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 «Портфолио»

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

подпись, дата

Студент УБ22-09Б, 432216061 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Кене Л.Н.

подпись, дата

Красноярск 2023

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc156382038)

[1 Теоретическая часть 4](#_Toc156382039)

[1.1 HTML 4](#_Toc156382040)

[1.2 CSS 5](#_Toc156382041)

[1.3 JavaScript 7](#_Toc156382042)

[1.4 PHP 8](#_Toc156382043)

[1.5 MySQL 9](#_Toc156382044)

[2 Практическая часть 12](#_Toc156382045)

[2.1 Требования к web-приложению и его характеристика 12](#_Toc156382046)

[2.2 Создание базы данных на MySQL в PHPMyAdmin 14](#_Toc156382047)

[2.3 Регистрация и авторизация пользователей, общие и схожие блоки кода 16](#_Toc156382048)

[2.4 Основные страницы – «Главная», «Портфолио», «Профиль» 29](#_Toc156382049)

[2.5 Дополнительные страницы 31](#_Toc156382050)

[2.6 «Админская панель» 32](#_Toc156382051)

[ЗАКЛЮЧЕНИЕ 35](#_Toc156382052)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 36](#_Toc156382053)

# ВВЕДЕНИЕ

В наши дни создание личного сайта-портфолио является жизненно важным. В эпоху цифровизации и интернета, когда все больше людей и организаций ищут информацию и услуги в Интернете, личные веб-сайты-портфолио становятся необходимым инструментом для представления своих навыков, работ и достижений.

Первым преимуществом создания личного сайта-портфолио является то, что он доступен. В отличие от обычных резюме в виде документов, он позволяет максимально просто и наглядно представить свои работы и проекты в любом формате и размере. Без учета количества слов или размера файла пользователь может увидеть всю информацию, которую хочет предоставить автор.

Персонализация также имеет решающее значение. Личный сайт-портфолио отличается от платформ для размещения резюме или социальных сетей тем, что позволяет создавать эксклюзивный дизайн, который отражает стиль и характер автора. Это позволяет выделиться среди других и запомниться работодателям и клиентам.

Личный сайт-портфолио также полезен для поиска работы и продвижения. Поисковые системы позволяют использовать ключевые слова и запросы для поиска клиентов и работодателей. Кроме того, создание личного сайта-портфолио позволяет автору активно использовать социальные сети и медиа-платформы для распространения контактов и своих работ. Это улучшает вашу репутацию в Интернете и расширяет ваши возможности получения работы и предложений.

Цель: разработать сайт, используя технологии HTML, CSS, Javascript, PHP, MySQL на тему «Портфолио».

Объект исследования: создание web-приложения на тему «Портфолио».

Предмет исследования: процесс создания web-приложения на тему «Портфолио».

# Теоретическая часть

## **HTML**

HTML, также известный как Hypertext Markup Language, – это код, который используется для организации и отображения веб-страниц и их контента. Например, контент может быть структурирован в виде маркированных списков, параграфов или с использованием изображений и таблиц данных.

HTML – это не язык программирования, – это язык разметки, который помогает вашему браузеру отображать веб-страницы. HTML состоит из множества элементов, которые вы можете использовать, чтобы включать или оборачивать отдельные части контента, чтобы заставить его отображаться или действовать определенным образом. Ограждающие теги могут изменять текст или изображение, делать слова курсивом, изменять размер шрифта и так далее [1].

С его помощью можно описать структуру и содержимое страницы, определить различные элементы и их взаимосвязь. HTML является основным языком для создания содержимого веб-сайтов и организации информации для отображения в браузере. Основными компонентами HTML являются теги, которые определяют начало и конец элемента на веб-странице. Различные теги описывают разные элементы страницы, отображая текст, изображения, ссылки, списки, таблицы и другие объекты. Разработчики используют комбинацию тегов для создания структуры и внешнего вида страницы. Например, для создания заголовка страницы используется тег "<h1>", а для параграфа – тег "<p>". Ссылки создаются с помощью тега "<a>", а изображения – с помощью тега "<img>" [2]. Теги могут иметь атрибуты, которые дополняют или изменяют их поведение и внешний вид. Например, атрибут "href" определяет адрес, на который будет вести ссылка. Кроме основных элементов, HTML также поддерживает встраиваемые скрипты и стили. Скрипты позволяют добавить динамическое поведение на страницу, в то время как стили используются для определения внешнего вида элементов, об этом в параграфах 1.2 и 1.3.

HTML также позволяет организовать страницу в различные секции с помощью элементов, таких как "<header>", "<nav>", "<main>", "<section>", "<aside>" и "<footer>". Это помогает разбить страницу на логические блоки и облегчает навигацию и понимание структуры страницы. Для создания сложной таблицы данных можно использовать элементы "<table>", "<tr>", "<td>" и другие. Элементы формы, такие как "<input>", "<textarea>" и "<select>", позволяют создавать интерактивный контент и собирать информацию от пользователей.

Можно рассмотреть на примере, из чего состоит элемент абзаца. Запись «<p>Текст<p>» содержит открывающий тег <p> и закрывающий тег </p> в угловых скобках, контент (Текст), – все это вместе составляющие элемента абзаца [1].

HTML является открытым стандартом, что означает, что любой может использовать и расширять его. Новые версии HTML, такие как HTML5, вводят новые возможности и элементы, улучшающие пользовательский интерфейс, производительность и доступность и делающие разработку более простой и эффективной. HTML является основой веб-разработки и неотъемлемой частью создания веб-страниц. Он позволяет веб-разработчикам создавать красивые, удобочитаемые и уникальные страницы, а также обеспечивает доступность и интерактивность для пользователей. Знание HTML является необходимым для всех, кто интересуется созданием и разработкой веб-сайтов.

## **CSS**

Набор параметров форматирования, который применяется к элементам документа, чтобы изменить их внешний вид, называется стилем или CSS (Cascading Style Sheets, каскадные таблицы стилей). Современные издательские системы и текстовые редакторы давно поддерживают возможность работы со стилями, позволяя одним нажатием кнопки придать тексту заранее заданный вид. Создатели сайтов могут теперь использовать это, храня цвет, размеры текста и другие параметры в одном месте и легко «прикручивая» их к любому тегу. Стили также предлагают гораздо больше возможностей для форматирования, чем обычный HTML, что является дополнительным преимуществом [3].

Выделяют следующие преимущества использования каскадных таблиц стилей:

* Большое количество возможностей для веб-дизайнера, поскольку CSS – это стандарт W3C, разработанный специально для дизайна сайтов. Не все идеи можно реализовать с помощью HTML.
* Стили и их описания могут быть включены в отдельный файл. Таким образом, работа параллельно с дизайном и контентом сайта исключает путаницу.
* Из предыдущего пункта вытекают преимущества хранения стилей в одном файле (обычно style.css), а путь к нему указывается в начале документа. Таким образом, для изменения вида страницы достаточно использовать один файл.
* Поскольку таблицы стилей сохраняются в кэш-памяти, загрузка сайта происходит быстрее. Кроме того, можно значительно сократить количество кода, написанного с помощью CSS.
* Блочная ориентация облегчает разработку единого стиля для каждой страницы веб-сайта, поскольку все страницы объединены в одну таблицу. Кроме того, это позволяет более точно управлять оформлением, поскольку изменение одной строки изменит все представление на каждой странице.
* Таблицы стилей позволяют изменять представление страниц для различных устройств, что очень важно в настоящее время [4].

Чтобы задать красный цвет тексту во всех абзацах, можно написать следующий код: P {color: red;}.

Анатомия набора правил CSS, как можно видеть, проста:

* Существует селектор – элемент, к которому применяется изменение. В качестве селектора могут быть теги html, а также придуманные классы (.class), универсальный селектор (\*), идентификатор (#id), атрибут ([attr\*=value]).
* Объявление единственного правила, которое изменяет свойство элемента.
* Непосредственно сами свойства CSS, способы, которыми вы можете стилизовать определённый HTML-элемент.
* Справа от свойства, после двоеточия, у нас есть значение свойства, которое выбирает одно из множества возможных признаков для данного свойства.
* Каждый набор правил упакован в фигурных скобках.
* После объявления свойства CSS ставится двоеточие перед значением свойства. В каждом наборе правил необходимо использовать точку с запятой, чтобы отличать каждое объявление от следующего [5].

Важно отметить, что у CSS есть препроцессор SCSS или SASS. Это метаязык на основе CSS, предназначенный для увеличения уровня абстракции CSS-кода и упрощения файлов каскадных таблиц стилей, включён в состав языка разметки Haml.

## **JavaScript**

JavaScript (JS) – это высокоуровневый, интерпретируемый язык программирования, который применяется для создания динамических веб-страниц и веб-приложений. JS позволяет добавить интерактивность и функциональность на сайты, делая их более привлекательными для пользователей [6].

