|  |  |  |
| --- | --- | --- |
|  |  |  |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |

Институт Информационных технологий

Кафедра Инструментального и прикладного программного обеспечения

**ПРАКТИЧЕСКАЯ РАБОТА № 2**

по дисциплине «Разработка серверных частей интернет-ресурсов»

**Тема практической работы:** Работа с CRUD-операциями на языке PHP

|  |  |
| --- | --- |
| Студент группы ИКБО-32-21 | Быченков А.К. |

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись студента)

|  |  |
| --- | --- |
| **Руководитель практической работы** | преподаватель Волков М.Ю. |

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись руководителя)

Работа представлена «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2023 г.

Допущен к работе «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2023 г.

Москва 2023

**Цель работы:** получить навыки работы с базой данных и создания CRUD-операций на языке PHP.

**Задание**

Вам необходимо создать конфигурацию docker-compose для вашего веб-сервера.

Docker контейнер веб сервера должен включать:

* ОС Linux
* Веб-сервер Apache

Docker контейнер базы данных должен включать:

* ОС Linux
* Базу данных Mysql

Docker-compose:

* Docker-контейнеры, созданные ранее
* Примонтированные тома
* Настройки портов
* Связь между контейнерами
* Переменные внешнего окружения
* Настройки базы данных

1. Создать sql скрипт инициализации, где студенту необходимо реализовать любую одну сущность (например товары или студенты).
2. Реализовать связь между базой данных и веб-сервером.

Требуется создать 4 страницы php, где на каждой странице будет реализована одна из CRUD операций с выбранной вами сущностью.

**Ход работы**

Был написан sql-скрипт инициализации для создания сущности товара:

Листинг 1 – init.sql

CREATE DATABASE IF NOT EXISTS appDB;

CREATE USER IF NOT EXISTS 'user'@'%' IDENTIFIED BY 'password';

GRANT SELECT,UPDATE,INSERT ON appDB.\* TO 'user'@'%';

FLUSH PRIVILEGES;

USE appDB;

CREATE TABLE IF NOT EXISTS good (

id INT(11) NOT NULL AUTO\_INCREMENT,

title VARCHAR(63) NOT NULL,

description TEXT NOT NULL,

price DECIMAL(10,2) NOT NULL,

PRIMARY KEY (id)

);

INSERT INTO good (title, description, price) VALUES

('Jeans', 'Blue jeans', 100.00),

('T-shirt', 'White t-shirt', 50.00),

('Sweater', 'Red sweater', 150.00),

('Shoes', 'Black shoes', 200.00),

('Jacket', 'Green jacket', 250.00),

('Hat', 'Yellow hat', 50.00),

('Gloves', 'Black gloves', 50.00),

('Scarf', 'Red scarf', 50.00),

('Socks', 'White socks', 50.00),

('Underwear', 'Black underwear', 50.00)

Для отображения страниц был создан php-файл шаблона страницы:

Листинг 2 – layout.php

<html lang='ru'>

<head>

<title><?= $title; ?></title>

<link rel='stylesheet' href='style.css' type='text/css' />

</head>

<body>

<?php

include 'component/navigation.php';

echo navigation();

?>

<h1><?= $title; ?></h1>

<?= $content; ?>

</body>

</html>

Взаимодействие с базой данных реализовано в php-файле репозитория:

Листинг 3 – goodsRepository.php

<?php

include 'repository/db.php';

function getAllGoods() {

global $mysqli;

$result = $mysqli->query("SELECT \* FROM good");

return $result;

}

function getGoodById($id) {

global $mysqli;

$result = $mysqli->query("SELECT \* FROM good WHERE id='$id'");

return $result;

}

function createGood($title, $description, $price) {

global $mysqli;

$mysqli->query("INSERT INTO good (title, description, price) VALUES ('$title', '$description', '$price')");

return $mysqli->affected\_rows > 0;

}

function updateGood($id, $title, $description, $price) {

global $mysqli;

$mysqli->query("UPDATE good SET title='$title', description='$description', price='$price' WHERE id='$id'");

return $mysqli->affected\_rows > 0;

}

function deleteGood($id) {

global $mysqli;

$mysqli->query("DELETE FROM good WHERE id='$id'");

return $mysqli->affected\_rows > 0;

}

Шаблон отображения товаров:

Листинг 4 – goodsTable.php

<?php

function goodsTable($goods)

{

$table = '<table><tr><th>Идентификатор</th><th>Название</th><th>Цена</th></tr>';

foreach ($goods as $good) {

$table .= "<tr><td>{$good['id']}</td><td>{$good['title']}</td><td>{$good['price']}</td></tr>";

