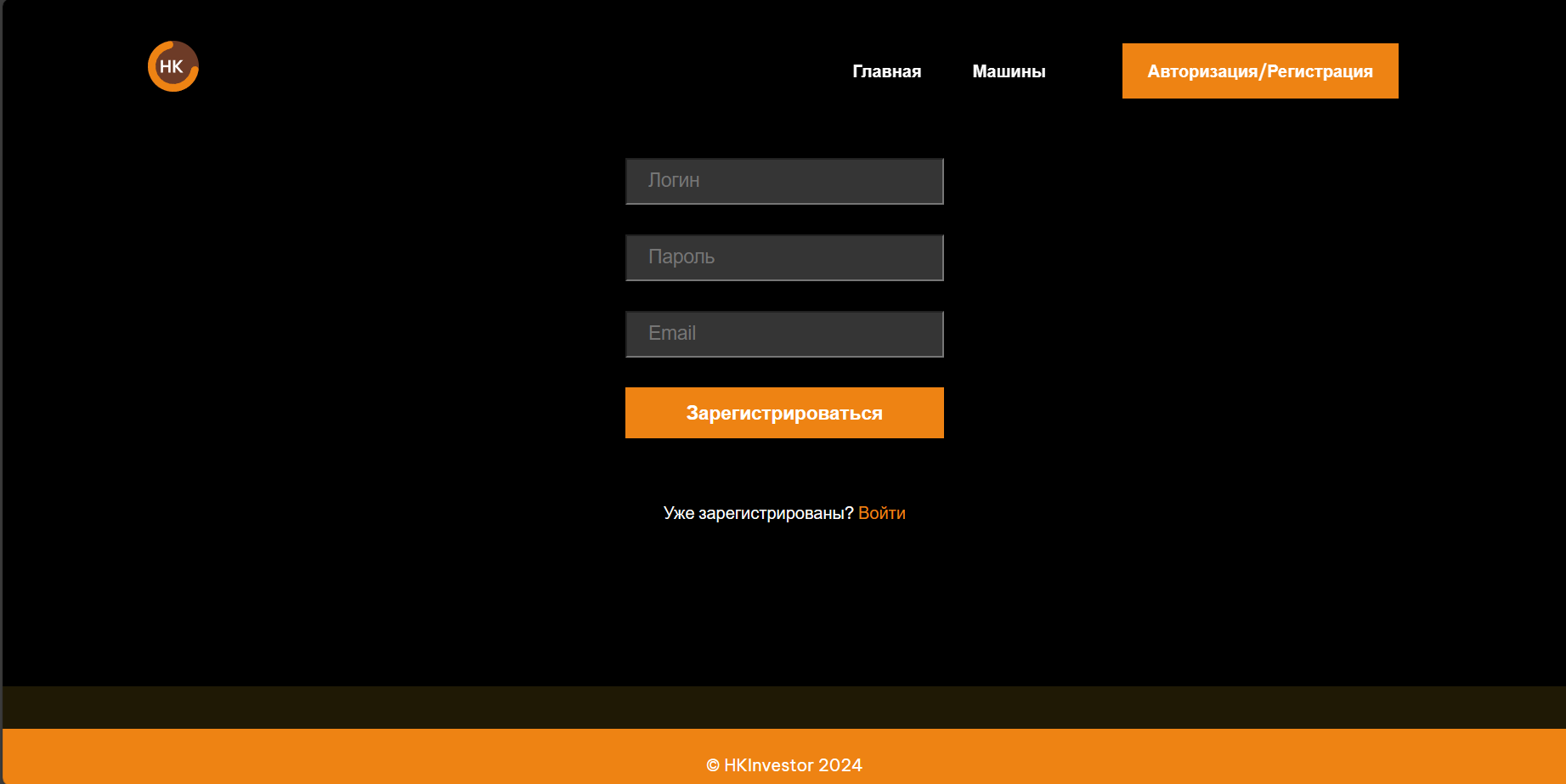


Рисунок 2.2.1 – Отображение формы регистрации в браузере

Мы успешно справились с первоочередной задачей. Теперь необходимо создать главную страницу и личный кабинет для нашего сайта.

## **2.3 Главная страница и личный кабинет**

Мы создадим два файла: index.php (главная страница) и person.php (личный кабинет). Поскольку на нашем сайте должна быть предусмотрена панель администратора, в которой администратор может редактировать пользователей, нам необходимо выбрать администратора. Мы выполним это действие простым способом, назначив пользователя с логином PALENOV в качестве администратора и используя сессии для создания условия, при котором, если логин пользователя не равен PALENOV, он не получит доступ к панели администратора.

Листинг HTML кода для главной страницы сайта:

<?php

 require\_once \_\_DIR\_\_ . "/incs/header.php";

 $title = 'Cars';

 ?>

<main>

        <section id="banner-section">

            <div class="container">

                <div class="banner">

                    <h2 class="banner\_\_header">Продажа автомобилей класса ЛЮКС</h2>

                    <p class="banner\_\_text">HKinvestor может предоставить в продажу самые эксклюзивные автомобили, такие как Matiz и Porsche.</p>

                    <div class="banner-links-wrap">

                        <a href="cars.php" class="btn-primary banner\_\_link">Машины</a>

                        <a href="https://www.youtube.com/watch?v=UMvKpDjlKnA" target="\_blank" class="btn-secondary banner\_\_btn">Смотреть видео</a>

                    </div>

                    <div class="banner-features">

                        <span class="banner-features\_\_text"><strong class="banner-features\_\_text\_strong">500+</strong>Люкс автомобилей</span>

                        <span class="banner-features\_\_text"><strong class="banner-features\_\_text\_strong">#1</strong>Автомобиль по лучшей цене</span>

                    </div>

                </div>

            </div>

            <div class="banner-img"></div>

        </section>

        <section id="rent-section">

            <div class="container">

                <h2 class="title-section">Приобретите автомобиль класса люкс</h2>

                <div class="rent-wrap">

                    <div class="rent">

                        <div class="rent-content">

                            <span class="rent\_\_name">Sportcars</span>

                            <a class="rent\_\_link" href="cars.php">Discover</a>

                        </div>

                    </div>

                    <div class="rent">

                        <div class="rent-content rent-content2">

                            <span class="rent\_\_name">Sportcars</span>

                            <a class="rent\_\_link" href="cars.php">Discover</a>

                        </div>

                    </div>

                    <div class="rent">

                        <div class="rent-content rent-content3">

                            <span class="rent\_\_name">Кроссоверы</span>

                            <a class="rent\_\_link" href="cars.php">Discover</a>

                        </div>

                    </div>

                    <div class="rent">

                        <div class="rent-content rent-content4">

                            <span class="rent\_\_name">Премиум Кроссоверы</span>

                            <a class="rent\_\_link" href="cars.php">Discover</a>

                        </div>

                    </div>

                </div>

            </div>

        </section>

        <section id="services-section">

            <div class="container">

                <h2 class="title-section services-section\_\_title">Другие наши услуги</h2>

                <div class="services-wrap">

                    <div class="services">

                        <img class="services\_\_img" src="img/main/services\_1.jpg" alt="Private Charter service">

                        <h3 class="services\_\_title">Трансфер</h3>

                        <p class="services\_\_text"> HKinvestor также предлагает вам незаметный и быстрый трансфер на вертолете или частном самолете до дома.</p>