JavaScript разработан Brendan Eich в 1995 году для использования в Netscape Navigator. Сегодня он является одним из трех основных языков веб-разработки, вместе с HTML и CSS. JS может работать как на стороне клиента (front-end), так и на стороне сервера (back-end) с использованием платформы Node.js.

JavaScript включает в себя множество функций и возможностей. Он поддерживает объектно-ориентированное программирование (ООП), что позволяет разработчикам организовывать код вокруг объектов и их взаимодействия. Также JS поддерживает функциональное программирование, которое облегчает работу с функциями как с объектами первого класса.

JS имеет широкий набор встроенных функций и методов, которые позволяют работать с строками, числами, массивами, объектами и другими типами данных. Он также предоставляет возможности для манипуляции DOM (Document Object Model) - структура веб-страницы, что позволяет изменять содержимое и стиль элементов в реальном времени.

JS имеет поддержку асинхронного программирования с помощью промисов и асинхронных функций. Это особенно полезно при выполнении операций, которые могут занимать много времени, таких как загрузка данных с сервера или выполнение сложных вычислений. JS также позволяет создавать анимации, добавлять эффекты перехода и взаимодействия с элементами страницы через CSS-анимации и CSS-трансформации. Для работы с JS разработчикам доступны различные инструменты и фреймворки. Некоторые из самых популярных фреймворков включают React.js, Angular.js и Vue.js, которые предоставляют разнообразные инструменты и библиотеки для разработки сложных веб-приложений и компонентов пользовательского интерфейса [7].

## **PHP**

PHP, аббревиатура от «Hypertext Preprocessor», является языком программирования, предназначенным для веб-разработки. PHP специально сконструирован для веб-разработок и его код может внедряться непосредственно в HTML. Многие люди используют его для создания динамичных веб-страниц и приложений. PHP имеет множество возможностей для разработчиков и является одним из наиболее популярных языков программирования на сервере.

PHP является бесплатным и открытым языком программирования. В связи с тем, что он работает на сервере, весь код PHP выполняется на сервере перед тем, как данные отправляются на компьютер пользователя. Способность взаимодействовать с базами данных, обрабатывать формы и создавать динамический контент на веб-страницах являются основными преимуществами PHP.

PHP предлагает разработчикам множество встроенных функций, которые позволяют им создавать мощные веб-приложения. Он поддерживает объектно-ориентированное программирование (OOP), которое позволяет организовывать код вокруг объектов и их взаимодействия. Кроме того, PHP имеет много встроенных функций, которые позволяют работать с файлами, строками, массивами и другими типами данных.

Возможность работы с базами данных является одной из основных функций PHP. PHP работает вместе с несколькими системами управления базами данных (СУБД), такими как MySQL, PostgreSQL и SQLite. Это облегчает создание, извлечение, изменение и удаление данных из баз данных и использование их на веб-страницах.

При написании страницы сайта на html можно вставить что-либо на языке PHP просто написав такие «скобки»: «<?php /Блок PHP кода/ ?>» [8].

## **MySQL**

MySQL – это реляционная база данных (СУБД), предназначенная для обработки, организации и хранения больших количеств данных. Она предлагает широкий набор функций для удобной работы с данными и является одной из самых популярных и широко используемых СУБД в мире.

Ядро MySQL, MySQL Workbench, сервер MySQL и протокол командного интерфейса являются основными компонентами MySQL.

Ядро MySQL является важной частью СУБД и выполняет важные функции, такие как управление данными, обработка транзакций и поддержка языка запросов. С его помощью можно создавать, изменять и удалять базы данных, таблицы и индексы, а также выполнять сложные запросы на извлечение данных.

Программное обеспечение под названием сервер MySQL обрабатывает запросы клиентов и управляет доступом к данным. Он является ключевым компонентом, обеспечивающим масштабируемость и надежность MySQL. Различные модули и расширения, поддерживаемые сервером MySQL, позволяют расширять его функциональность и добавлять новые возможности.

Через командную строку или скрипты на языке Perl или PHP протокол командного интерфейса позволяет взаимодействовать с базой данных MySQL. Благодаря этому простому, но мощному интерфейсу администраторы могут управлять базами данных, выполнять запросы для проверки и обрабатывать данные. Поддержка транзакций и многопользовательский доступ являются важными компонентами MySQL. Транзакции гарантируют, что данные остаются защищенными и что информация будет правильно восстановлена в случае сбоев. MySQL также предлагает систему управления доступом, которая позволяет пользователям назначать различные уровни доступа к базам данных и таблицам, чтобы обеспечить безопасность данных.

MySQL популярен среди разработчиков и администраторов из-за его гибкой архитектуры и простоты использования. Веб-разработка, управление данными и аналитика – лишь некоторые из областей, в которых он используется. MySQL продолжает оставаться одной из ведущих систем управления базами данных, добавляя новые возможности, чтобы удовлетворить потребности современных проектов [9].

Одним из совместимых с MySQL веб-приложением является PHPMyAdmin, которое представляет из себя веб-интерфейс для администрирования СУБД MySQL.

В данной главе были рассмотрены теоретические аспекты о HTML, CSS, JS, PHP, MySQL, которые являются основными технологиями для создания web-приложений. Таким образом, HTML используется для структурирования контента, CSS - для визуального оформления, JS - для создания интерактивности на веб-странице, PHP - для обработки данных на сервере, а MySQL - для управления базами данных. В контексте разработки сайта-портфолио, эти технологии являются не только актуальными, но и практически обязательными. HTML позволяет создавать различные разделы и компоненты страницы, CSS придает им стиль и внешний вид, JS делает сайт интерактивным и динамическим, а PHP с MySQL позволяют хранить и получать данные о портфолио пользователя. Без этих технологий создание динамического и функционального сайта-портфолио будет затруднительным. HTML, CSS и JS - это основополагающие языки разработки веб-сайтов, а PHP и MySQL обеспечивают функциональность и возможность хранения информации. Их использование позволяет создавать интерактивные, адаптивные и практичные сайты-портфолио, привлекая посетителей и успешно демонстрируя достижения и навыки автора.

# Практическая часть

## **Требования к web-приложению и его характеристика**

К проекту, разрабатываемому web-приложению, были поставлены следующие требования:

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.

Также необходимо разработать базу данных для управления пользователями, используя СУБД MySQL. Реализовать добавление, редактирование, просмотр и удаление пользователей. Также предусмотреть функции авторизации и аутентификации. Более подробное описание реализации следующее:

* 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.

Далее важно отметить, что из себя представляет разрабатываемое we-приложение. Изначально, сайт-портфолио подразумевался как место, в котором будут храниться всевозможные достижения пользователя. Однако в силу возможностей реализовать такое пока что не представляется возможным, потому количество достижений, ссылок и прочих блоков для портфолио является ограниченным. Написанный сайт-портфолио является учебным макетом и прототипом одного из подобных полноценных сайтов. Планируется в последующем доработать функционал web-приложения и добавить возможность просмотра пользователями портфолио других пользователей, зарегистрированных на сайте. Также в портфолио пользователя планируется сделать личные ссылки на портфолио пользователей, чтобы они могли делиться с незарегистрированными пользователями своими персональными сайтами-портфолио со всей необходимой информации о них.

Созданное web-приложение представляет собой платформу, на которой зарегистрированный пользователь может собрать свое портфолио из блоков, подобно конструктору. Выбор блоков пользователем происходит на главной странице, далее выбранные блоки отображаются на странице «Портфолио» пользователя. В следующих параграфах будет более детально описано, какие функции присутствуют на сайте и как это реализовано.

## **Создание базы данных на MySQL в PHPMyAdmin**

Перед началом работы, необходим веб-сервер, на котором будет располагаться web-приложение. Для этого отлично подойдет XAMPP – кроссплатформенная сборка веб-сервера, содержащая Apache, MySQL, интерпретатор скриптов PHP, язык программирования Perl и большое количество дополнительных библиотек, позволяющих запустить полноценный веб-сервер. Необходимо переместить папку YDS.loc (проект) в htdocs в папке xampp. Далее необходимо провести настройку конфигурации дял запуска на локальном сервере: в папке xampp выбрать папку apache>conf>extra и отредактировать файл «httpd-vhosts.conf». В него нужно внести следующие данные:

##<VirtualHost \*:80>

##ServerAdmin webmaster@YDS.loc

##DocumentRoot "C:/xampp/htdocs/YDS.loc/www"

##ServerName YDS.loc

##ServerAlias www.YDS.loc

##ErrorLog "C:/xampp/htdocs/YDS.loc/logs/error.log"

##CustomLog "C:/xampp/htdocs/YDS.loc/logs/access.log" combined

##<Directory "C:/xampp/htdocs/YDS.loc/www">

##AllowOverride All

##Order allow,deny

##Allow from all

##</Directory>

##</VirtualHost>

После этого необходимо зайти в директорию System32>drivers>etc и отредактировать «hosts». Открываем hosts при помощи любого текстового редактора и добавляем туда строчку: # 127.0.0.1 YDS.loc.

