|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | | МИНОБРНАУКИ РОССИИ | | | | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА** – **Российский технологический университет»**  **РТУ МИРЭА** | | | | |
| Институт информационных технологий | |
| Кафедра Инструментального и прикладного программного обеспечения (ИиППО) | |

|  |  |  |  |
| --- | --- | --- | --- |
| **ПРАКТИЧЕСКАЯ РАБОТА №5** | | | |
| по дисциплине «Разработка серверных частей интернет-ресурсов» | | | |
| Сессии. Файлы. БД. | | | |
| **Студент группы ИКБО-16-19** | | Мурадов Н.Н.  ­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись студента) | |
|  | |  | |
| Руководитель практической работы | | Преподаватель Волков М.Ю.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | |
| Работа представлена | «\_\_\_»\_\_\_\_\_\_\_2021 г. | | (подпись руководителя) |
| Допущен к работе | «\_\_\_»\_\_\_\_\_\_\_2021 г. | |  |
|  |  | |  |

Москва 2021

Ход работы

[Задание 2](file:///C:\Users\AramK\OneDrive\Документы\Учеба\РСЧИР\Практика%205\RSChIR_5.docx#_Toc84426618)

[Упражнение 2](file:///C:\Users\AramK\OneDrive\Документы\Учеба\РСЧИР\Практика%205\RSChIR_5.docx#_Toc84426619)

[Вывод: 2](file:///C:\Users\AramK\OneDrive\Документы\Учеба\РСЧИР\Практика%205\RSChIR_5.docx#_Toc84426620)

[Вопросы: 2](file:///C:\Users\AramK\OneDrive\Документы\Учеба\РСЧИР\Практика%205\RSChIR_5.docx#_Toc84426621)

[Список использованной литературы: 2](file:///C:\Users\AramK\OneDrive\Документы\Учеба\РСЧИР\Практика%205\RSChIR_5.docx#_Toc84426622)

## Задание

Предполагается выполнить апгрейд разрабатываемого в процессе первых 4 практических работ интернет-ресурса механизмами обработки сессий и согласования контента. Предлагается добавить следующую функциональность: Загрузка файлов в формате pdf на сервер и хранение их в файловой системе сервера, а также их выдача обратно пользователю по запросу, сохранение информации о сессии в Redis, а также выдача пользователю персонализированного контента на основе файлов cookie. Предполагается создание стабильной версии интернет-ресурса и сохранение предыдущей функциональности с практических работ 1-4.

Спецификация

Вход происходит на страничке index.php. Сразу слева снизу можно заметить «тумблеры» для переключения темы странички и языка. Также по всей работе сохранена возможность переключения темы. Появилась возможность пользоваться файлами cookie при регистрации/авторизации/изменении пользователя. Сразу после авторизации появляется возможность загрузить свой pdf файл. Также, во время работы пришлось подружить php и js.

## Упражнение

Код /src/php/start.php

<?php

$cookieExpire = time()+(60\*60\*24);

if(isset($\_GET['reset']) && $\_GET['reset'] == true)

{

if(isset($\_GET['log']) && isset($\_GET['pas']) && isset($\_GET['ch']))

{

setcookie('login', $\_GET['log'], $cookieExpire, '/');

setcookie('pass', $\_GET['pas'], $cookieExpire, '/');

setcookie('ch', $\_GET['ch'], $cookieExpire, '/');

echo $\_GET['log'];

return;

}

}

if(isset($\_GET['log']))

{

if(isset($\_GET['pas']))

{

if(!$\_COOKIE['login'] && !$\_COOKIE['pass'] && !$\_COOKIE['ch'])

{

setcookie('login', $\_GET['log'], $cookieExpire, '/');

setcookie('pass', $\_GET['pas'], $cookieExpire, '/');

setcookie('ch', $\_GET['ch'], $cookieExpire, '/');

echo $\_GET['log'];

return;