                    </div>

                    <div class="services">

                        <img class="services\_\_img" src="img/main/services\_2.jpg" alt="Chauffer Driven Car Rentals service">

                        <h3 class="services\_\_title">Шофер до дома</h3>

                        <p class="services\_\_text">Наши шоферы - профессиональные водители, которые обеспечивают сдержанное, надежное, пунктуальное и профессиональное обслуживание.</p>

                    </div>

                    <div class="services">

                        <img class="services\_\_img" src="img/main/services\_3.jpg" alt="Wedding Car Rentals service">

                        <h3 class="services\_\_title">Жизнь в движении</h3>

                        <p class="services\_\_text">Если вы хотите кататься стильно и элегантно, выберите один автомобиль из нашей роскошной коллекции.</p>

                    </div>

                </div>

            </div>

        </section>

        <section id="location-section">

            <iframe src="https://yandex.ru/map-widget/v1/?um=constructor%3A3491417e4eebdadc9f3b098ef7f9b156948613c64825bc74e800d013988d96c8&amp;source=constructor" width="100%" height="400" frameborder="0"></iframe>

        </section>

<?php require\_once \_\_DIR\_\_ . "/incs/footer.php"; ?>

    </main>

</body>

</html>

Теперь мы рассмотрим шаги, необходимые для установления связи между базой данных и этой страницей.

Листинг PHP кода для соединения с базой данных и аутентификации пользователя:

require\_once \_\_DIR\_\_ . "/config/db.php";

if ($conn->connect\_error) {

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

}

session\_start();

if (!$\_SESSION['authorized']) {

    header('Location: index.php');

    exit;

}

$sql = "SELECT id, login, password, email FROM users";

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

if ($\_SESSION['username']!="PALENOV") {

  header('Location: index.php');

  exit; }

echo "<table>";

echo "<tr><th>ID</th><th>Login</th><th>Password</th><th>Email</th></tr>";

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

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

    echo "<tr><td>" . $row["id"]. "</td><td>" . $row["login"]. "</td><td>" . $row["password"]. "</td><td>" . $row["email"]. "</td></tr>";

  }

} else {

  echo "0 results";

}

echo "</table>";

echo "

<style>

table {

    box-sizing: border-box;

  border-collapse: collapse;

  width: 100%;

}

th, td {

  border: 1px solid black;

  padding: 8px;

  text-align: center;

}

th {

  background-color: #f2f2f2;