Теперь запустим Apache и MySQL в XAMPP и откроем ссылку <http://localhost/phpmyadmin/index.php>. Это и будет PHPMyAdmin, в котором необходимо создать базу данных «your\_digital\_showcase» – так называется сайт, «Your Digital Showcase», переводя дословно «Ваша цифровая витрина». Далее необходимо создать таблицу «users» вручную или через SQL запросы: «CREATE TABLE `users` ( `id` int(10) UNSIGNED NOT NULL, `login` varchar(255) NOT NULL, `surname` varchar(255) NOT NULL, `name` varchar(255) NOT NULL, `patronymic` varchar(255) NOT NULL, `email` varchar(255) NOT NULL, `birth\_date` date NOT NULL, `tel` varchar(255) NOT NULL, `password` varchar(255) NOT NULL, `photo` varchar(255) NOT NULL, `IsAdmin` tinyint(1) NOT NULL ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci; ALTER TABLE `users` ADD PRIMARY KEY (`id`); ALTER TABLE `users` MODIFY `id` int(10) UNSIGNED NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=66; COMMIT;».

Таблица users содержит основную информацию о пользователях и имеет столбцы: id (идентификатор), login (логин), surname (фамилия), name (имя), patronymic (отчество), email (адрес электронной почты), birth\_date (дата рождения), tel (номер телефона), password (пароль), photo (аватар), IsAdmin (является ли администратором).

Для наделения пользователя правами администратора нужно в PHPMyAdmin поменять значение у выбранного пользователя с 0 на 1 в столбце IsAdmin.

Необходимо также создать вторую таблицу «users\_portfolio», в которой будет содержаться информация о том, какие данные будут отображаться у пользователя на странице «Портфолио». SQL запрос для создания этой таблицы следующий: «CREATE TABLE `users\_portfolio` ( `id` int(10) UNSIGNED NOT NULL, `job` tinyint(1) NOT NULL, `job\_place` varchar(255) NOT NULL, `job\_post` varchar(255) NOT NULL, `education` tinyint(1) NOT NULL, `education\_place` varchar(255) NOT NULL, `resume` tinyint(1) NOT NULL, `resume\_url` varchar(255) NOT NULL, `vk` tinyint(1) NOT NULL, `vk\_url` varchar(255) NOT NULL, `tg` tinyint(1) NOT NULL, `tg\_url` varchar(255) NOT NULL, `bg` varchar(255) NOT NULL ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci; ALTER TABLE `users\_portfolio` ADD PRIMARY KEY (`id`), ADD KEY `id` (`id`) USING BTREE; ALTER TABLE `users\_portfolio` MODIFY `id` int(10) UNSIGNED NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=66; COMMIT;».

Теперь база данных для разрабатываемого web-приложения готова.

## **Регистрация и авторизация пользователей, общие и схожие блоки кода**

На сайте согласно требованиям предусмотрена регистрация и авторизация пользователей. Страницы login.php для авторизации и register.php для регистрации представлены на рисунке 1 и 2.

