Министерство образования Республики Беларусь

Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»

Факультет информационных технологий и управления

Кафедра информационных технологий автоматизированных систем

ЛАБОРАТОРНАЯ РАБОТА №1

«Основы *HTML* и *CSS*»

по дисциплине

«Интернет технологии и веб-программирование»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | ст. гр. 220602  В. И. Баханович |
|  |  |  |
| Проверил: |  | А. Л. Гончаревич |

Минск 2025

СОДЕРЖАНИЕ

[Введение 3](#_Toc159000268)

[1 Постановка задачи 4](#_Toc159000269)

[2 Теоретическая часть 5](#_Toc159000270)

[3 Ход работы 7](#_Toc159000271)

[Заключение 1](#_Toc159000272)2

**ВВЕДЕНИЕ**

*HTML* и *CSS* являются фундаментальными технологиями для создания веб-страниц. *HTML* (*HyperText Markup Language*) представляет собой язык разметки, который определяет структуру и содержание веб-страницы, используя различные теги для обозначения заголовков, параграфов, списков и других элементов.

*CSS* (*Cascading Style Sheets*) дополняет *HTML*, отвечая за визуальное оформление страницы. С помощью *CSS* разработчики могут задавать цвета, размеры, расположение элементов, анимации и другие стилевые характеристики. Стили могут быть встроены непосредственно в *HTML*-документ или храниться в отдельных файлах.

Взаимодействие *HTML* и *CSS* позволяет создавать современные, привлекательные и отзывчивые веб-страницы. Правильное использование этих технологий обеспечивает корректное отображение контента на различных устройствах и в разных браузерах, что является важным аспектом веб-разработки.

**1 ПОСТАНОВКА ЗАДАЧИ**

Для выполнения лабораторной работы требуется:

* изучить теоретический материал по основам *HTML* и *CSS*;
* установить и настроить текстовый редактор *VS CODE,* включая необходимые расширения;
* спроектировать структуру веб-сайта с использованием макета *Figma*;
* разработать веб-сайт с использованием *HTML* и *CSS* согласно заданию варианта;
* развернуть разработанный веб-сайт на ресурсе *Github Pages.*

Цель лабораторной работы ­– разработка веб-сайта с использованием *HTML* и *CSS*.

**2 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ**

*HTML* (*HyperText Markup Language*) – язык разметки, предназначенный для создания структуры веб-страниц. *HTML* определяет расположение элементов на странице, таких как текст, изображения, ссылки и мультимедийные объекты. Использование *HTML* позволяет создавать веб-документы, которые браузеры интерпретируют и отображают пользователям.

Основные возможности *HTML*:

* Создание структуры веб-страницы – теги *HTML* позволяют задавать заголовки, параграфы, списки, таблицы и другие элементы контента.
* Гиперссылки – элементы <*a*> обеспечивают переход между страницами, создавая удобную навигацию.
* Работа с мультимедиа – тег <*img*> позволяет встраивать изображения на веб-страницы.
* Формы для ввода данных – элементы <*form*> и <*input*> позволяют пользователям вводить и отправлять данные, например, в формах регистрации или поиска.

*CSS* (*Cascading Style Sheets*) – каскадные таблицы стилей, отвечающие за внешний вид *HTML*-документов. *CSS* позволяет задавать цвета, шрифты, размеры элементов и другие стилистические параметры веб-страницы.

Основные возможности *CSS*:

* Оформление текста – изменение цвета, размера, шрифта и отступов текста на странице.
* Настройка фона и границ – задание фона для элементов, создание теней, градиентов и границ.
* Размещение элементов – управление позиционированием блоков, настройка отступов, выравнивание контента с помощью *flexbox* и *grid*.
* Адаптивный дизайн – использование медиазапросов для создания веб-страниц, оптимизированных под разные размеры экранов.

Для создания веб-страниц *HTML* и *CSS* используются совместно. *HTML* отвечает за структуру, а *CSS* – за стилизацию и внешний вид. Взаимодействие между ними осуществляется несколькими способами:

* Встроенные стили – *CSS*-код добавляется непосредственно в атрибут *style* внутри *HTML*-тега.
* Внутренние стили – стили указываются внутри тега <*style*> в разделе <*head*> документа.
* Внешние таблицы стилей – *CSS*-код хранится в отдельном файле .*css*, который подключается к HTML-документу через <*link*>.