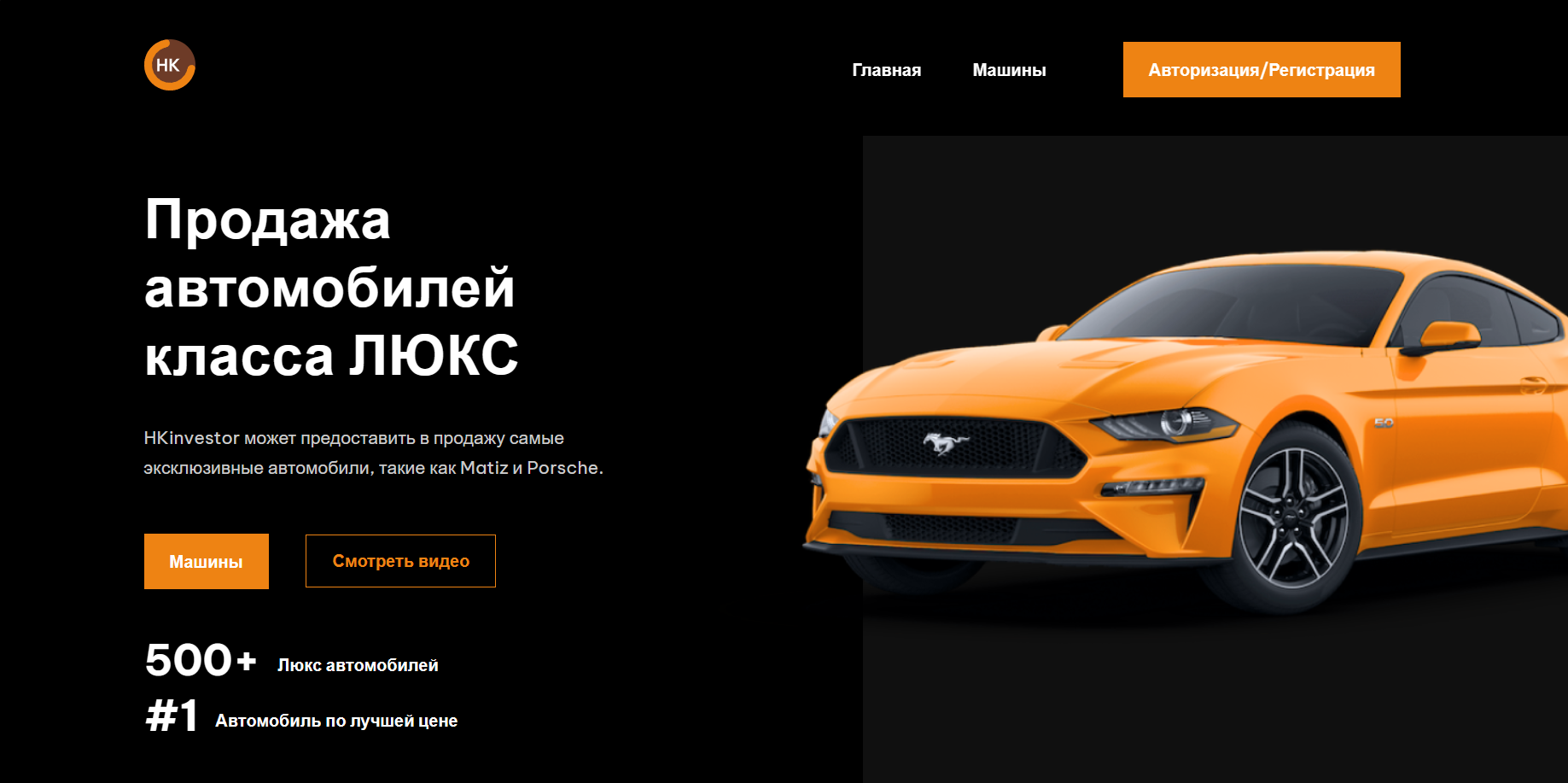
}

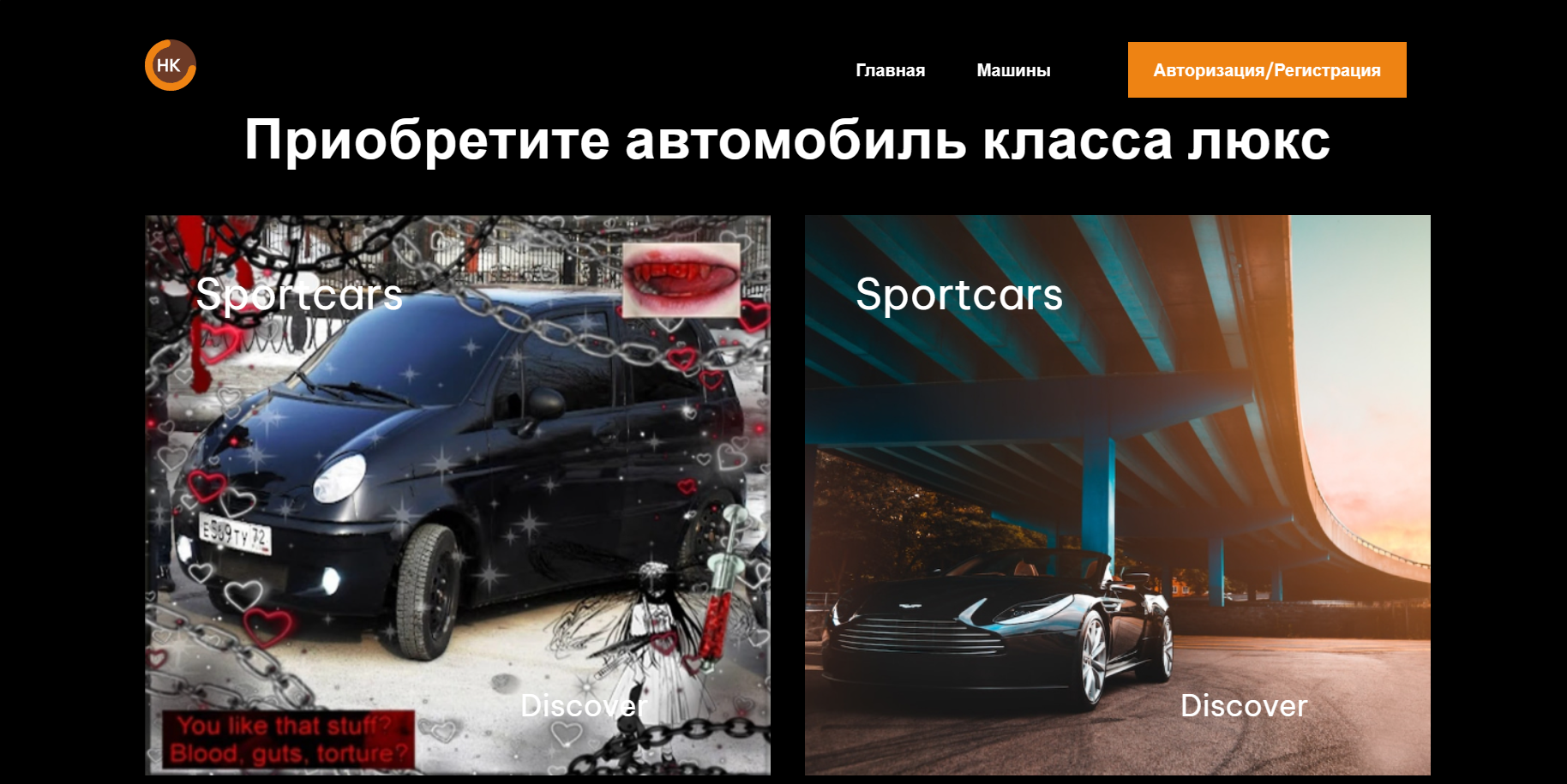
</style>

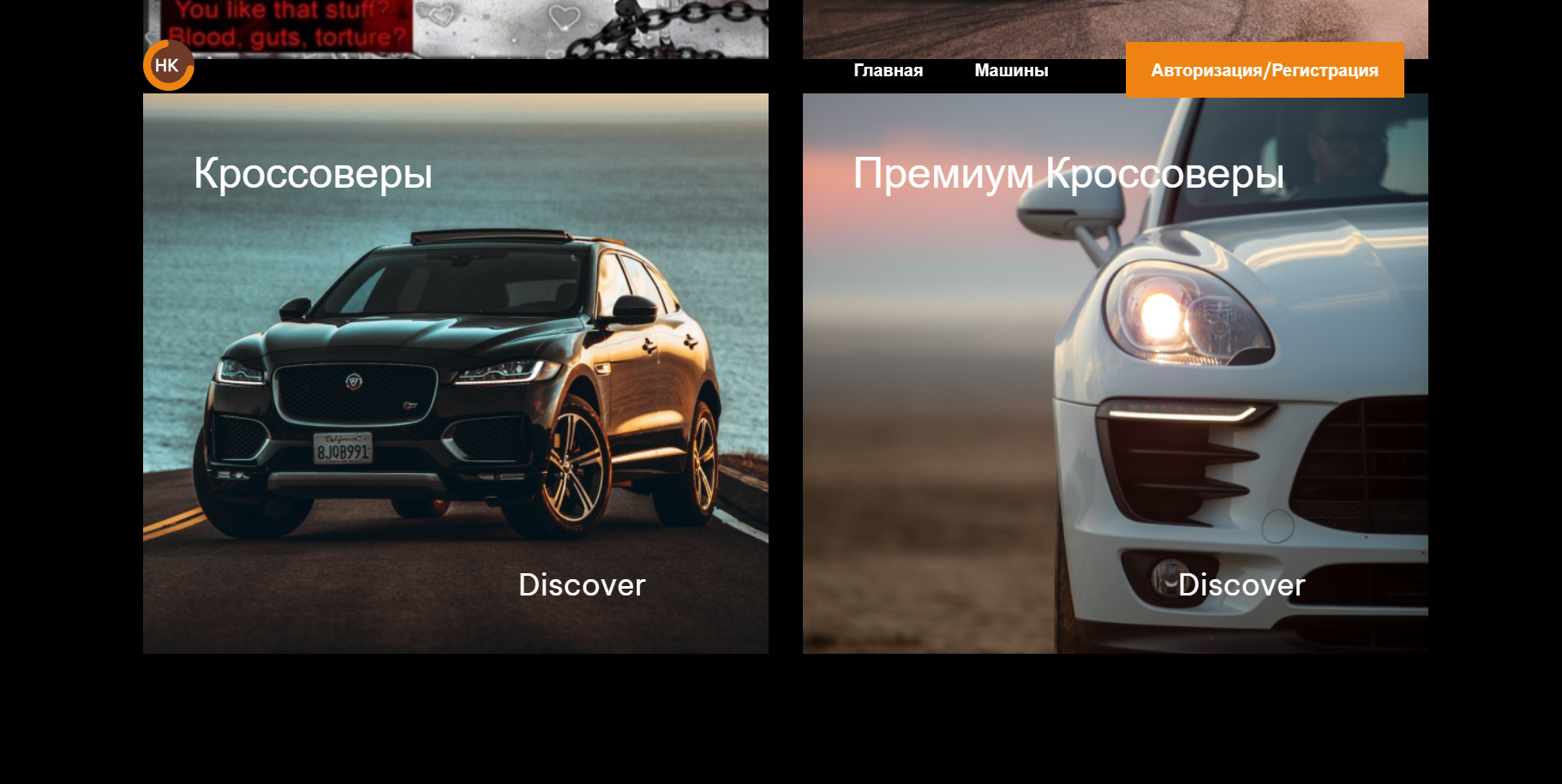
";