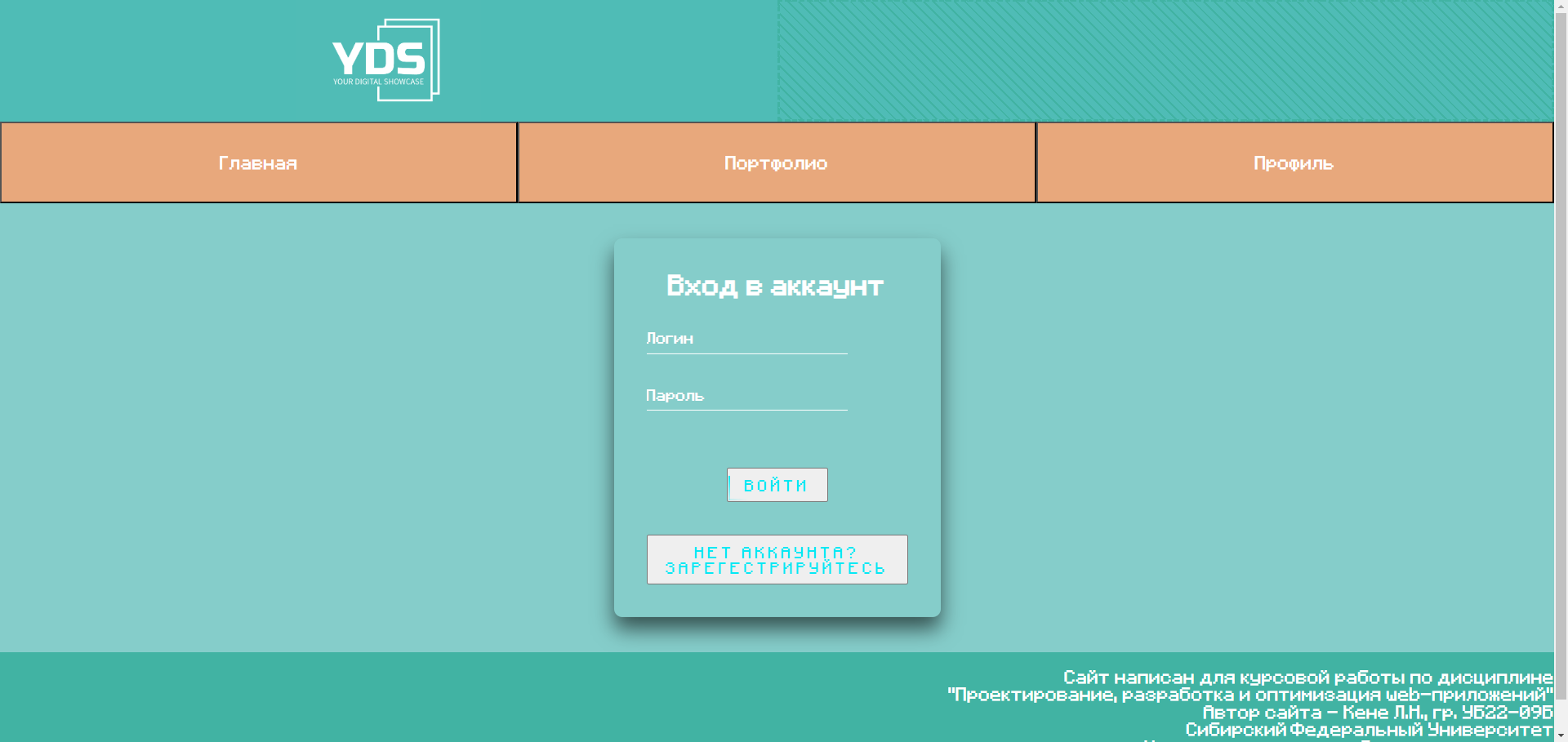


Рисунок 1 – Страница login.php

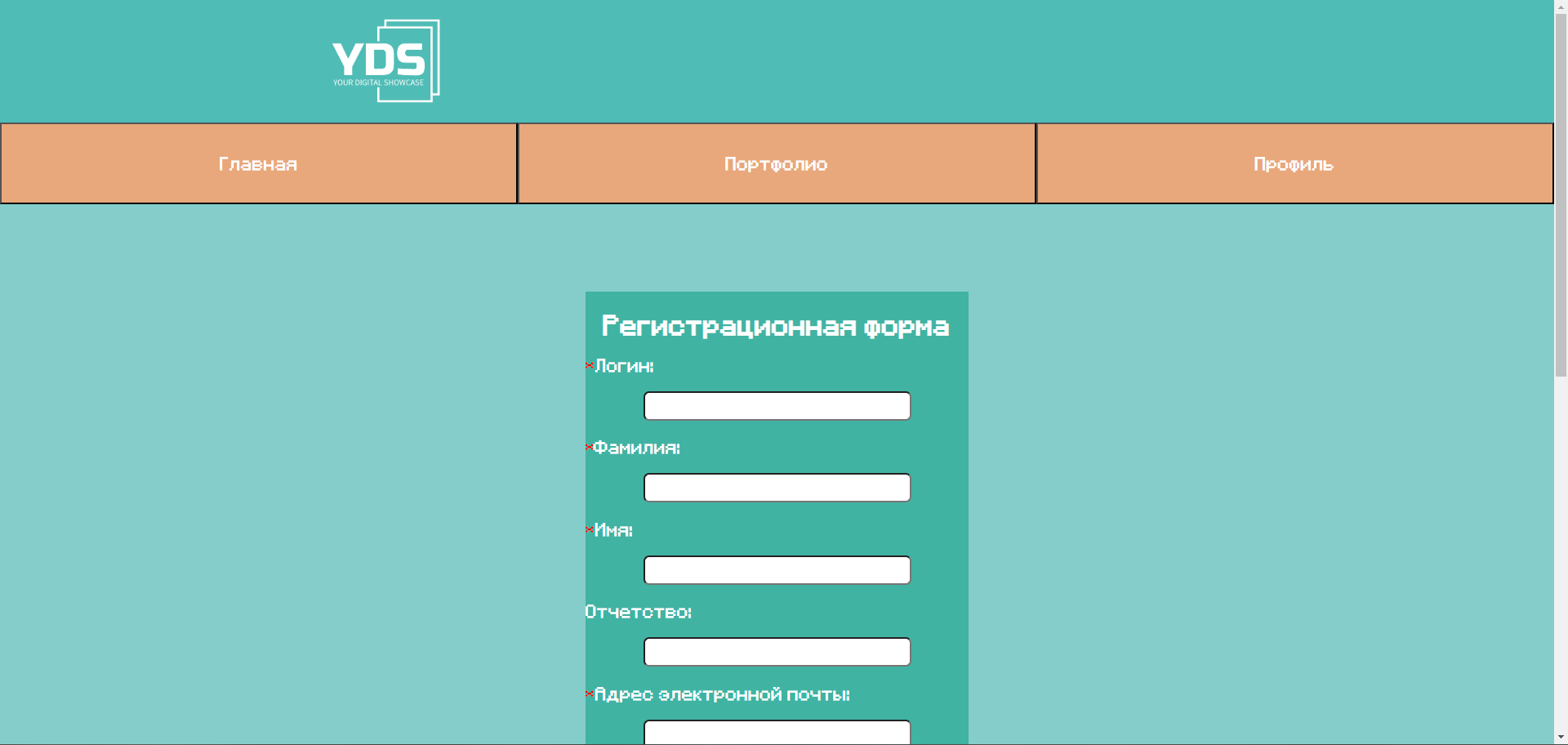


Рисунок 2 – Страница register.php

Первое, что нужно сказать о визуальной составляющей – каскадных таблицах стилей, «шапка» и «подвал» сайта везде одинаковые, практически все основные правила стилей заданы в файле main.css, который подключается в начале, а также легко приметить специальный шрифт для сайта. Main.css содержит классы, которые задают ширину блока и могут меняться в зависимости от разрешения экрана, на котором открыт сайт, а именно на телефоне это разрешение в 368px (пикселей), планшете – 760px, настольном компьютере – 1200px. Это называется адаптивной версткой (адаптивным макетом) и она применяется на каждом сайте в разработанном web-приложении. Для сброса обычных настроек CSS, установленных в браузере, также подключается файл normalize.css. Листинг стандартной заготовки под каждый сайт:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <title>Document</title>

    <link rel="stylesheet" href="CSS/normalize.css" />

    <link rel="stylesheet" href="CSS/main.css" />

</head>

<body>

    <div class="row">

        <div class="rez8 rez\_m\_8 rez\_b\_7">

            <div class="block-2 col1 logo">

                <!-- Логотип -->

                <img src="img/logo.png" class="logo">

            </div>

        </div>

        <div class="rez5 rez\_m\_5 rez\_b\_7">

            <div class="block-2 col1 <?php echo isset($\_SESSION['user\_id']) ? '' : 'dash'?>">

                <!-- Статус -->

                <div class="block-2 col1 centrify <?php echo isset($\_SESSION['user\_id']) ? '' : 'dash'?>">

                    <?php

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

                            ?>

                    Вы авторизованы

                    <?php

                        } else {?>

                    <form action="/YDS.loc/www/login.php" method="post"><button class="custom-btn btn-13">Войти</button>

                    </form>

                    <?php }?>

                </div>

            </div>

        </div>

    </div>

    <div class="row">

        <!-- Главная -->

        <form action="/YDS.loc/www/main.php" method="post"><button

                class="btn-new divlike rez9 block-1 rez\_m\_5 rez\_b\_5 col3">Главная</button></form>

        <div class="rez9 rez\_m\_8 rez\_b\_8">

            <div class="row">

                <!-- Портфолио -->

                <form action="/YDS.loc/www/portfolio.php" method="post"><button

                        class="btn-new divlike block-1 rez7 rez\_m\_7 rez\_b\_7 col2">Портфолио</button></form>

                <!-- Профиль -->

                <form action="/YDS.loc/www/profile.php" method="post"><button

                        class="btn-new divlike block-1 rez7 rez\_m\_7 rez\_b\_7 col2">Профиль</button></form>

            </div>

        </div>

    </div>

    <div class="row">

        <div class="rez9 rez\_m\_9 rez\_b\_9">

            <!-- Основная -->

            <!-- Код страницы -->

        </div>

    </div>

    <div class="row">

        <div class="rez7 rez\_m\_7 rez\_b\_7">

            <!-- Подвал -->

            <div class="block-2 col5">

                <!-- Админ -->

                <?php

                if (isset($\_SESSION['admin']) && $\_SESSION['admin']) {

                    ?>Админ, посмотри сюда:<p></p>

                <form action="/YDS.loc/www/admin\_users.php" method="post"><button class="custom-btn btn-13">Админская

                        панель</button></form>

                <?php

                } else { }

                            ?>

            </div>

        </div>

        <div class="rez7 rez\_m\_7 rez\_b\_7">

            <div class="block-2 col5 algn\_r">

                <!-- О разработчике -->

                <p>Сайт написан для курсовой работы по дисциплине

                    <br>"Проектирование, разработка и оптимизация web-приложений"

                    <br>Автор сайта - Кене Л.Н., гр. УБ22-09Б

                    <br>Сибирский Федеральный Университет

                    <br>Институт управления бизнес-процессами

                    <br>Кафедра Бизнес-информатики и моделирования бизнес-процессов

            </div>

        </div>

    </div>

</body>

</html>

Листинг 1 – Стандартная заготовка под все страницы

Для авторизации на сайте необходимо заполнить форму: ввести логин и пароль и нажать «Войти». После этого форма авторизации будет отправлена на эту же страницу и через PHP будут обработаны POST-запросы, скрытые от пользователя и посылаемые в адресной строке. В отличие от GET-запросов они являются более устойчивыми поскольку не раскрывают передаваемые данные напрямую в адресной строке, потому POST-запросы более устойчивы и к SQL-инъекциям. Листинг обработки запросов на PHP на примере login.php:

<?php session\_start();

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

        header("Location: main.php");

    }

    else{

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

        $login = $\_POST['login'];

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

        $pdo = new PDO('mysql:host=localhost;port=3306;dbname=your\_digital\_showcase', 'root', '');

        $pdo->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);

        global $pdo;

        $stmt = $pdo->prepare("SELECT \* FROM users WHERE login = ? AND password = ?");

        $stmt->execute([$login, $password]);

        $IsAuth = $stmt->fetch(PDO::FETCH\_ASSOC);

        $stmt = $pdo->prepare("SELECT id, IsAdmin FROM users WHERE login = ?");

        $stmt->execute([$login]);

        $result = $stmt->fetch(PDO::FETCH\_ASSOC);

        $id = $result['id'];

        $isAdmin = $result['IsAdmin'];

        if ($IsAuth) {

            echo "Успех";

            if ($result['IsAdmin'] == 1){

                sleep(1);

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

                $\_SESSION['admin'] = true;

                header("Location: main.php");

                $pdo = null;

                exit();

            }

            sleep(1);

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

            header("Location: main.php");

            $pdo = null;

            exit();

        }

        else {

            echo "Логин или пароль введены неправильно. Попробуйте снова.";

            header("Refresh: 3; url=login.php");

            $pdo = null;

            exit();

        }

    }

    else{

    }

    }?>

Листинг 2 – Обработка POST-запросов

При авторизации пользователю присваивается глобальная переменная $\_SESSION[‘user\_id’]. Если эта переменная есть, пользователь перенаправляется на главную страницу, если нет, то сверяются данные логина и пароля, если соответствуют, то пользователю присваивается $\_SESSION[‘user\_id’]. Если пользователь еще и является админом, то присваивается $\_SESSION[‘admin’]. И открывается дополнительный функционал – доступ к «Админской панели».

Не все правила стилей можно поместить в универсальный файл main.css, потому сверху в коде каждой страницы в теге <head> в <style> прописаны правила, которые необходимы только для конкретной страницы.