}

if($\_GET['log'] == $\_COOKIE['login'] && $\_GET['pas'] == $\_COOKIE['pass'])

{

echo 1;

return;

} else {

echo 0;

return;

}

}

if(!$\_GET['log'])

{

echo $\_COOKIE['login'];

return;

}

}

if(isset($\_GET['ch']))

{

if(!$\_GET['ch'])

{

echo $\_COOKIE['ch'];

return;

}

}

if(isset($\_GET['theme']))

{

if(!$\_GET['theme'])

{

echo $\_COOKIE['theme'];

return;

}else{

setcookie('theme', $\_GET['theme'], $cookieExpire, '/');

echo $\_GET['theme'];

return;

}

}

if(isset($\_GET['en']))

{

if(!$\_GET['en'])

{

echo $\_COOKIE['en'];

return;

}else{

setcookie('en', $\_GET['en'], $cookieExpire, '/');

echo $\_GET['en'];

return;

}

}

Код /src/uploadfile.php

<html lang="en">

<meta charset="UTF-8">

<body>

<?php

try {

switch ($\_FILES['upfile']['error']) {

case UPLOAD\_ERR\_OK:

break;

case UPLOAD\_ERR\_NO\_FILE:

throw new RuntimeException('No file sent.');

case UPLOAD\_ERR\_INI\_SIZE:

case UPLOAD\_ERR\_FORM\_SIZE:

throw new RuntimeException('Exceeded filesize limit.');

default:

throw new RuntimeException('Unknown errors.');

}

$finfo = new finfo(FILEINFO\_MIME\_TYPE);

if (false === $ext = array\_search(

$finfo->file($\_FILES['upfile']['tmp\_name']),

array(

'pdf' => 'application/pdf'

),

true

)) {

throw new RuntimeException('Invalid file format.');

}

$hash = hash\_file("sha512", $\_FILES['upfile']['tmp\_name']);

$clientRedis = new Redis();

$clientRedis->connect("redis", 6379);

if ($clientRedis->get($hash)) {

echo "<p>File already exists.</p>";

echo '<meta http-equiv="refresh" content="2;url=getfile.php?q='.$hash.'">';

die();

}

$clientRedis->set($hash, file\_get\_contents($\_FILES['upfile']['tmp\_name']));

echo 'File is uploaded successfully.';

echo '<meta http-equiv="refresh" content="2;url=getfile.php?q='.$hash.'">';

die();

} catch (RuntimeException $e) {

echo $e->getMessage();