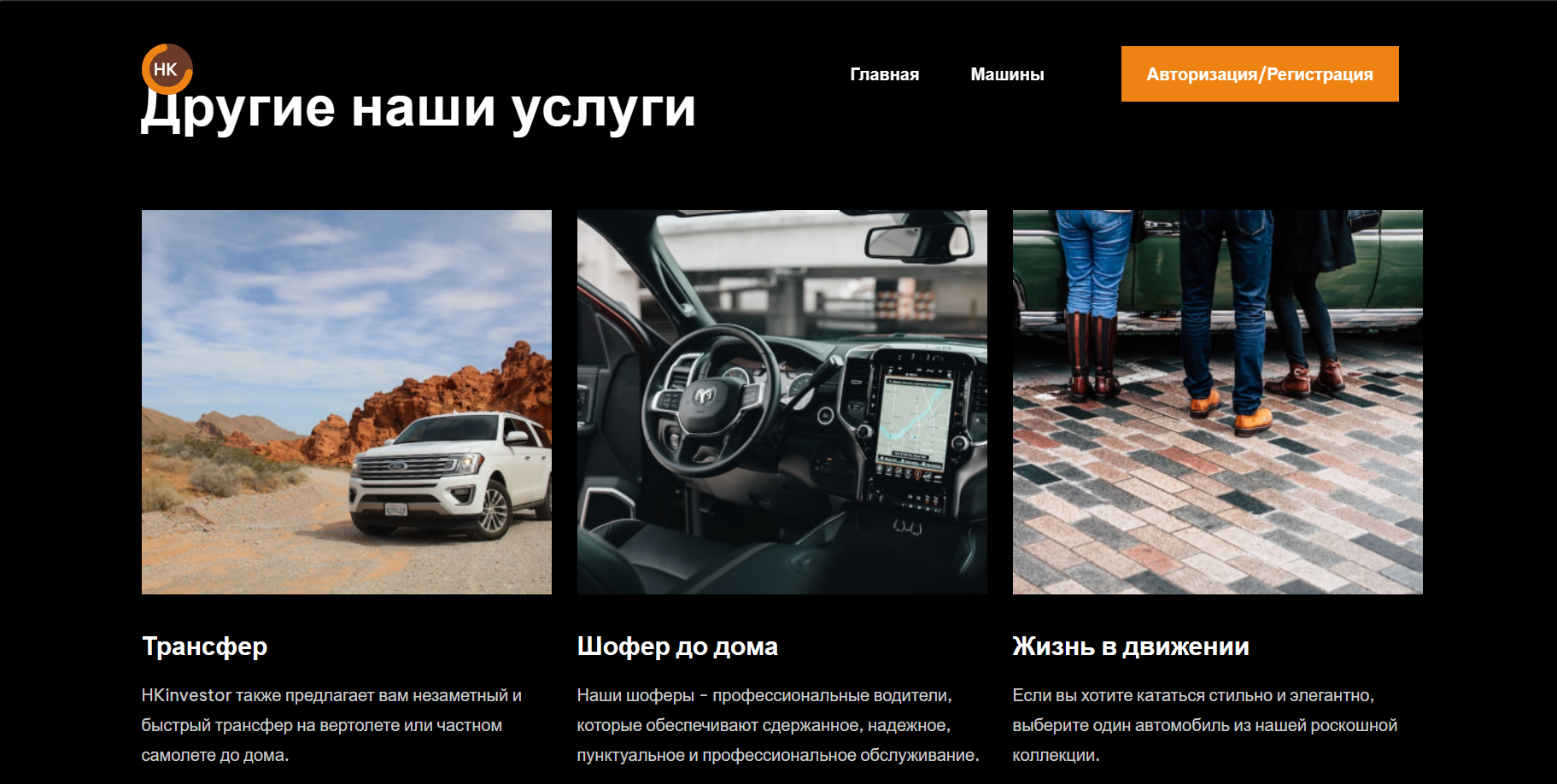
?>

Данный код включает в себя установку соединения с базой данных, проверку сеанса и аутентификации пользователя. Затем происходит отображение главной страницы приложения в веб-браузере.









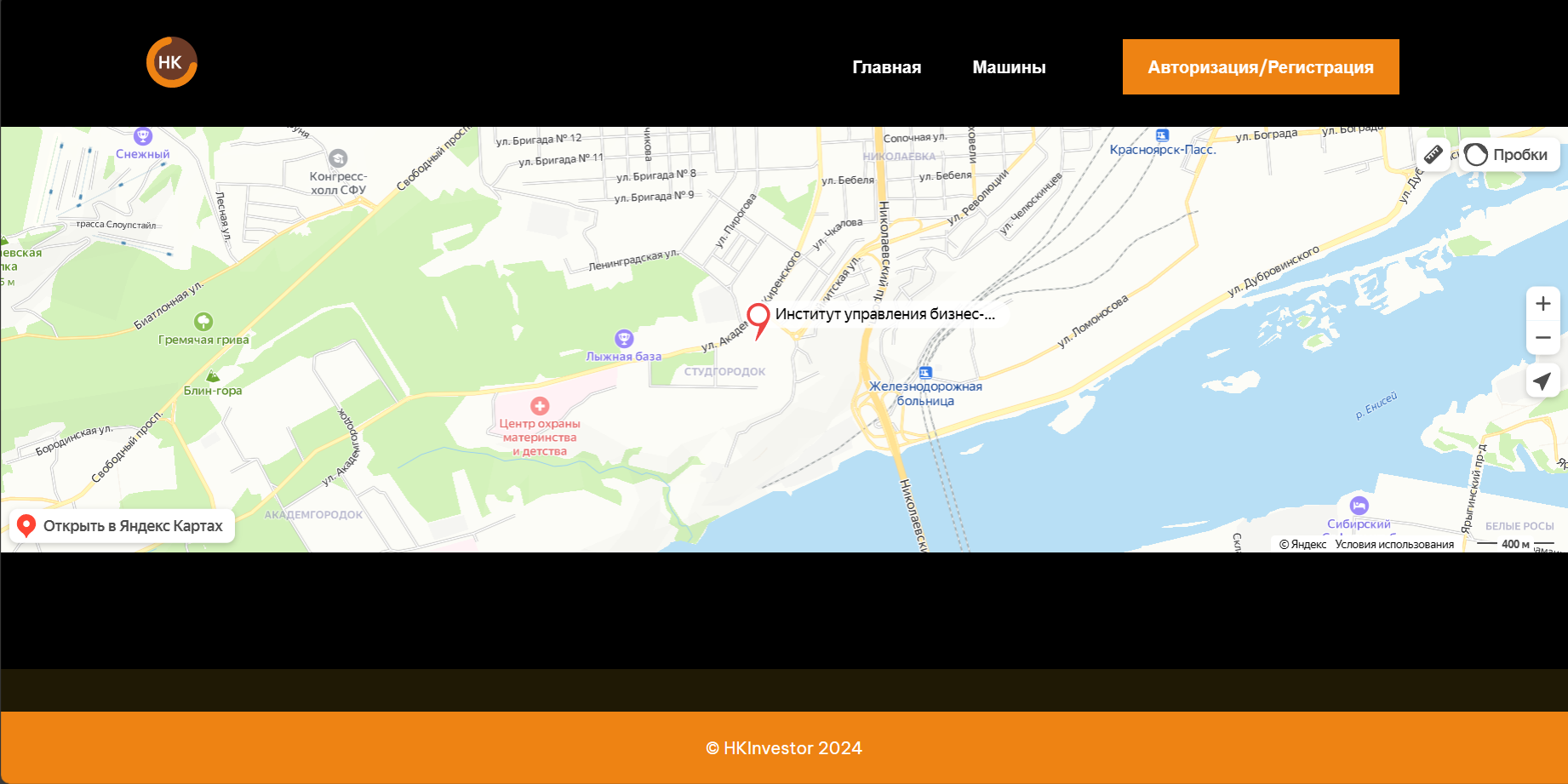


Рисунок 2.3.1 – Отображение страницы index.php в браузере

Продолжим процесс создания личного кабинета. Согласно требованиям проекта, необходимо обеспечить пользователям возможность редактировать собственные данные. Поэтому мы реализуем отображение информации о пользователе в личном кабинете и добавим функцию изменения пароля. Изменение логина и адреса электронной почты считается непрактичным.