Листинг формы авторизации:

      <div class="login-box col4 algn\_c">

        <h2>Вход в аккаунт</h2>

        <form class="col4" method="post" action="">

          <div class="user-box col4">

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

            <label>Логин</label>

          </div>

          <div class="user-box col4">

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

            <label>Пароль</label>

          </div>

          <button name="submit-name" class="butto">

            <span></span>

            <span></span>

            <span></span>

            <span></span>

            Войти

          </button>

        </form>

        <form class="" action="/YDS.loc/www/register.php" method="post"><span></span>

          <span></span>

          <span></span>

          <span></span><button class=" butto">Нет аккаунта? Зарегестрируйтесь</button>

        </form>

      </div>

      <br>

      <div class="col4 ">

      </div>

    </div>

Листинг 3 – Форма авторизации

Листинг регистрационной формы:

<script src="/YDS.loc/www/JS/reg\_scr.js"></script>

            <div class="col5 algn\_c">

                <h2 class='align\_c'>Регистрационная форма</h2>

                <form class="col5 algn\_c" method="post" action="" id="Register">

                    <label for="login"><span class="red\_form">\*</span>Логин:</label><br>

                    <input class="full" type="text" id="login" name="login" placeholder="\*Логин" required><br><br>

                    <label for="surname"><span class="red\_form">\*</span>Фамилия:</label><br>

                    <input class="full" type="text" id="surname" name="surname" placeholder="\*Фамилия" required><br><br>

                    <label for="name"><span class="red\_form">\*</span>Имя:</label><br>

                    <input class="full" type="text" name="name" id="name" placeholder="\*Имя" required><br><br>

                    <label for="patronymic">Отчетство:</label><br>

                    <input class="full" type="text" id="patronymic" name="patronymic" placeholder="Отчество"><br><br>

                    <label for='email'><span class="red\_form">\*</span>Адрес электронной почты:</label><br>

                    <input class="full" type='email' id='email' name='email' placeholder="\*Адрес вашей электронной почты" class="email" pattern="^[\w]{1}[\w\-\.]\*@[\w\-]+\.[a-z]{2,4}$" required /><br>

                    <p class=" algn\_l">

                        <span class="red\_form">\*</span>Дата рождения:<br>

                    </p>

                    <input class="full" type="date" name="date\_of\_birth" id="date\_of\_birth" required><br><br>

                    <label for='tel'><span class="red\_form">\*</span> Номер телефона:</label><br>

                    <input class="full" type='tel' id='tel' name='tel' placeholder="\* Ваш номер телефона +7(\_\_\_)\_\_\_-\_\_-\_\_" class="tel" required pattern="^(?:\+)?\d(?:[ \(]+)?\d{3}(?:[ \)]+)?\d{3}(?:[- ]+)?\d{2}(?:[- ]+)?\d{2}$" /><br><br>

                    <label for="password"><span class="red\_form">\*</span>Пароль:</label><br>

                    <input class="full" type="password" id="password" name="password" placeholder="\*Пароль" required><br><br>

                    <label for="2\_password"><span class="red\_form">\*</span>Повторите пароль:</label><br>

                    <input class="full" type="password" id="password2" name="2\_password" placeholder="\*Пароль еще раз" required><br><br>

                    <label for="avatar\_img">Ссылка на аватар:</label><br>

                    <input class="full" type="text" id="avatar\_img" name="avatar\_img" placeholder="Ссылка на аватар"><br><br>

                    <p class=" centrify"><button class="full algn\_c centrify">Зарегистрироваться</button></p>

                    <span class="red\_form">\*</span> - обязательные поля для заполнения

                    <?

                    if (isset($\_POST['login']) && isset($\_POST['surname']) && isset($\_POST['name']) && isset($\_POST['email']) && isset($\_POST['date\_of\_birth']) && isset($\_POST['tel']) && isset($\_POST['password'])) {

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

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

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

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

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

                        $date\_of\_birth = $\_POST['date\_of\_birth'];

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

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

                        if (!isset($\_POST['avatar\_img'])) {

                            $photo = "https://avatars.mds.yandex.net/i?id=24ea928ef83997ec3abeaa5e719aba654499767f-9852567-images-thumbs&ref=rim&n=33&w=200&h=200";

                        }

                        $photo = $\_POST['avatar\_img'];

                        $pdo = new PDO('mysql:host=localhost;port=3306;dbname=your\_digital\_showcase', 'root', '');

                        global $pdo;

                        $stmt = $pdo->prepare("INSERT INTO users\_portfolio (job, education, resume, vk, tg, bg) VALUES (?, ?, ?, ?, ?, ?)");

                        $stmt->execute(["0", "0", "0", "0", "0", "https://uploads-ssl.webflow.com/5c8d3f1d0fcf5a3ae2c52ba3/5d826e8c2e88f893d9cdd560\_texture-p-1600.jpeg"]);

                        $user\_id = $pdo->lastInsertId();

                        $stmt = $pdo->prepare("INSERT INTO users (id, login, surname, name, patronymic, email, birth\_date, tel, password, photo, IsAdmin) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)");

                        if (isset($\_POST['patronymic']) && isset($\_POST['photo'])) {

                            $stmt->execute([$user\_id, $login, $surname, $name, $patronymic, $email, $date\_of\_birth, $tel, $password, $photo, "0",]);

                            session\_start();

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

                            echo $login, ", Вы успешно зарегистрированы! Переадресация...";

                            sleep(3);

                            header("Location: main.php");

                            $pdo = null;

                            exit();

                        } else if (isset($\_POST['patronymic'])) {

                            $stmt->execute([$user\_id, $login, $surname, $name, $patronymic, $email, $date\_of\_birth, $tel, $password, "https://avatars.mds.yandex.net/i?id=24ea928ef83997ec3abeaa5e719aba654499767f-9852567-images-thumbs&ref=rim&n=33&w=200&h=200", "0",]);

                            session\_start();

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

                            echo $login, ", Вы успешно зарегистрированы! Переадресация...";

                            sleep(3);

                            header("Location: main.php");

                            $pdo = null;

                            exit();

                        } else if (isset($\_POST["photo"])) {

                            $stmt->execute([$user\_id, $login, $surname, $name, "", $email, $date\_of\_birth, $tel, $password, $photo, "0",]);

                            session\_start();

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

                            echo $login, ", Вы успешно зарегистрированы! Переадресация...";

                            sleep(3);

                            header("Location: main.php");

                            $pdo = null;

                            exit();

                        } else if (!isset($\_POST["patronymic"]) && !isset($\_POST['photo'])) {

                            $stmt->execute([$user\_id, $login, $surname, $name, "", $email, $date\_of\_birth, $tel, $password, "https://avatars.mds.yandex.net/i?id=24ea928ef83997ec3abeaa5e719aba654499767f-9852567-images-thumbs&ref=rim&n=33&w=200&h=200", "0",]);

                            session\_start();

                            echo $login, ", Вы успешно зарегистрированы! Переадресация...";

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

                            sleep(3);

                            header("Location: main.php");

                            $pdo = null;

                            exit();

                        }

                    } else {

                        echo "<br>Пожалуйста, заполните все поля формы.";

                    }

                    ?>

                </form>

            </div>

Листинг 4 – Регистрационная форма

Важно отметить, что в регистрационной форме подключается файл reg\_scr.js – файл JavaScript, который обрабатывает заполненные поля и ищет ошибки. Часто применяются так называемые паттерны для проверки. Листинг программы reg\_scr.js:

window.addEventListener('DOMContentLoaded', function() {

  document.getElementById('Register').addEventListener('submit', function(event) {

    event.preventDefault();

    validateForm();

  });

});

function validateForm() {

  var login = document.getElementById('login').value;

  var surname = document.getElementById('surname').value;

  var name = document.getElementById('name').value;

  var patronymic = document.getElementById('patronymic').value;

  var email = document.getElementById('email').value;

  var date\_of\_birth = document.getElementById('date\_of\_birth').value;

  var tel = document.getElementById('tel').value;

  var password = document.getElementById('password').value;

  var password2 = document.getElementById('password2').value;

  var avatar\_img = document.getElementById('avatar\_img').value;

  var errors = [];

  loginsigns = /^[a-zA-Z]+$/;

  if (!loginsigns.test(login)) {

    errors.push('Логин должен содержать только латинские буквы');

  }

  var fiosigns = /^[а-яА-Яa-zA-Z]+$/;

  if (!fiosigns.test(surname)) {

    errors.push('Фамилия должна содержать только русские или латинские буквы');

  }

  if (!fiosigns.test(name)) {

    errors.push('Имя должно содержать только русские или латинские буквы');

  }

  if (patronymic){

  if (!fiosigns.test(patronymic)) {

    errors.push('Отчество должно содержать только русские или латинские буквы');

  }}

  emailsigns = /^[\w]{1}[\w-\.]\*@[\w-]+\.[a-z]{2,4}$/i;

  if (!emailsigns.test(email)) {

    errors.push('Проверьте, правильно ли заполнено поле "Электронная почта"');

  }

  if (!date\_of\_birth) {

    errors.push('Проверьте, правильно ли заполнено поле "Дата рождения"');

  }

  var phonesigns = /^(?:\+)?\d(?:[ (]+)?\d{3}(?:[ )]+)?\d{3}(?:[- ]+)?\d{2}(?:[- ]+)?\d{2}$/;

  if (!phonesigns.test(tel)) {

    errors.push('Телефон должен содержать только цифры и соответствовать одному из форматов записи');

  }

  var passwordsigns = /^(?=.\*[A-Za-z])(?=.\*\d)[A-Za-z\d]{8,}$/;

  if (!passwordsigns.test(password)){

    errors.push('Пароль должен содержать цифры, латинские буквы и быть длиной от 8 символов');

  }

  if (password != password2) {

    errors.push('Пароли не совпадают');

  }

  if (avatar\_img){

  source = /^(http:\/\/www\.|https:\/\/www\.|http:\/\/|https:\/\/)?[a-z0-9]+([\-\.]{1}[a-z0-9]+)\*\.[a-z]{2,5}(:[0-9]{1,5})?(\/.\*)?$/;;

  if (!source.test(avatar\_img)) {

    errors.push('Указана недействительная ссылка на фотографию');

  }}

  if (errors.length > 0) {

    alert("Ошибки при заполнении");

    document.getElementById("errorDiv").innerHTML = errors.join(". <br>");

    // alert(errors.join(", "));

  }

  else{

    document.getElementById('Register').submit();

  }

}

Листинг 5 – Файл reg\_scr.js

Если есть ошибки, то они выводятся в блок <div id=”errorDiv”>, если нет, то форма подтверждается, отправляется и обрабатывается на стороне сервера, происходит добавление пользователя в таблицы users и users\_portfolio.