Основные принципы использования *HTML* и *CSS*:

1. Создание структуры страницы – с помощью *HTML* создаются элементы, определяющие контент веб-страницы.
2. Применение стилей – *CSS* позволяет задать оформление элементов, делая страницу удобочитаемой и эстетически привлекательной.
3. Использование селекторов – *CSS* применяет стили к элементам HTML через классы, идентификаторы и атрибуты.
4. Адаптация страницы – благодаря гибким настройкам *CSS*, веб-страницы могут корректно отображаться на различных устройствах, включая смартфоны и планшеты.

**3 ХОД РАБОТЫ**

По заданию лабораторной работы необходимо установить и настроить локальный сервер для разработки на *PHP*, установить и настроить СУБД *MySQL,* создать несколько таблиц с использованием *SQL* и написать скрипты на *PHP* для подключения к базе данных и выполнения *SQL*-запросов к БД.

Для работы установим и настроем локальный сервер *XAMPP.* После установки сервера можно выбрать определенные модули и запустить их на определенном порту. Контрольная панель для запуска сервера изображена на рисунке 1.

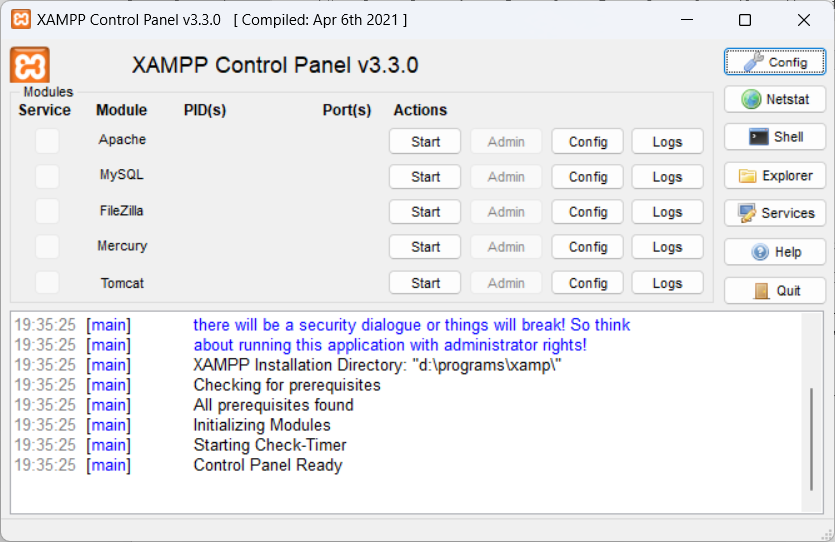


Рисунок 1 – Контрольная панель *XAMPP*

Для работы запустим модули *Apache* и *MySQL.* В папке, где установлен сервер *XAMPP* есть папка *htdocs*. Эта папка является корневой директорией, где находятся все *PHP* файлы и файлы *HTML* и *CSS.*

При открытии вею-приложения пользователь попадает на главную страницу, показанную на рисунке 2.