Листинг HTML кода создания страницы личного кабинета:

<?php require\_once \_\_DIR\_\_ . "/incs/authheader.php"?>

<section class="person">

<div class="person\_\_title">

Личный кабинет

</div>

<div class="person\_\_info">

<div class="person\_\_info-info">

<div class="person\_\_info-info-title">

Информация о пользователе

</div>

<div class="person\_\_info-info-login">

Логин: <span> <?php echo "<td>" . $row["login"] . "</td>"; ?> </span>

</div>

<div class="person\_\_info-info-email">

E-mail: <span> <?php echo "<td>" . $row["email"] . "</td>"; ?> </span>

</div>

<div class="person\_\_info-info-pass">

Пароль: <span> <?php echo "<td>" . $row["password"] . "</td>"; ?> </span>

</div>

</div>

<div class="person\_\_repass">

<div class="person\_\_repass-title">

Изменить пароль

</div>

<div class="person\_\_repass-repass">

<form class="person\_\_repass-repass-form" action="<?php echo htmlspecialchars($\_SERVER["PHP\_SELF"]); ?>" method="POST">

<input class="repass-input" type="password" id="new\_password" name="new\_password" placeholder="Новый пароль">

<input class="repass-btn" type="submit" value="Изменить пароль">

</form>

</div>

</div>

</div>

<div class="person\_\_logOut">

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

<input type="submit" class="logout" name="logout" value="Выйти из аккаунта">

</form>

</div>

</section>

<?php require\_once \_\_DIR\_\_ . "/incs/footer.php"?>

</body>

</html>

Так как в личном кабинете имеется возможность выхода из аккаунта, необходимо внести дополнительные инструкции для удаления сессии в базе данных, чтобы информация о пользователях была корректной.

Листинг PHP person.php:

<?php

require\_once \_\_DIR\_\_ . "/config/db.php";

if ($conn->connect\_error) {

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

}

$errorMessage = "";

session\_start();

$sql = "SELECT \* FROM users WHERE login='{$\_SESSION['username']}'";

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

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

    $row = $result->fetch\_assoc();

    $login = $row["login"];

    $password = $row["password"];

    $email = $row["email"];

    if (!$\_SESSION['authorized']) {

        header('Location: index.php');

        exit;

    }

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

        if (isset($\_POST["logout"])) {

            session\_unset();

            session\_destroy();

            header("Location: index.php");

            exit;

        }

        $newPassword = $conn->real\_escape\_string($\_POST["new\_password"]);

        $userid = $row['id'];

        $sql = "UPDATE users SET password = '$newPassword' WHERE id = '$userid'";

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

            $successMessage = "Пароль успешно обновлен";

        } else {

            $errorMessage = "Ошибка обновления пароля: " . $conn->error;

        }

    }

} else {

    $errorMessage = "Ошибка получения данных пользователя";

}

$conn->close();

?>

Дальнейшая часть кода состоит из стилей, которые аналогичны стилям на странице index.php, поэтому можно перейти к демонстрации внешнего вида этой страницы в браузере. На рисунке 2.3.2 показано отображение страницы person.php в браузере.

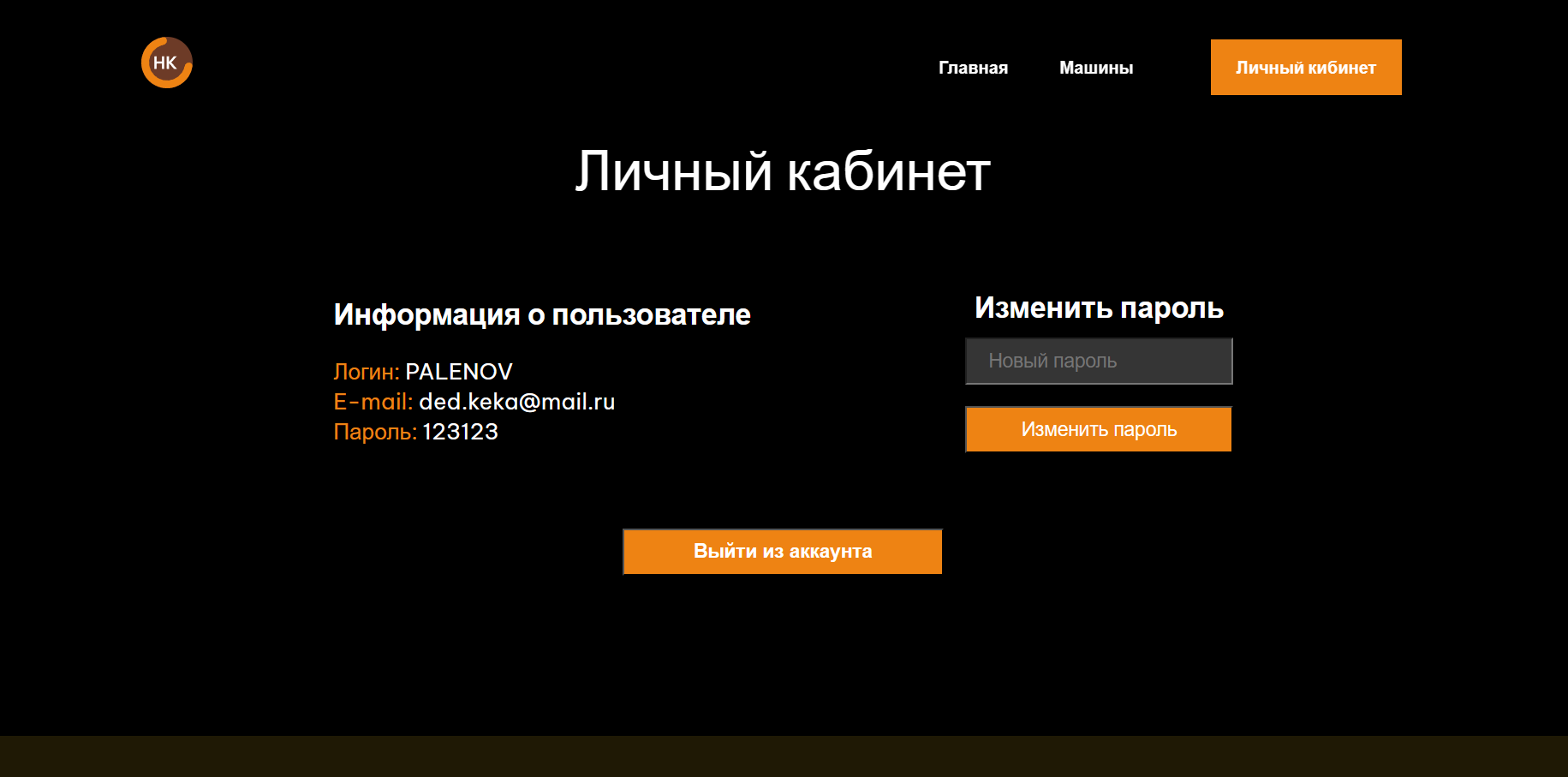


Рисунок 2.3.2 – Отображение страницы person.php в браузере

## **2.4 Панель администратора**

Смысл панели администратора заключается в том, что администратор может редактировать пользователей, нажимая на кнопки. В нашем приложении на панель администратора может перейти только пользователь PALENOV, с любой страницы.