При регистрации пользователю присваивается статус авторизированного пользователя.

## **Основные страницы – «Главная», «Портфолио», «Профиль»**

Основными страницами, на которые можно перейти, нажав на кнопки в шапке, являются «Главная», «Портфолио» и «Профиль». «Главную» страницу могут просматривать и неавторизованные пользователи, и изучить, какие блоки в свое портфолио они могут получить. Если неавторизованный пользователь нажмет на «Портфолио» или «Профиль», его перекинет на страницу авторизации. На рисунке 3 показана страница «Главная» для неавторизованного пользователя, и на рисунках 4 и 5 показаны страницы «Портфолио» и «Профиль» для авторизованных пользователей.



Рисунок 3 – Страница «Главная»

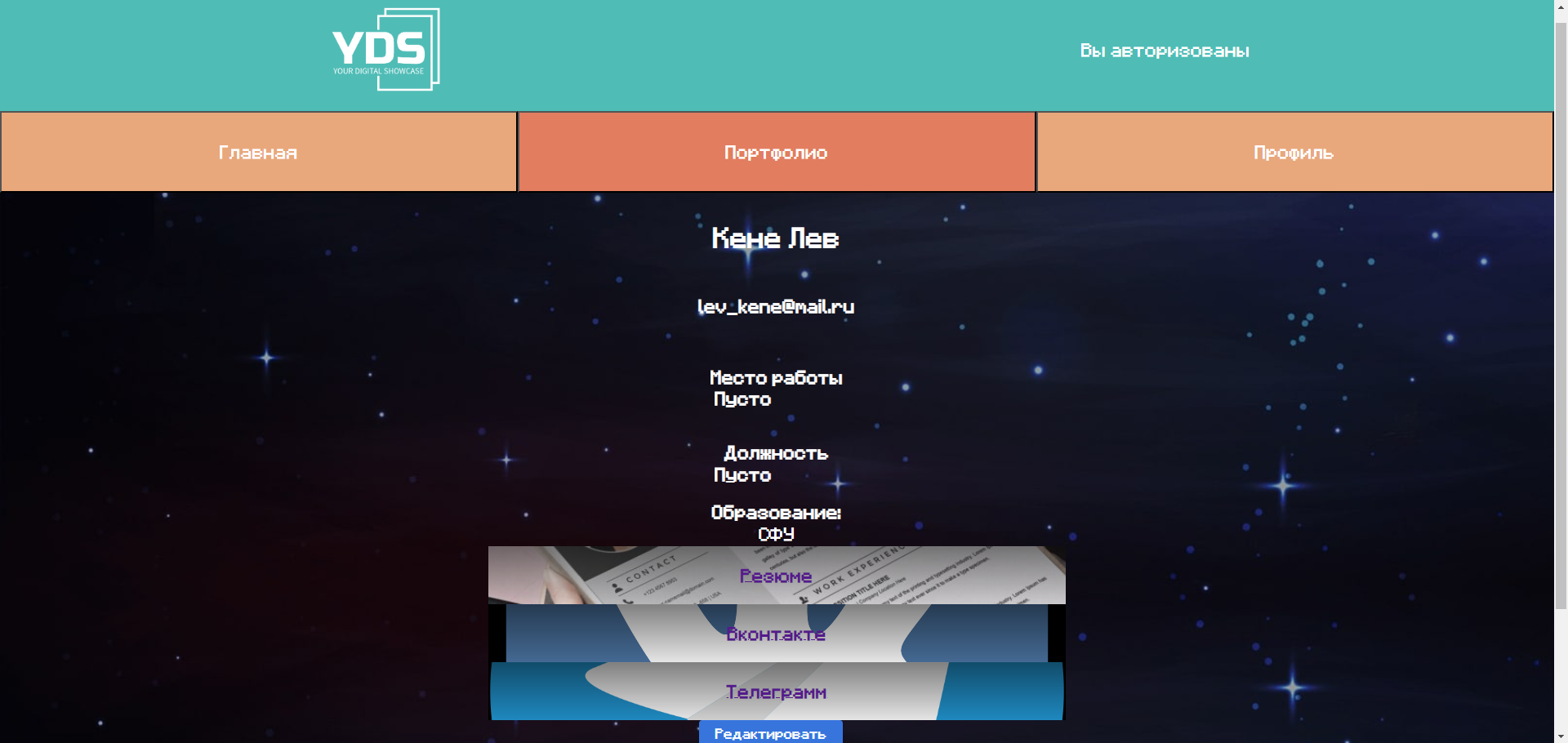


Рисунок 4 – Страница «Портфолио»



Рисунок 5 – Страница «Профиль»

На рисунке 3 видно, что в правом верхнем углу есть кнопка для авторизации, если пользователь не авторизован, тогда как на рисунке 4 и 5 видна надпись: «Вы авторизованы». На рисунке 5 также бросается в глаза кнопка, ведущая в «Админскую панель», она отображается при условии, что авторизованный пользователь является администратором.

Функционал «Главной» страницы заключается в том, что на ней для портфолио предлагается выбрать блоки. Нужно осуществить выбор и нажать на блок «Сохранить». После этого на странице «Портфолио» отобразятся выбранные блоки – на рисунке 4 показаны все возможные блоки. Также в «Портфолио» можно задать свой фон.

Чтобы изменить то, что содержится в выбранных блоках для портфолио, необходимо нажать на кнопку «Редактировать», она отправит на port\_edit.php.

На странице «Профиль» мы видим личную информацию пользователя и действия – редактировать профиль, удалить профиль, выйти. Чтобы изменить личную информацию необходимо нажать на кнопку «Редактировать», она отправит на edit.php. При нажатии на удалить вынесется предупреждение на удаление, и при согласии запись о пользователе удалится из двух таблиц в базе данных, реализовано через скрипт внутри. Кнопка «Выйти» – очищает глобальную переменную $\_SESSION и пользователь выходит из своей учетной записи.

## **Дополнительные страницы**

К дополнительным страницам можно отнести страницы для редактирования портфолио и личных данных – port\_edit.php и edit.php.

Страницы представляют собой формы, в которые уже внесены некоторые данные, которые были получены из базы данных. К формам подключены JavaScript файлы для проверки на заполнение – reg\_scr.js и port\_scr.js. Страницы для редактирования представлены на рисунках 6 и 7.