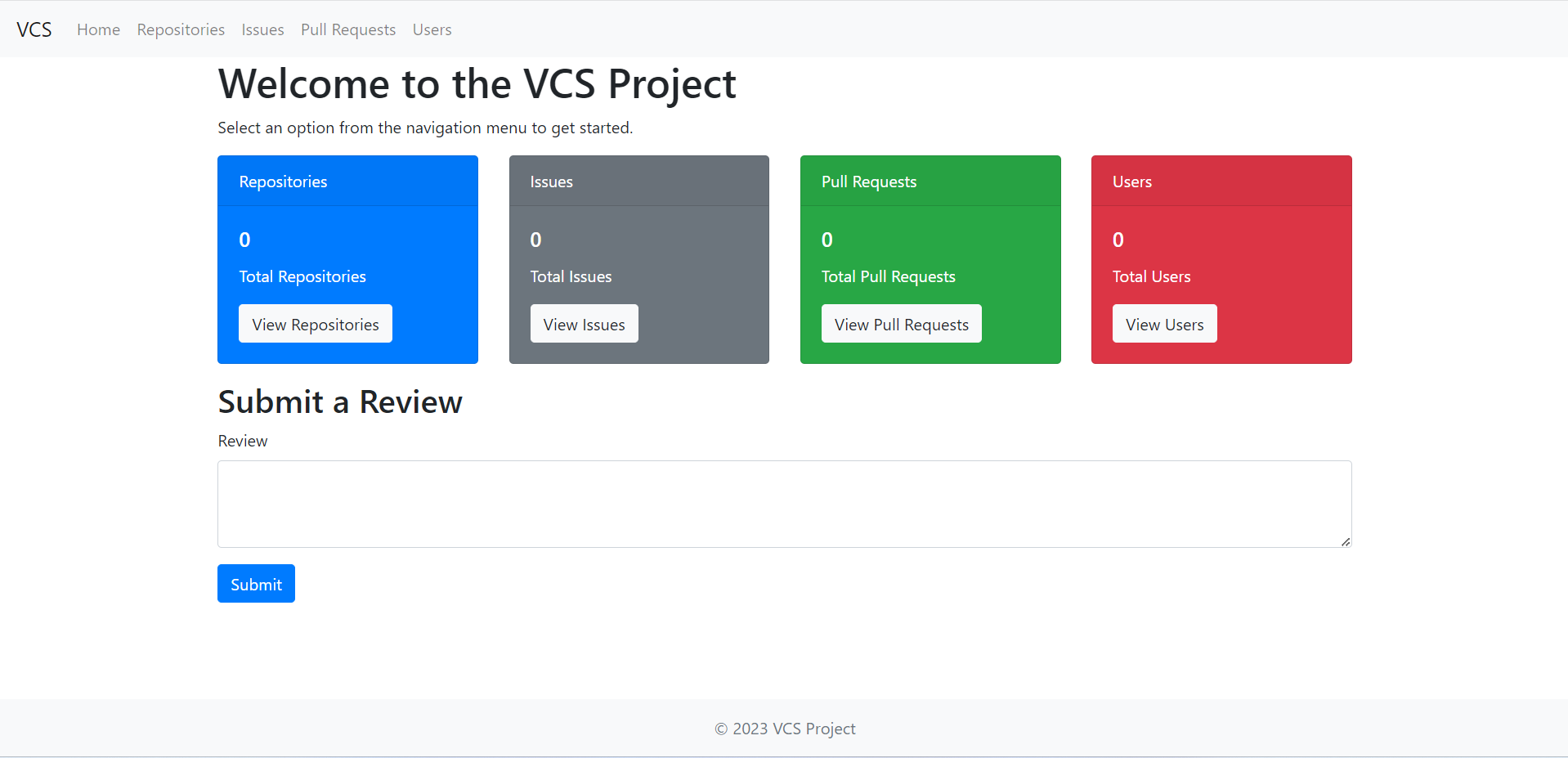


Рисунок 2 – Главная страница

Если пользователь нажмет на кнопку просмотра одной из сущностей либо воспользуется меню навигации, он попадет на страницу отображения данных из соответствующей таблицы в *MySQL*. Для пример на рисунке 3 продемонстрирована страница просмотра репозиториев. Страницы просмотров остальных сущностей выглядят аналогично.