Перейдем к созданию панели. Ниже приведен код для создания страницы с панелью администратора.

Листинг HTML,CSS,PHP кода для создания панели администратора:

<!DOCTYPE html>

<html>

<head>

    <title>Админ панель</title>

    <style>

body {

      font-family: Arial, sans-serif;

      background-color: #1a1a1a;

      color: #ffffff;

    }

    .form {

      margin-top: 20px;

      text-align: center;

    }

    .input1 {

      padding: 10px;

      margin: 5px;

      border: 2px solid #ff6600;

      border-radius: 5px;

    }

    .input1:focus {

      outline: none;

      border: 2px solid #ffcc99;

    }

    .input1::placeholder {

      color: #ff6600;

    }

    .input1[type="submit"] {

      background-color: #ff6600;

      color: #ffffff;

      padding: 10px 20px;

      border: none;

      border-radius: 5px;

      cursor: pointer;

    }

    .input1[type="submit"]:hover {

      background-color: #ff8533;

    }

    .logout {

      background-color: #ff6600;

      color: #ffffff;

      padding: 10px 20px;

      border: none;

      border-radius: 5px;

      cursor: pointer;

    }

    .logout:hover {

      background-color: #ff8533;

    }

    th {

      color: #ff6600;

    }

    </style>

</head>

<body>

<form class='form'action='delete\_user.php' method='post'>

  <input type='text' class='input1' name='id' placeholder='ID пользователя'>

  <input type='submit' class='input1' value='Удалить'>

</form>

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

  <input type='text' class='input1' name='id' placeholder='ID пользователя'>

  <input type='text' class='input1' name='login' placeholder='Новый логин'>

  <input type='text' class='input1' name='password' placeholder='Новый пароль'>

  <input type='text' class='input1' name='email' placeholder='Новый email'>

  <input type='submit' class='input1' value='Редактировать'>

</form>

<form amethod="post" action="authindex.php">

    <input type="submit" class="logout" name="logout" value="На главную">

</form>

</body>

</html>

<?php

require\_once \_\_DIR\_\_ . "/config/db.php";

if ($conn->connect\_error) {

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

}

session\_start();

if (!$\_SESSION['authorized']) {

    header('Location: index.php');

    exit;

}

$sql = "SELECT id, login, password, email FROM users";

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

if ($\_SESSION['username']!="PALENOV") {

  header('Location: index.php');

  exit; }

echo "<table>";

echo "<tr><th>ID</th><th>Login</th><th>Password</th><th>Email</th></tr>";

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

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

    echo "<tr><td>" . $row["id"]. "</td><td>" . $row["login"]. "</td><td>" . $row["password"]. "</td><td>" . $row["email"]. "</td></tr>";

  }

} else {

  echo "0 results";

}

echo "</table>";

echo "

<style>

table {

    box-sizing: border-box;

  border-collapse: collapse;

  width: 100%;

}

th, td {

  border: 1px solid black;

  padding: 8px;

  text-align: center;

}

th {

  background-color: #f2f2f2;

}

</style>

";

?>

<?

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

    if (isset($\_POST["logout"])) {

        session\_unset();

        session\_destroy();

        header("Location: main.php");

        exit;

    }

}

$conn->close();

?>

Для того, чтобы эта панель функционировала должным образом, будет необходимо создать два дополнительных файла: edit\_user.php и delete\_user.php. Первый файл предназначен для редактирования данных пользователей, а второй - для их удаления.

Листинг PHP кода delete\_user.php:

<?php

require\_once \_\_DIR\_\_ . "/config/db.php";

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

if($id == 1) {

  echo "Ты че, не трогай его";

} else {

  $sql = "DELETE FROM users WHERE id=$id";

  if ($conn->query($sql) === TRUE) {

    header('Location:admin.php');

  } else {

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

  }

}

$conn->close();

?>

Листинг PHP код файла edit\_user.php:

<?php

require\_once \_\_DIR\_\_ . "/config/db.php";

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

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

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

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

$sql = "UPDATE users SET login='$login', password='$password', email='$email' WHERE id=$id";

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

  header('Location:admin.php');

} else {

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

}

$conn->close();

?>

Разработав стили, можно считать, что панель администратора создана и готова к использованию. На рисунке 2.4.1 можно увидеть, как выглядит панель администратора в браузере.