}

?>

</body>

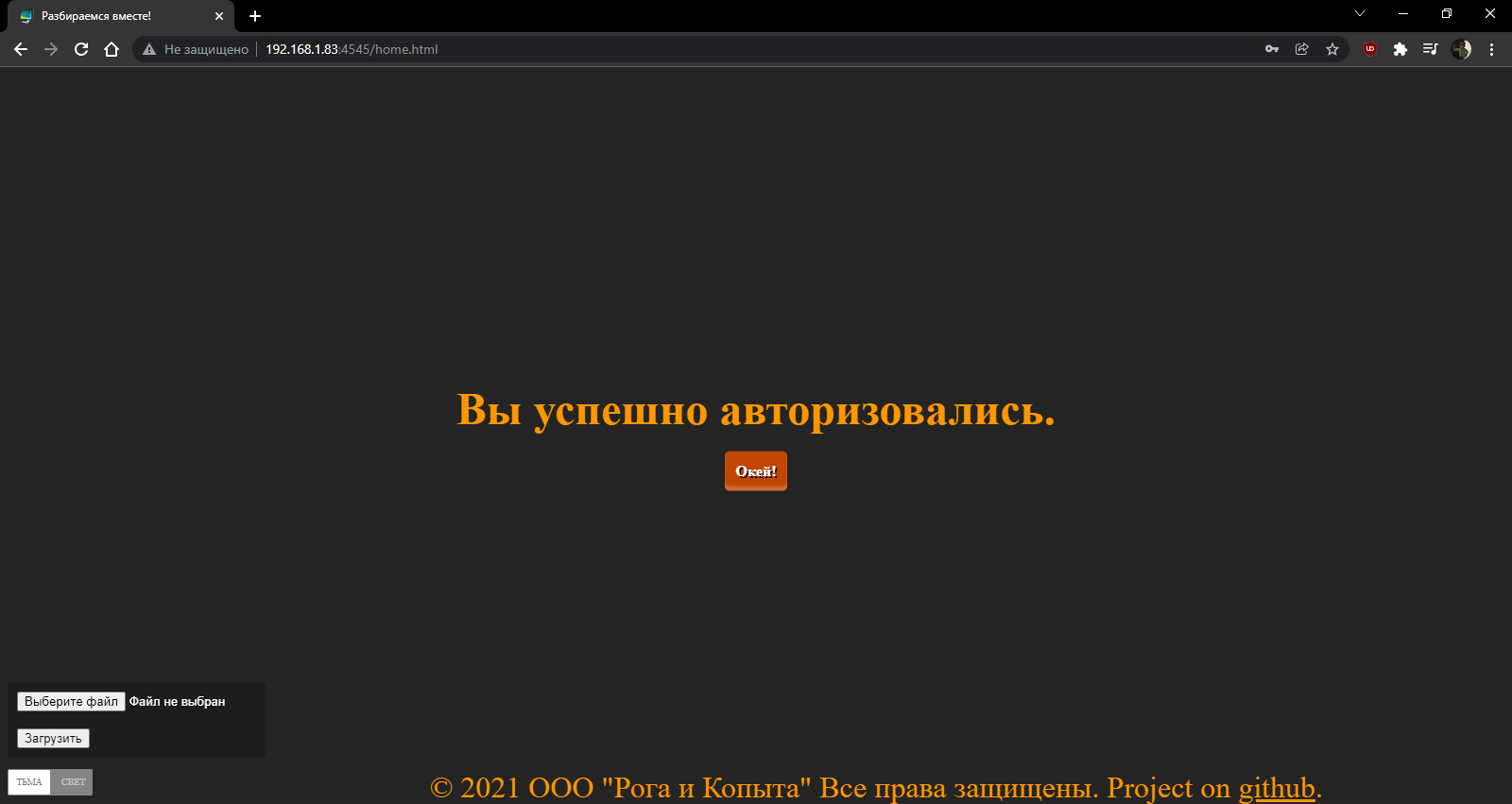


Рис.1 - поле для загрузки файла.

Код .htaccess

Allow from all

php\_value session.save\_handler "redis"

php\_value session.save\_path "tcp://redis:6379"

Код start.js

var vxod, reg, r, v, warnc, logv, logr, pasv, pasr, warnev, warner, warnew, regb, vxodb, ch1, ch2, ch3, wt, butr, butv, warncr, posform, tlang, ttheme, over;

window.onload = ini;

function ini(event){

vxod = document.getElementById("vxod");

reg = document.getElementById("reg");

r = document.getElementById("r");

v = document.getElementById("v");

warnc = document.getElementById("warnc");

warncr = document.getElementById("warncr");

warnev = document.getElementById("warnev");

warner = document.getElementById("warner");

warnew = document.getElementById("warnew");

logv = document.getElementById("logv");

logr = document.getElementById("logr");

pasv = document.getElementById("pasv");

pasr = document.getElementById("pasr");

ch1 = document.getElementById("ch1");

ch2 = document.getElementById("ch2");

ch3 = document.getElementById("ch3");

wt = document.getElementById("wt");

butr = document.getElementById("butr");

butv = document.getElementById("butv");

tlang = document.getElementById("tlang");

ttheme = document.getElementById("ttheme");

posform = document.getElementById("posform");

over = document.getElementsByClassName("over");

window.addEventListener('keydown', checkCaps);

logv.addEventListener('input', inpchv);