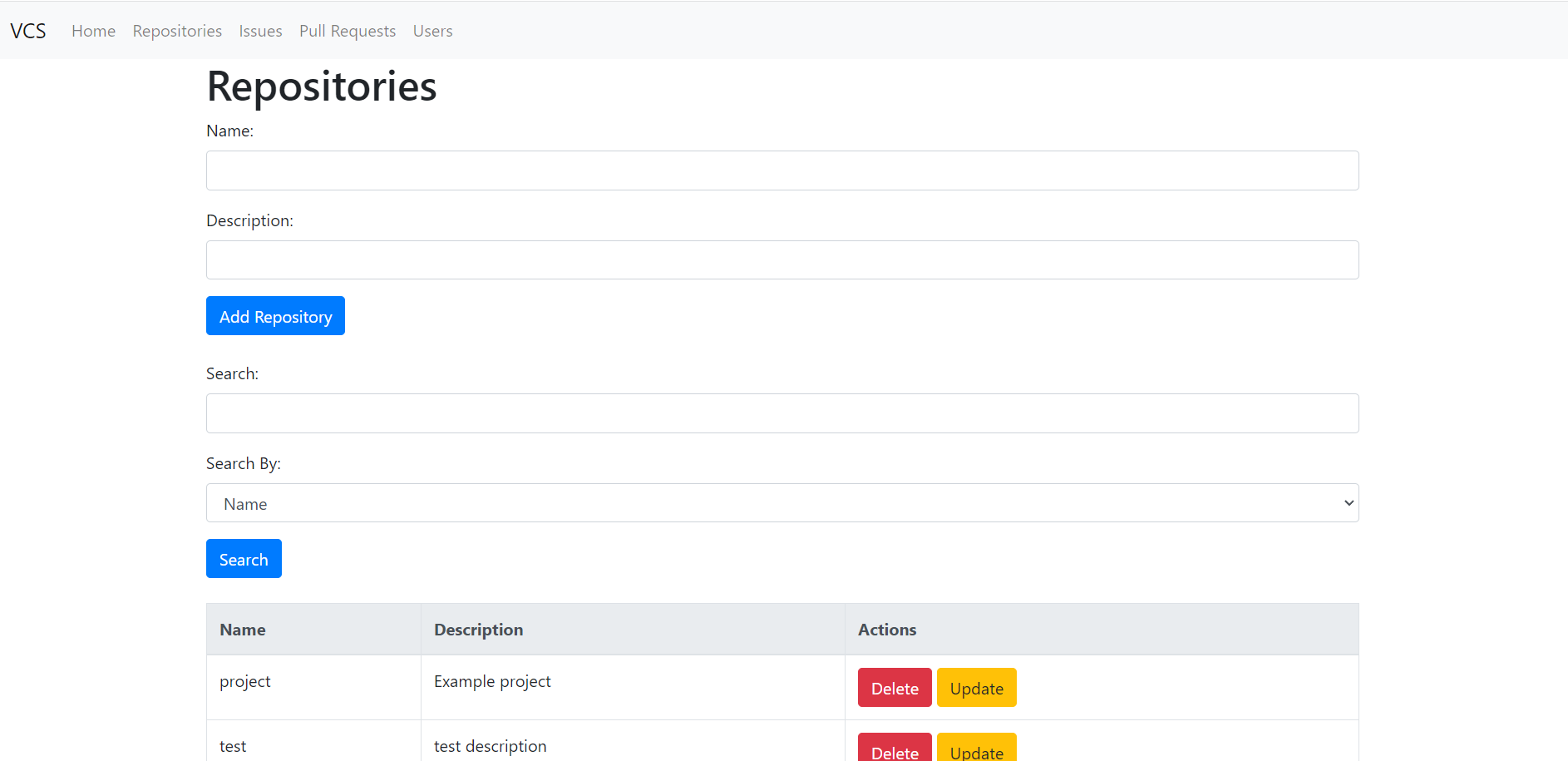


Рисунок 3 – Страница просмотра репозиториев

Реализовано добавление сущности в базу данных. Для этого необходимо ввести желаемые данные в поля, расположенные над таблицей с данными и нажать кнопку добавления. В данном примере с репозиториями для добавления следует нажать кнопку «*Add Repository*», как показано на рисунке 4. После добавления новой записи она отобразится в таблице с данными, как показано на рисунке 5.