}

return $table . '</table>';

}

Страница просмотра товаров:

Листинг 5 – index.php

<?php

include 'repository/goodsRepository.php';

include 'component/goodsTable.php';

$title = 'Список товаров';

$content = goodsTable(getAll());

include 'templates/layout.php';

Страница просмотра одного товара:

Листинг 6 – readOne.php

<?php

include 'repository/goodsRepository.php';

include 'component/goodsTable.php';

$title = 'Найти товар';

$content = '<form action="readOne.php" method="post">

<label>Идентификатор</label><input type="number" name="id" required><br>

<input type="submit" value="Найти">';

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

$result = getById($\_POST['id']);

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

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

$content .= goodsTable($result);

} else {

$content .= "<p>Товар не найден</p>";

}

}

include 'templates/layout.php';

Страница редактирования товаров:

Листинг 7 – update.php

<?php

include 'repository/goodsRepository.php';

include 'component/goodsTable.php';

$title = 'Редактировать товар';

$content = '<form action="update.php" method="post">

<label>Идентификатор</label><input type="number" name="id" required><br>

<label>Название</label><input type="text" name="title" required><br>

<label>Описание</label><input type="text" name="description" required><br>

<label>Цена</label><input type="number" name="price" required><br>

<input type="submit" value="Обновить">';

if (isset($\_POST['id']) && isset($\_POST['title']) && isset($\_POST['description']) && isset($\_POST['price'])) {

$updated = updateById($\_POST['id'], $\_POST['title'], $\_POST['description'], $\_POST['price']);

if ($updated)

$content .= "<p>Товар обновлен</p>";

else

$content .= "<p>Товар не обновлен</p>";

}

include 'templates/layout.php';

Страница удаления товаров:

Листинг 8 – delete.php

<?php

include 'repository/goodsRepository.php';

include 'component/goodsTable.php';

$title = 'Удалить товар';

$content = '<form action="delete.php" method="post">

<label>Идентификатор</label><input type="number" name="id" required><br>

<input type="submit" value="Найти">';

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

$deleted = deleteById($\_POST['id']);

$content .= $deleted ? "<p>Товар удален</p>" : "<p>Товар не найден</p>";

}

include 'templates/layout.php';

**Вывод**

В ходе выполнения практической работы были получены навыки работы с базой данных и создания CRUD-операций на языке PHP.

**Ответы на вопросы**

1. **В чём преимущество языка PHP для backend’a?**

Преимущества языка PHP для backend’a:

* 1. Работа с базами данных: PHP имеет широкую поддержку для множества систем управления базами данных (например, MySQL, PostgreSQL, SQLite), что делает его отличным выбором для создания веб-приложений с базами данных.
  2. Создание API: PHP может использоваться для создания веб-сервисов и API, что делает его полезным для разработки клиент-серверных приложений.
  3. Универсальность и гибкость PHP. Он совместим с основными и самыми известными платформами: Unix, Linux, Windows и MacOS; поддерживается большинством веб-серверов (Apache, Netscape, Microsoft IIS, NGINX, iPlanet, Xitami, Caudium и Tornado) и баз данных (Mysql, PostgreSQL, MongoDB и другие). Из-за данных плюсов PHP его часто выбирают для разработки кроссплатформенных приложений: это помогает использовать максимум существующей инфраструктуры с оптимальным бюджетом.

1. **Как достигается тонкий клиент при использовании PHP?**

Тонкий клиент при использовании PHP достигается за счёт:

* 1. Разработки серверной логики: основная логика вашего приложения должна быть реализована на сервере с использованием PHP. Это включает в себя обработку запросов, взаимодействие с базой данных, вычисления и любую другую бизнес-логику.
  2. Создания API: для обеспечения взаимодействия между сервером и клиентом, создайте API (Application Programming Interface) с использованием PHP. API будет предоставлять доступ к данным и функциональности сервера через HTTP запросы (часто используется JSON для передачи данных).

1. **Что такое SSR?**

SSR (анг. аббревиатура от Server Side Rendering – рендеринг на стороне сервера) – содержимое веб-страницы рендерится на сервере и клиенту отправляются уже готовая веб-страница с данными.

**Ссылка на удаленный репозиторий проекта**

<https://github.com/descenty/server-side-5-sem>

**Список использованной литературы**

1. Официальная документация PHP по работе с MYSQL: <https://www.php.net/manual/ru/book.mysql.php>
2. Статья про использование MySQL в PHP: <https://htmlacademy.ru/blog/php/mysql>