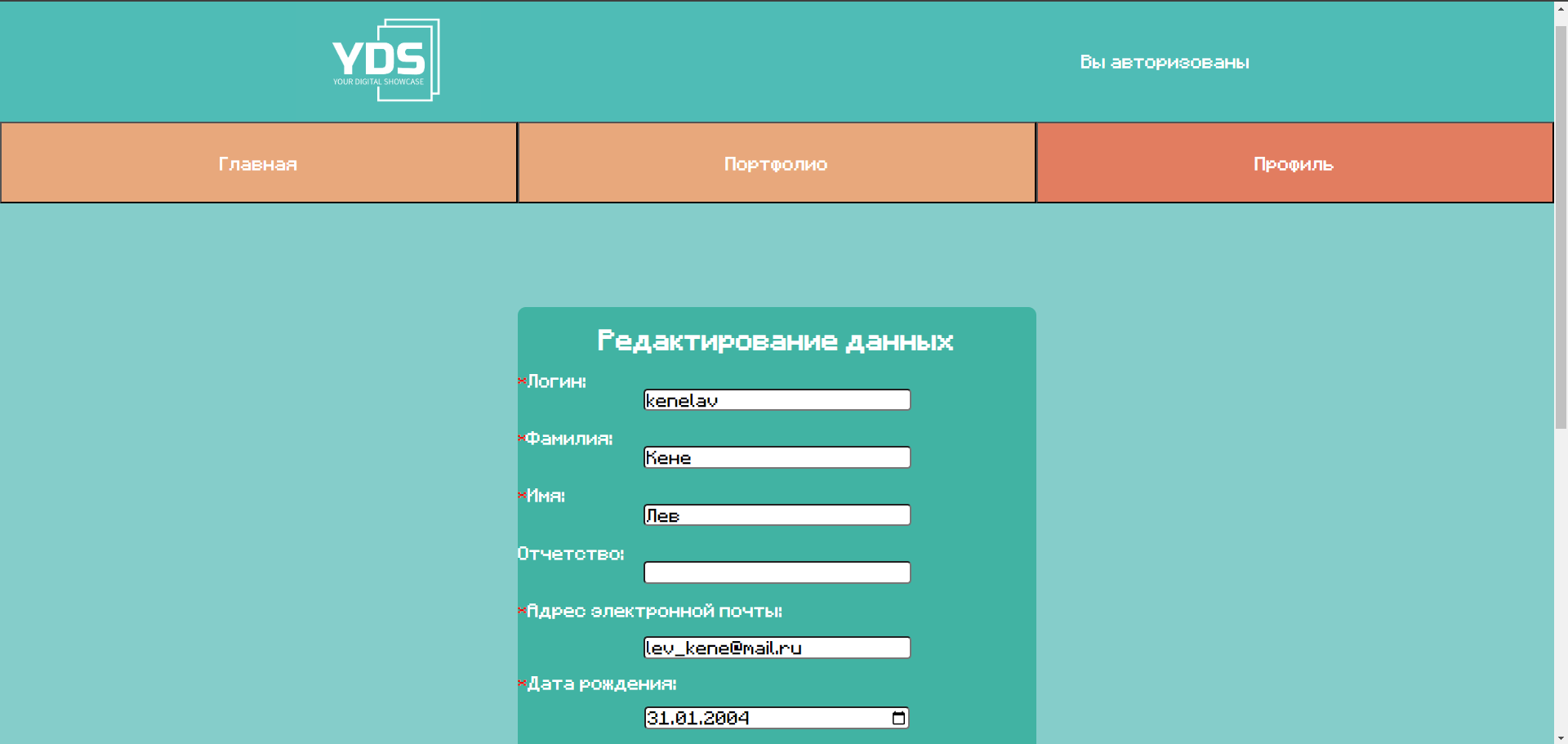


Рисунок 6 – Страница редактирования данных в «Профиль»

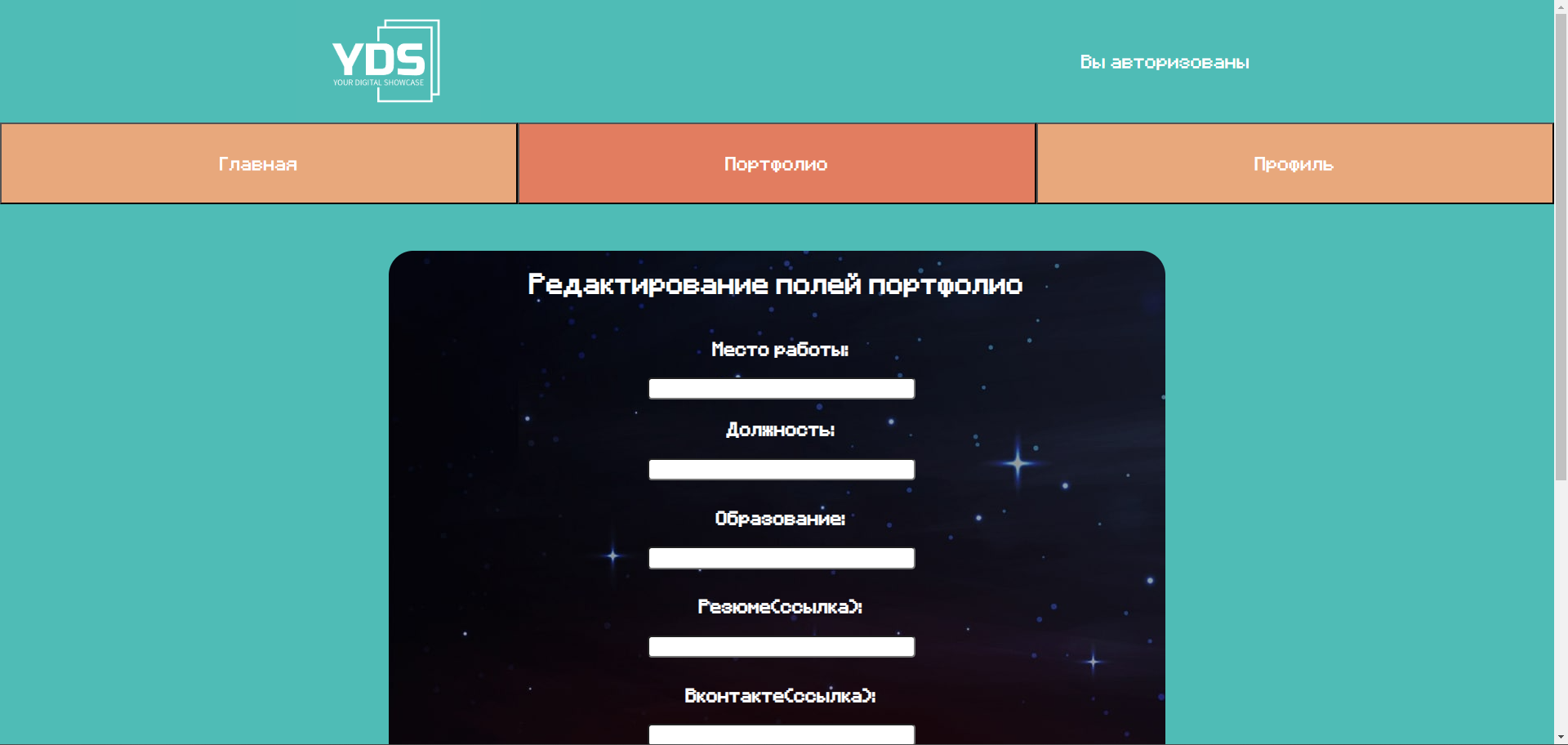


Рисунок 7 – Страница редактирования данных в «Портфолио»

Ошибки при вводе выводятся над формами.

Также вполне очевидно, что пользователь может изменять и просматривать только свои данные.

## **«Админская панель»**

Как описывалось ранее, если администратор был авторизован в системе, то ему отображается кнопка для перехода в «админскую панель».

Главная страница в админской панели – таблица «users» из базы данных, строки в которой можно редактировать данные определенного пользователя и удалять его данные, рисунок 8. Или же добавить пользователя, тогда запись о нем появится в двух таблицах как при регистрации, рисунок 9. Так как редактирование и добавление пользователей связано с заполнением, к ним привязаны JavaScript файлы, которые проверяют заполнение отправляемых на сервер форм.