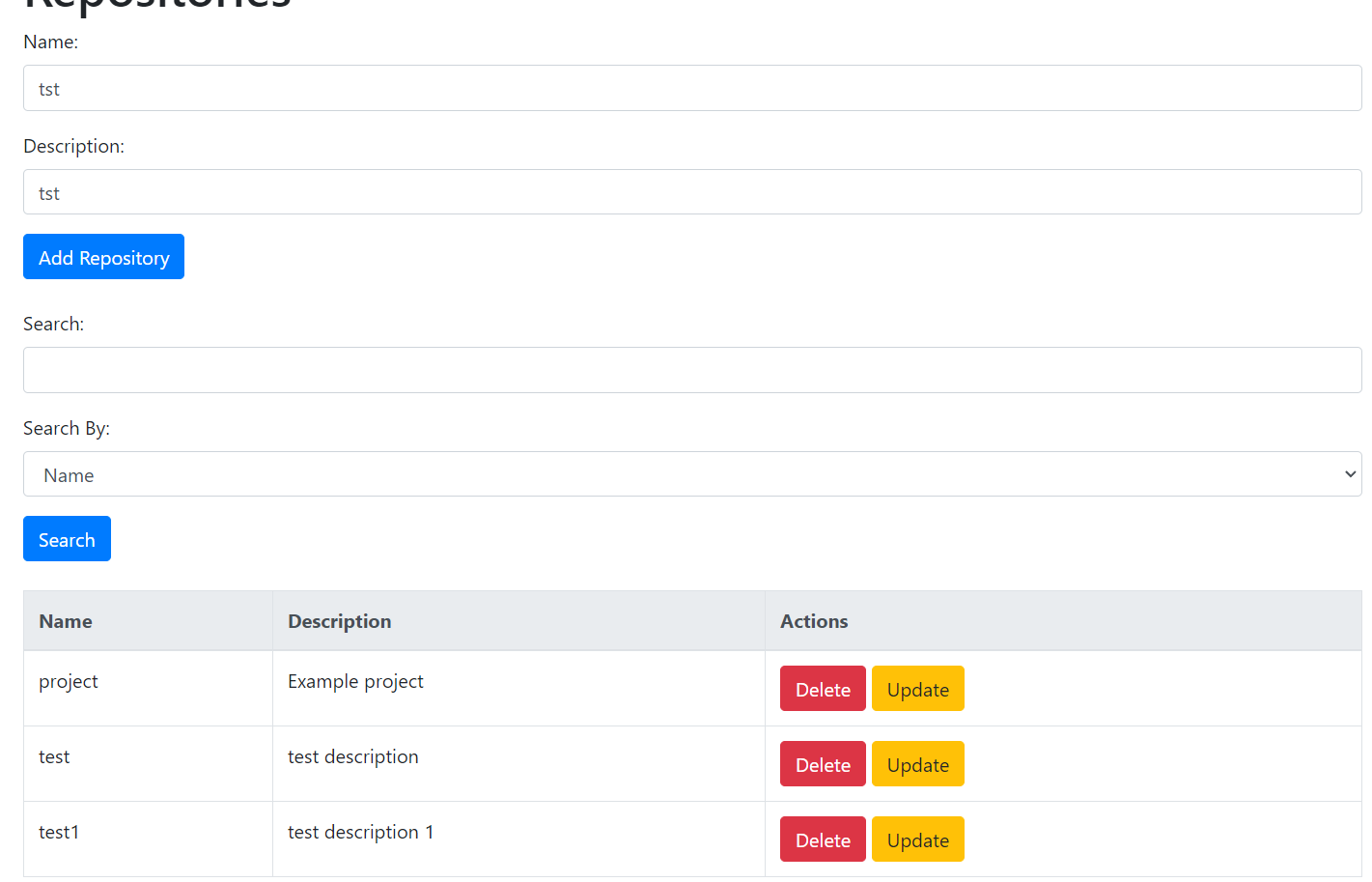


Рисунок 4 – Форма добавления нового репозитория

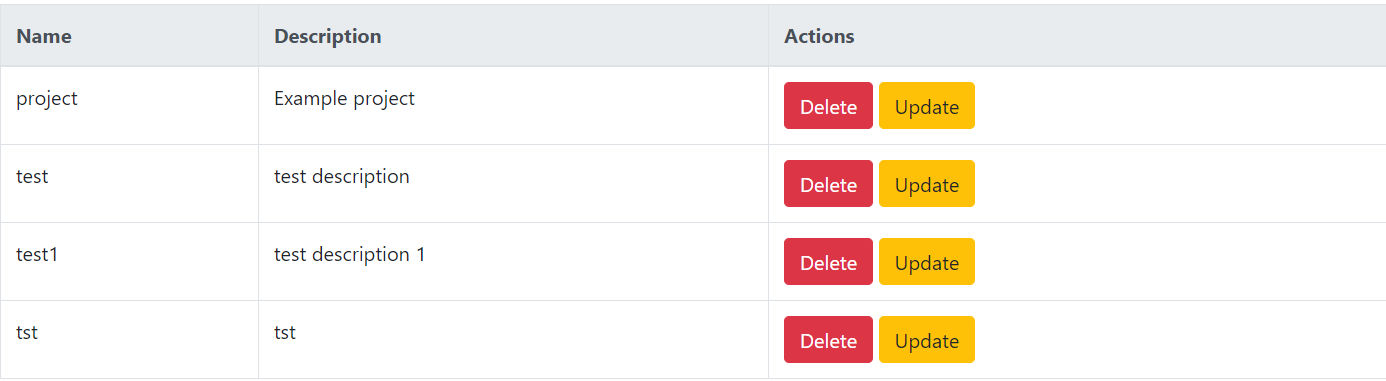


Рисунок 5 – Результат добавления

Реализован функционал поиска по некоторым атрибутам сущности. Атрибут для поиска выбирается из выпадающего списка. Например, для репозиториев доступен поиск по атрибутам *Name* и *Description*. Пример результата поиска по имени репозитория приведен на рисунке 6.