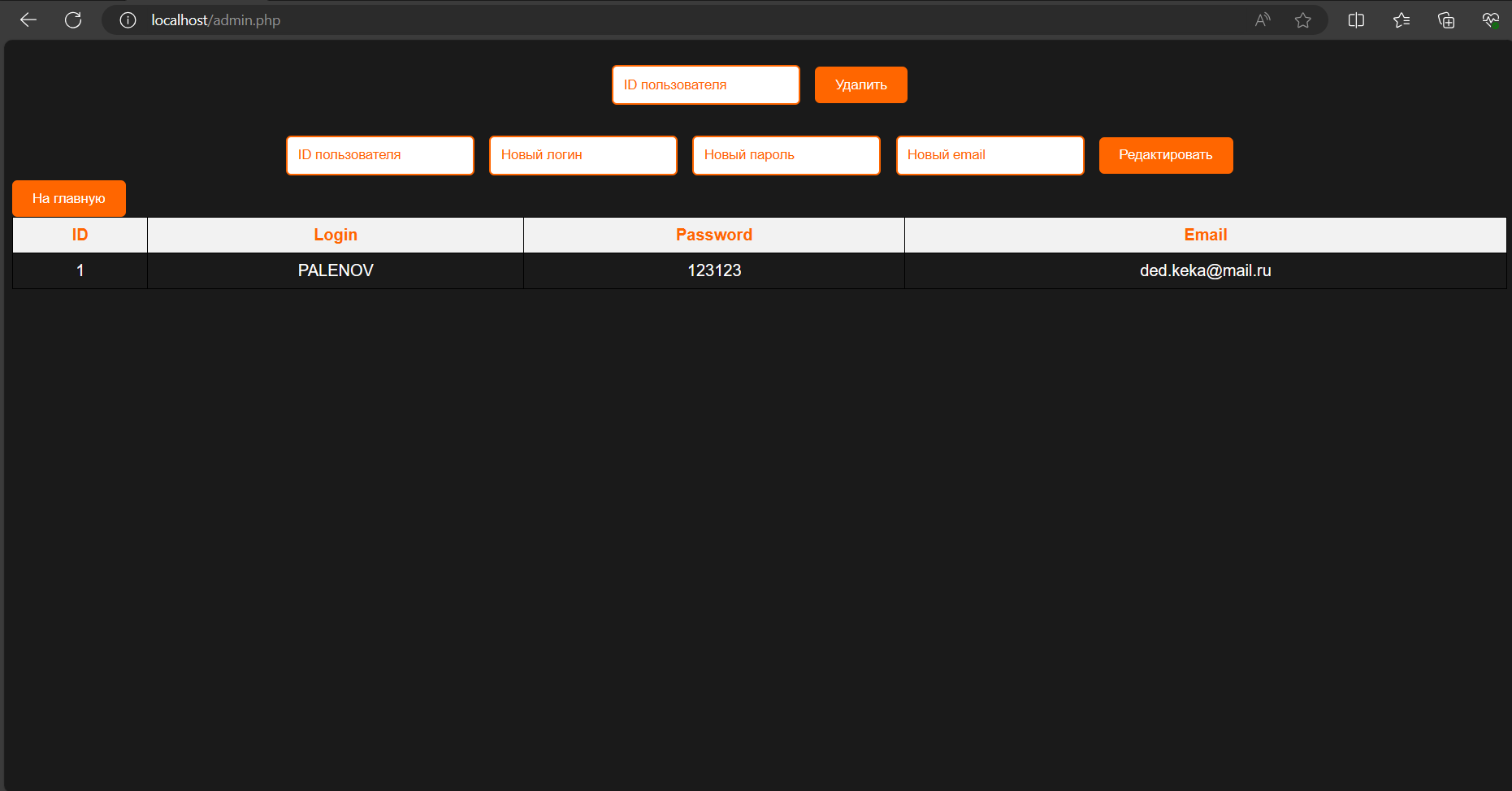


Рисунок 2.4.1 – Панель администратора в браузере

## **2.5 Страницы с основным контентом**

Сейчас мы разработали форму регистрации, основную страницу, страницу личного кабинета и панель администратора. Теперь следует создать страницы, которые будут содержать основной контент нашего приложения. Предлагается создать 4 файла: cars.php, index.php, authindex.php, authcars.php. Первые два файла аналогичны третьему и четвертому, но они будут представлять обзор услуг сайта для неавторизованных пользователей. Доступ к автомобилям будет возможен только на страницах authindex.php и authcars.php. Остальные страницы будут иметь аналогичный код, за исключением информации о машинах.

Листинг страницы authcars.php:

<?php

require\_once \_\_DIR\_\_ . "/config/db.php";

if ($conn->connect\_error) {

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

}

$errorMessage = "";

session\_start();

$sql = "SELECT \* FROM users WHERE login='{$\_SESSION['username']}'";

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

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

    $row = $result->fetch\_assoc();

    $login = $row["login"];

    $password = $row["password"];

    $email = $row["email"];

} else {

    $errorMessage = "Ошибка получения данных пользователя";

}

    if (!$\_SESSION['authorized']) {

        header('Location: index.php');

        exit;

    }

    $conn->close();

?>

<?php require\_once \_\_DIR\_\_ . "/incs/authheader.php" ?>

<section class="cars\_\_title">

        <div class="container">

            Наши машины(и один матиз(на плове))