logr.addEventListener('input', inpchr);

pasv.addEventListener('input', inpchv);

pasr.addEventListener('input', inpchr);

tlang.addEventListener('change', chlang);

ttheme.addEventListener('change', chtheme);

regb = false;

vxodb = false;

var xhr = new XMLHttpRequest();

xhr.open('GET', `/php/start.php?en`, true);

xhr.send(); // (1)

xhr.onreadystatechange = function() { // (3)

if (xhr.readyState != 4) return;

if (xhr.status == 200 && event != null) {

var bul = xhr.responseText != "true";

tlang.firstElementChild.checked = bul;

prlang(bul);

}

xhr.open('GET', `/php/start.php?theme`, true);

xhr.send(); // (1)

xhr.onreadystatechange = function() { // (3)

if (xhr.readyState != 4) return;

if (xhr.status == 200) {

var bul1 = xhr.responseText == "true";

ttheme.firstElementChild.checked = bul1;

prtheme(bul1);

}

}

}

}

function chlang(event){

console.log('еСТЬ КОНТАКТ');

var dat = event.target;

console.log(dat);

console.log(dat.checked);

prlang(dat.checked);

var xhr = new XMLHttpRequest();

xhr.open('GET', `/php/start.php?en=${!dat.checked}`, true);

xhr.send(); // (1)

xhr.onreadystatechange = function() { // (3)

if (xhr.readyState != 4) return;

if (xhr.status == 200) {

console.log(xhr.responseText);

}

}

}

function chtheme(event){

console.log('еСТЬ КОНТАКТ');

var dat = event.target;

console.log(dat);

prtheme(dat.checked);

var xhr = new XMLHttpRequest();

xhr.open('GET', `/php/start.php?theme=${dat.checked}`, true);

xhr.send(); // (1)

xhr.onreadystatechange = function() { // (3)

if (xhr.readyState != 4) return;

if (xhr.status == 200) {

console.log(xhr.responseText);

}

}

}

function prtheme(bul)

{

if(bul)

over[0].style.backgroundColor = "rgba(255,255,255, 0.3)";

else

over[0].style.backgroundColor = "rgba(0,0,0, 0.7)";

}

function prlang(bul)

{

if(bul) {

var star = document.getElementsByClassName("start")[0];

star.innerHTML = `

Хотите собрать себе новый ПК, но не знаете с чего начать?

<span><br>Мы вам поможем!</span>

`;

posform.innerHTML = `

<header class="help">

<span id="r">Нет аккаунта? <a class="helpa" onclick="onreg();">Регистрация!</a></span>

<span id="v">Есть аккаунт? <a class="helpa" onclick="onvxod();">Вход!</a></span>

</header>

<div class="vxod" id="vxod">

<div>

<input class="login" type="text" placeholder="Логин" id="logv" required pattern="^[a-zA-Z0-9]+$">

</div>

<div>

<input class="pass" type="password" placeholder="Пароль" id="pasv" required pattern="^[a-zA-Z0-9]+$">

</div>

<span class="warn" id="warnc"><img src="media/warning.png" class="warnimg">Включён Caps Lock!</span>

<div class="button" onclick="vxo();">ВОЙТИ!</div>

</div>

<div class="reg" id="reg">

<div class="logo">

<p>Выберите иконку для профиля:</p>

<input id="ch1" name="ico" type="radio" value="1" checked><img class="logoi" src="media/ls-icon1.png">

<input id="ch2" name="ico" type="radio" value="2"><img class="logoi" src="media/ls-icon2.png">

<input id="ch3" name="ico" type="radio" value="3"><img class="logoi" src="media/ls-icon3.png">

</div>

<div>

<input class="login" type="text" placeholder="Логин" id="logr" required pattern="^[a-zA-Z0-9]+$">

</div>

<div>

<input class="pass" type="password" placeholder="Пароль" id="pasr" required pattern="^[a-zA-Z0-9]+$">