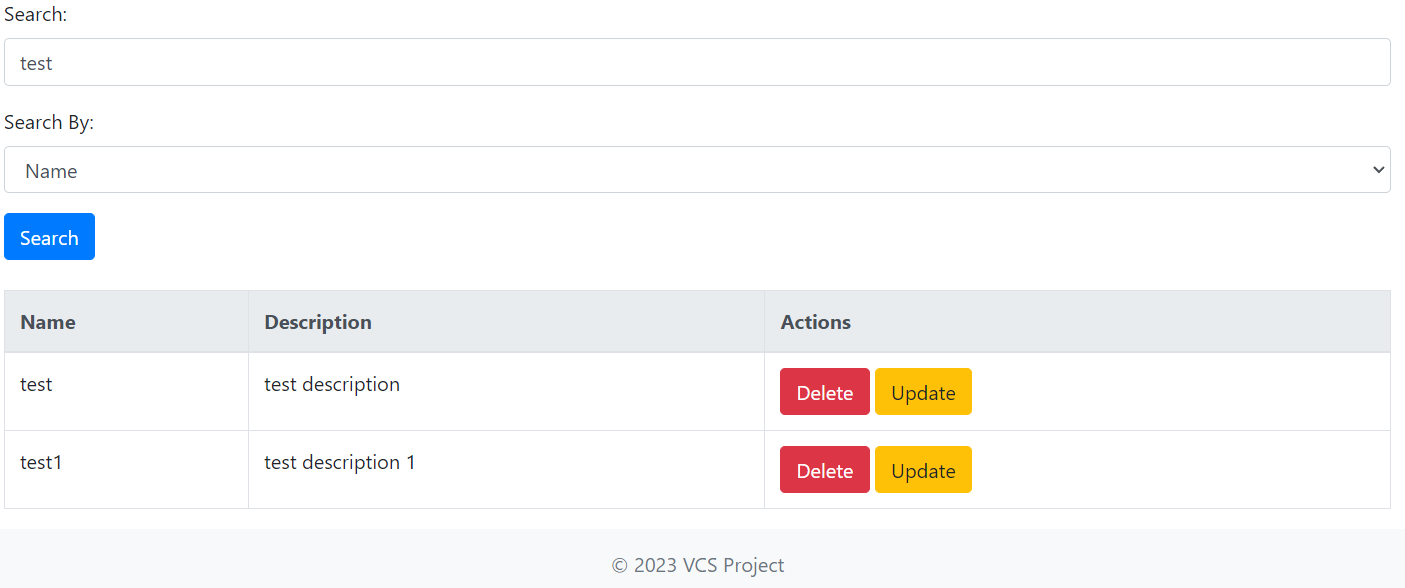


Рисунок 6 – Пример результата поиска

Реализованы удаление и изменение записей. При нажатии на кнопку «*Update*» появляется модальное окно, в котором пользователь вводит обновленные данные. Пример приведен на рисунке 7.