        </div>

    </section>

    <div class="cars\_\_cars">

        <div class="container">

            <div class="cars\_\_cards">

            <div class="cars\_\_card">

                    <div class="cars\_\_card-img">

                        <img src="img/main/rent.jpg" alt="">

                    </div>

                    <div class="cars\_\_card-title">

                        Matiz на плове

                    </div>

                    <div class="cars\_\_card-price">

                        $250000

                    </div>

                    <div class="cars\_\_card-button">

                        <a href="/pdf/matiz.pdf" target="\_blank">Смотреть</a>

                    </div>

                </div>

                <div class="cars\_\_card">

                    <div class="cars\_\_card-img">

                    <img src="img/main/rent\_2.jpg" alt="">

                    </div>

                    <div class="cars\_\_card-title">

                        Машина похожая на Aston Martin

                    </div>

                    <div class="cars\_\_card-price">

                        $50 000

                    </div>

                    <div class="cars\_\_card-button">

                        <a href="/pdf/ast.pdf" target="\_blank">Смотреть</a>

                    </div>

                </div>

                <div class="cars\_\_card">

                    <div class="cars\_\_card-img">

                    <img src="img/main/rent\_3.jpg" alt="">

                    </div>

                    <div class="cars\_\_card-title">

                        Jaguar F-Pace

                    </div>

                    <div class="cars\_\_card-price">

                        $32 000

                    </div>

                    <div class="cars\_\_card-button">

                        <a href="/pdf/jaga1.pdf" target="\_blank">Смотреть</a>

                    </div>

                </div>

                <div class="cars\_\_card">

                    <div class="cars\_\_card-img">

                    <img src="img/main/rent\_4.jpg" alt="">

                    </div>

                    <div class="cars\_\_card-title">

                        Porsche Cayenne

                    </div>

                    <div class="cars\_\_card-price">

                        $40 000

                    </div>

                    <div class="cars\_\_card-button">

                        <a href="/pdf/por.pdf" target="\_blank">Смотреть</a>

                    </div>

                </div>

            </div>

        </div>

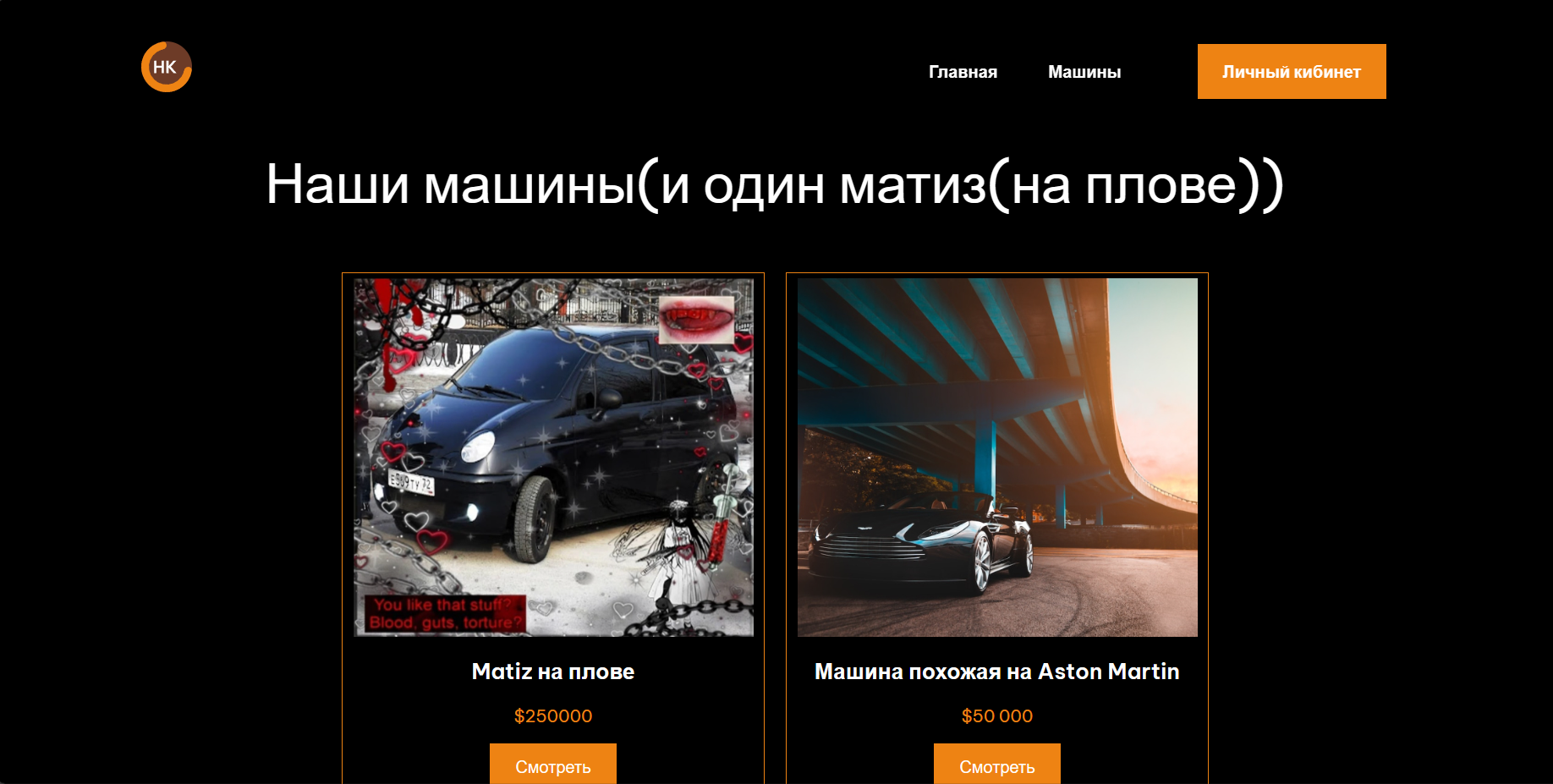
    </div>

<?php require\_once \_\_DIR\_\_ . "/incs/footer.php" ?>

</body>

</html>

Теперь мы можем просмотреть, как выглядят эти страницы в браузере. На рисунке 2.5.1 показан внешний вид страницы authcars.php.



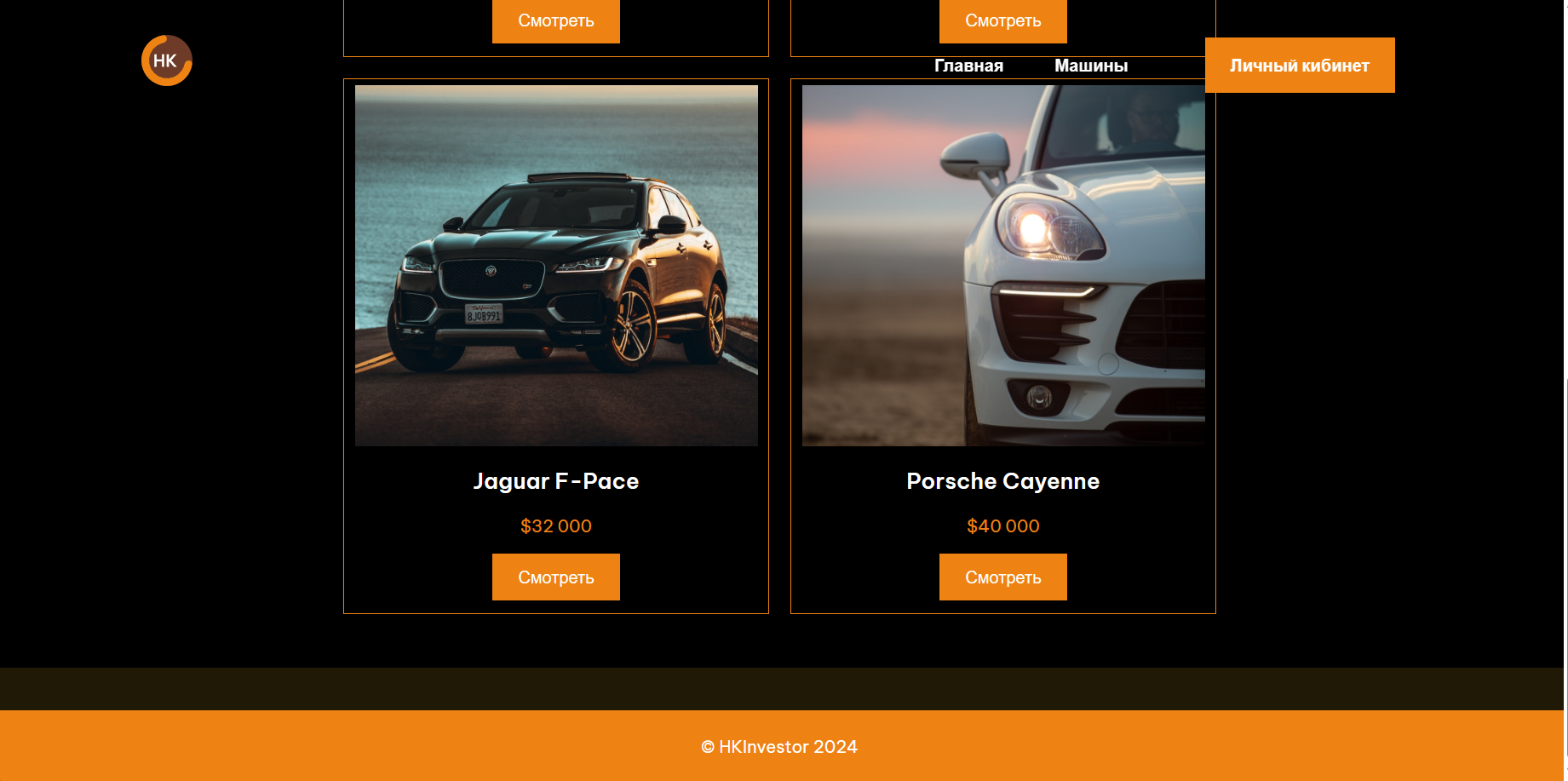


Рисунок 2.5.1 – Страница authcars.php в браузере

Просмотр страниц index.php и cars.php бессмысленен, так как их различие с authindex.php и authcars.php заключается в невозможности просмотра авторизованным пользователем. Доступ к этим страницам можно получить через форму регистрации.

# 

# ЗАКЛЮЧЕНИЕ

В рамках проекта было создано веб-приложение из семи страниц, посвященных книгам. Это включает страницу для регистрации logReg.php, главную страницу index.php, личный кабинет person.php, главную страницу для авторизованных пользователей authindex.php, страницу с машинами для авторизованных пользователей authcars.php, страницу с машинами для неавторизованных пользователей cars.php и страницу панели администратора admin.php.

Каждая страница приложения взаимодействует с базой данных "cars" и связанной с ней таблицей "users". На всех страницах происходит проверка аутентификации пользователя.

Администратор имеет возможность редактировать данные пользователей и удалять их через панель администратора. Доступ к этой панели предоставлен только пользователю с логином "PALENOV".

Это веб-приложение было разработано с использованием HTML, CSS, JavaScript, PHP и MySQL. Цель проекта достигнута.

# 

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Руководство по HTML5 и CSS3 // metanit URL: <https://metanit.com/web/html5/> (дата обращения: 19.12.2023).
2. Руководство по JS // metanit URL: <https://metanit.com/web/javascript/> (дата обращения: 21.12.2023).
3. Руководство по PHP // metanit URL: <https://metanit.com/php/tutorial/> (дата обращения: 20.12.2023).
4. Документация MySQL. // metanit URL: <https://metanit.com/sql/mysql/> (дата обращения: 21.12.2023).
5. PHP,MySQL,JavaScript,CSS и HTML5 / Робин Никсон 6-е издание, 2023. – 832 с. (дата обращения: 21.12.2023).