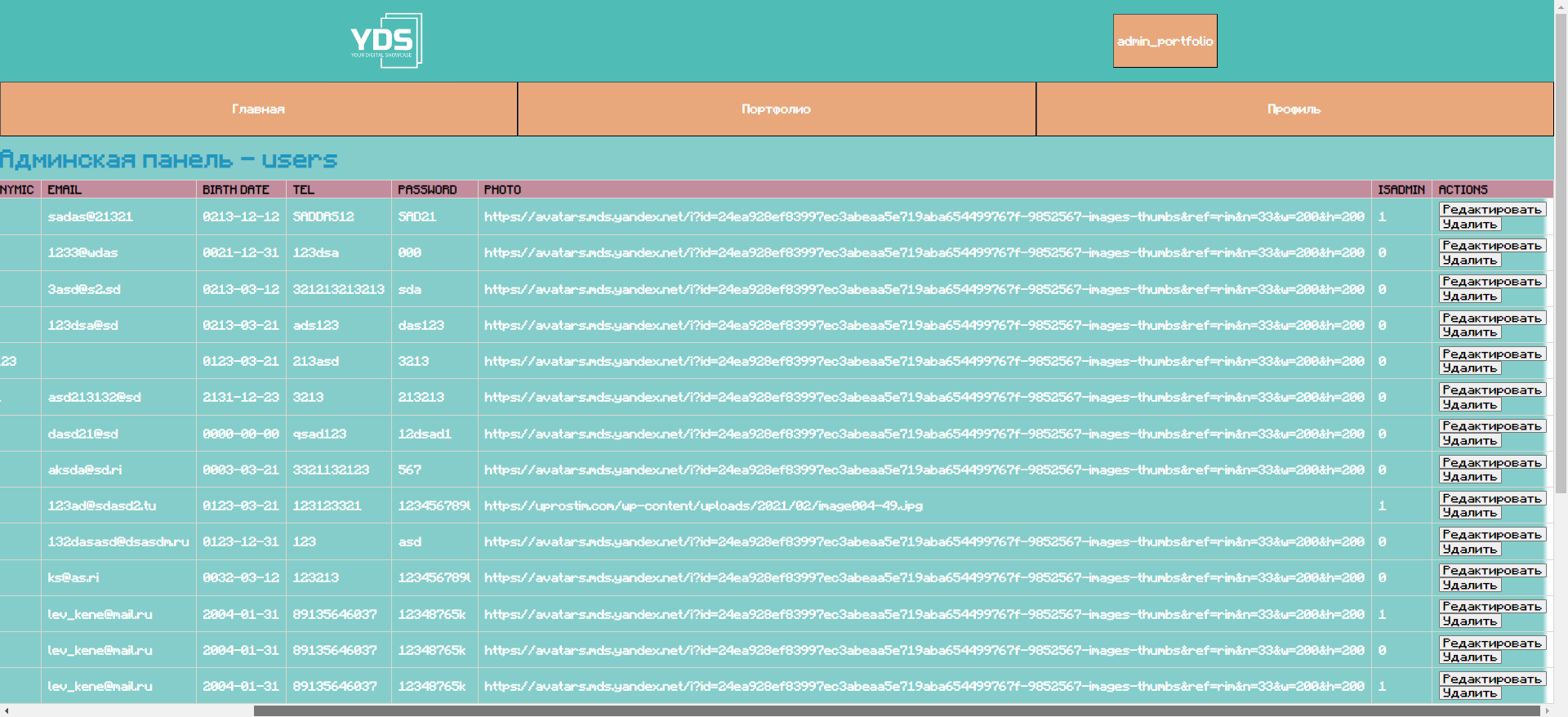


Рисунок 8 – Админская панель «users»

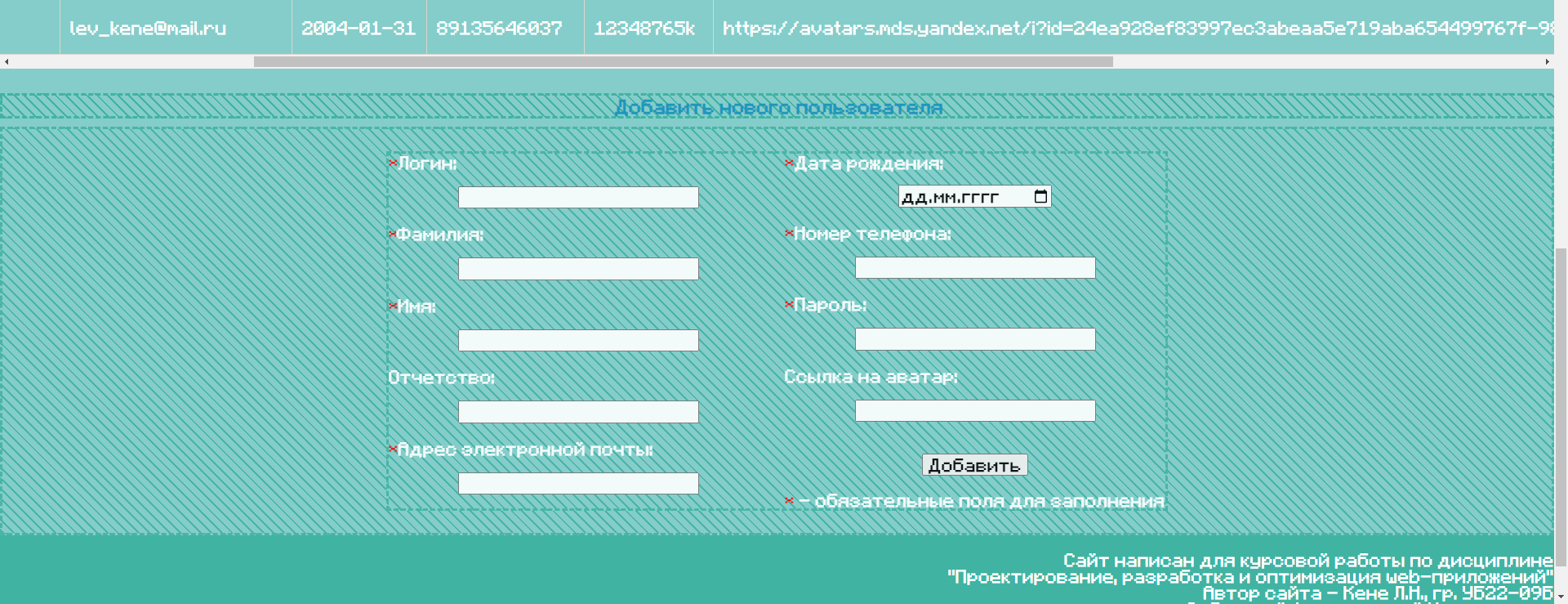


Рисунок 9 – Форма добавления нового пользователя

Сверху справа есть кнопка ведущая на вторую таблицу – users\_portfolio. На этой странице не доступно добавление нового пользователя, но доступно удаление пользователей и редактирование данных, рисунок 10.

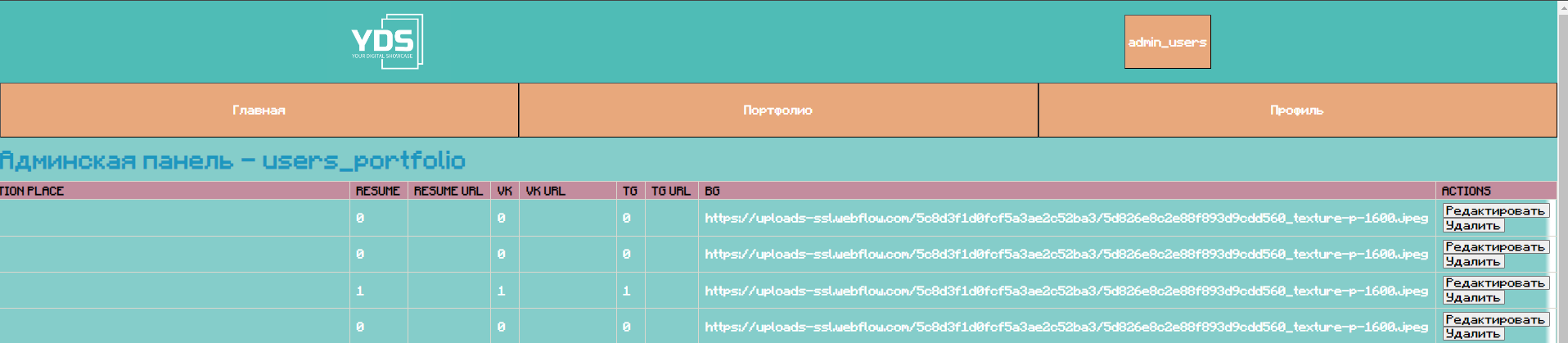


Рисунок 10 – Админская панель «users\_portfolio»

# ЗАКЛЮЧЕНИЕ

Таким образом, в ходе работы была подробно описана разработка web-приложения с применением технологий HTML, CSS, Javascript, PHP, MySQL на тему «Портфолио». Сайт полностью удовлетворяет установленным требованиям и функционирует без ошибок. Web-приложение состоит из: 9 PHP страниц – login.php, register.php, main.php, portfolio.php, profile.php, edit.php, port\_edit.php, admin\_users.php, admin\_portfolio.php; 4 JavaScript файлов – reg\_scr.js, port\_scr.js, admin\_scr.js, admin\_port\_scr.js; 2 таблиц в базе данных your\_digital\_showcase на MySQL; 2 CSS файлов – main.css и normalize.css. Поставленная цель достигнута.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Основы HTML // MDN Web Docs // developer.mozilla.org: [сайт]. – 2023 –. – URL: https://developer.mozilla.org/ru/docs/Learn/Getting\_started\_with\_the\_web/HTML\_basics
2. Язык HTML: что это такое и как он работает // Skillbox Media // skillbox.ru: [сайт]. – 2022 –. – URL: <https://skillbox.ru/media/code/chto_takoe_html/>
3. Самоучитель CSS // htmlbook.ru // htmlbook.ru: [сайт]. – 2010 –. – URL:https://htmlbook.ru/samcss
4. Грошкова А.А., Махова А.И. Преимущества использования CSS // Актуальные проблемы авиации и космонавтики. 2017. №13. URL: <https://cyberleninka.ru/article/n/preimuschestva-ispolzovaniya-css>
5. Основы CSS // MDN Web Docs // developer.mozilla.org: [сайт]. – 2023 –. – URL: <https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/CSS_basics>
6. JavaScript // MDN Web Docs // developer.mozilla.org: [сайт]. – 2023 –. – URL: <https://developer.mozilla.org/ru/docs/Web/JavaScript>
7. Введение в JavaScript // JavaScript.ru // learn.javascript.ru: [сайт]. – 2024 –. – URL: <https://learn.javascript.ru/intro>
8. Что такое PHP? // PHP // php.net: [сайт]. – 2024 –. – URL: <https://www.php.net/manual/ru/intro-whatis.php>
9. MySQL // Skillfactory Media // blog.skillfactory.ru: [сайт]. – 2023 –. – URL: https://blog.skillfactory.ru/glossary/mysql/