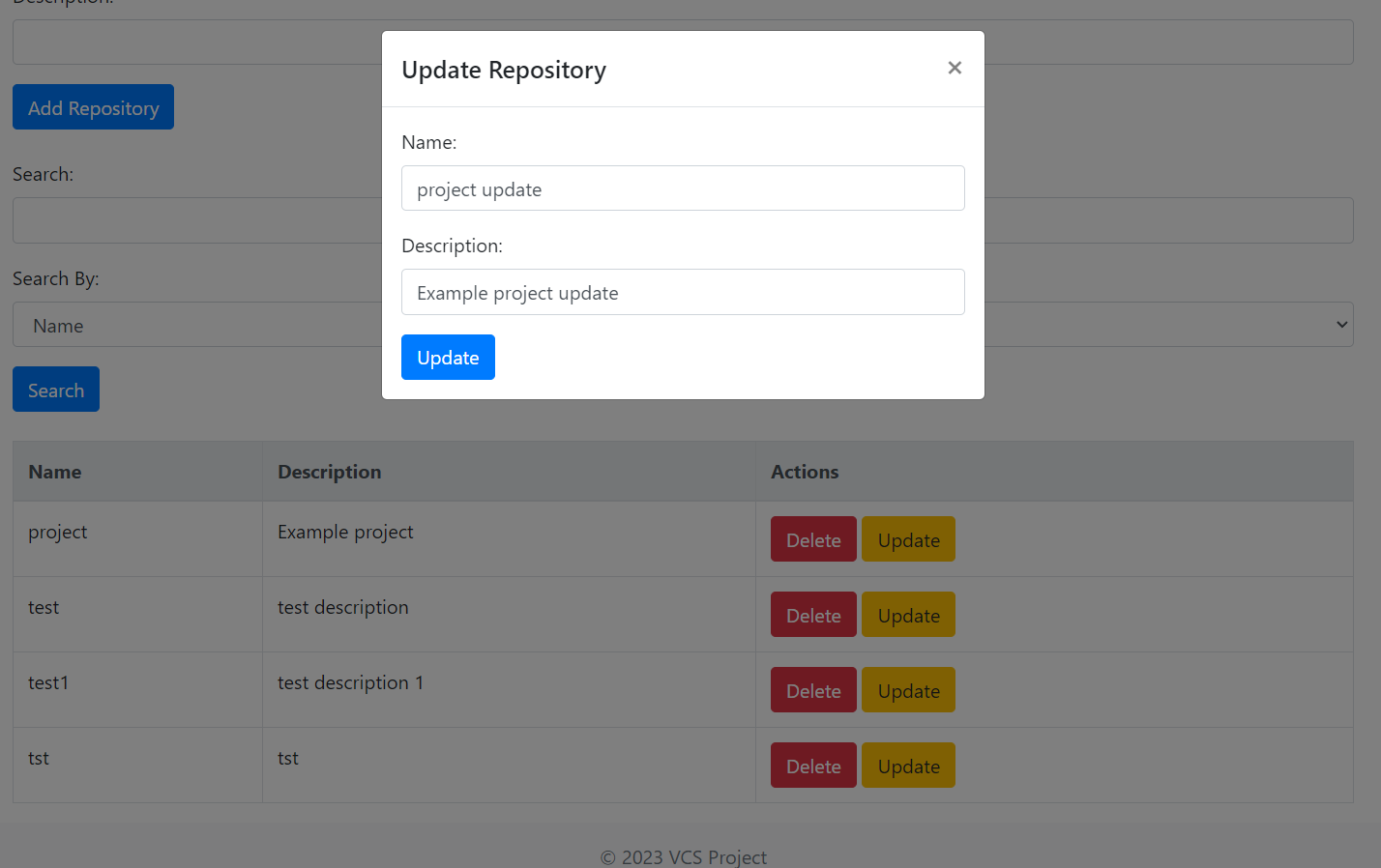


Рисунок 7 – Пример изменения записи

При нажатии на кнопку «Удалить» соответствующая запись будет удалена из таблицы в базе данных.

Пользователь перед тем, как начать работу с БД должен авторизоваться или зарегистрироваться, если у пользователя нету аккаунта. Имя пользователя уникально. Адрес электронной почты содержит символ «@» и «.». Пароль не менее 8 символов с содержанием специальных символов, цифр, верхнего и нижнего регистра. Изображения страниц регистрации и авторизации на рисунках 8, 9.



Рисунок 8 – Страница регистрации пользователя



Рисунок 9 – Страница авторизации пользователя

Если пользователь введет неправильные почту или пароль, то будет выведено сообщение, изображенное на рисунке 10.

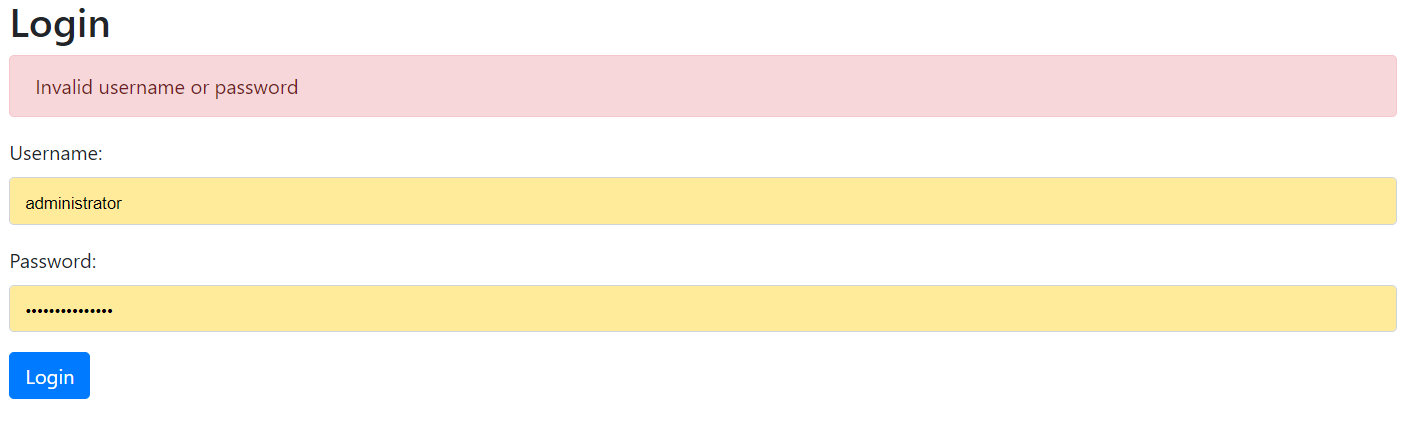


Рисунок 10 – Ошибка авторизации пользователя

При попытке ввода некорректных данных в форму регистрации, пользователю будет выведено сообщение о некорректных данных. Изображение с ошибкой показано на рисунке 11.

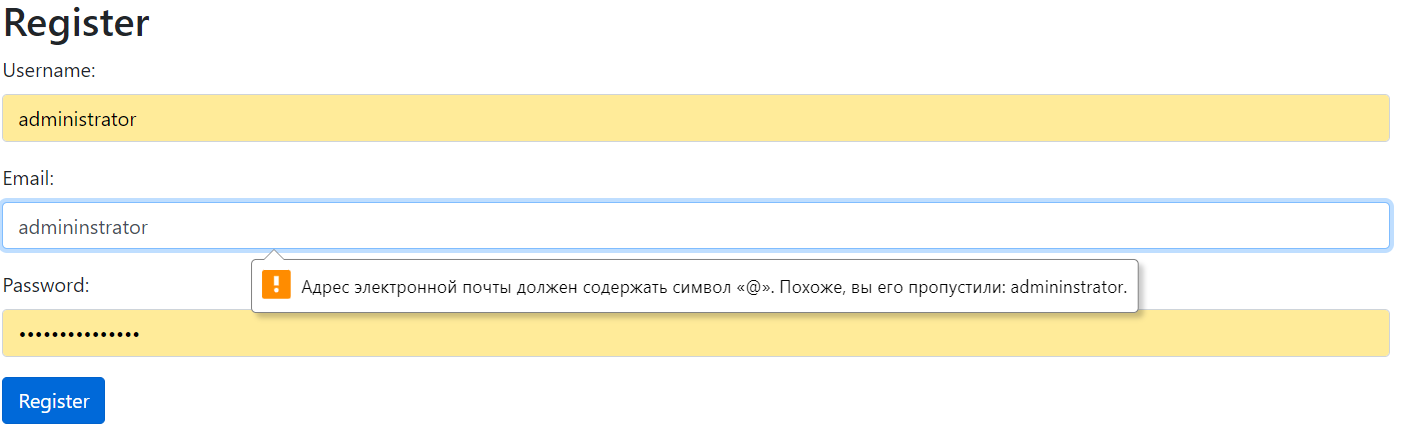


Рисунок 11 – Обработка ввода пользователя

# 

# **ЗАКЛЮЧЕНИЕ**

В процессе выполнения лабораторной работы изучен синтаксис языка программирования *PHP*, исследованы различные способы интеграции *PHP* с системой управления базами данных *MySQL*. Освоены методы подключения к базе данных, выполнение *SQL*-запросов для управления данными.

В результате выполнения работы удалось достичь глубокого понимания основ работы *PHP* и *MySQL*. Примененные знания позволили разработать функциональный веб-сайт, который эффективно взаимодействует с базой данных. Веб-сайт способен выполнять различные *SQL*-запросы, обеспечивая динамическое управление данными и интерактивное взаимодействие с пользователями.