<span class="rand" onclick="gen\_pas();"><img src="media/random.png" class="randimg">Случайный пароль</span>

</div>

<span class="warn" id="warncr"><img src="media/warning.png" class="warnimg">Включён Caps Lock!</span>

<div class="button" onclick="rego();">ЗАРЕГИСТРИРОВАТЬСЯ!</div>

</div>

`;

warnev.innerHTML = '<img src="media/warning.png" class="warnimg">Внимание! <p>Допустимы только латинница и цифры</p>';

warner.innerHTML = '<img src="media/warning.png" class="warnimg">Внимание! <p id="wt">Допустимы только латинница и цифры</p>';

warnew.innerHTML = '<img src="media/warning.png" class="warnimg">Внимание! <p>Неверный логин или пароль</p>';

var fut = document.getElementById("fut");

fut.innerHTML = `

<div id="fut" style="position: fixed; right: 12.5vw; font-size: 2vw; color: #ff9700; bottom: 0; z-index: -1;">

© 2021 ООО "Рога и Копыта" Все права защищены. Project on <a href="https://github.com/Ddudde/Kursach-HTML" style="color: #ff9700;">github</a>.

</div>

`;

ttheme.innerHTML = `

<input type="checkbox">

<span class="checkbox-green-switch" data-label-on="Свет" data-label-off="Тьма"></span>

`;

} else {

var star = document.getElementsByClassName("start")[0];

star.innerHTML = `

Do you want to build yourself a new PC, but don't know where to start?

<span><br>We will help you!</span>

`;

posform.innerHTML = `

<header class="help">

<span id="r">No account? <a class="helpa" onclick="onreg();">Registration!</a></span>

<span id="v">Have an account? <a class="helpa" onclick="onvxod();">Entrance!</a></span>

</header>

<div class="vxod" id="vxod">

<div>

<input class="login" type="text" placeholder="Login" id="logv" required pattern="^[a-zA-Z0-9]+$">

</div>

<div>

<input class="pass" type="password" placeholder="Password" id="pasv" required pattern="^[a-zA-Z0-9]+$">

</div>

<span class="warn" id="warnc"><img src="media/warning.png" class="warnimg">Caps Lock on!</span>

<div class="button" onclick="vxo();">ENTER!</div>

</div>

<div class="reg" id="reg">

<div class="logo">

<p>Select an icon for your profile:</p>

<input id="ch1" name="ico" type="radio" value="1" checked><img class="logoi" src="media/ls-icon1.png">

<input id="ch2" name="ico" type="radio" value="2"><img class="logoi" src="media/ls-icon2.png">

<input id="ch3" name="ico" type="radio" value="3"><img class="logoi" src="media/ls-icon3.png">

</div>

<div>

<input class="login" type="text" placeholder="Login" id="logr" required pattern="^[a-zA-Z0-9]+$">

</div>

<div>

<input class="pass" type="password" placeholder="Password" id="pasr" required pattern="^[a-zA-Z0-9]+$">

<span class="rand" onclick="gen\_pas();"><img src="media/random.png" class="randimg">Random pass</span>

</div>

<span class="warn" id="warncr"><img src="media/warning.png" class="warnimg">Caps Lock on!</span>

<div class="button" onclick="rego();">REGISTER!</div>

</div>

`;

warnev.innerHTML = '<img src="media/warning.png" class="warnimg">Warning! <p>Only Latin letters and numbers are allowed</p>';

warner.innerHTML = '<img src="media/warning.png" class="warnimg">Warning! <p id="wt">Only Latin letters and numbers are allowed</p>';

warnew.innerHTML = '<img src="media/warning.png" class="warnimg">Warning! <p>Invalid username or password</p>';

var fut = document.getElementById("fut");

fut.innerHTML = `

<div id="fut" style="position: fixed; right: 12.5vw; font-size: 2vw; color: #ff9700; bottom: 0; z-index: -1;">

© 2021 LLC "Рога и Копыта" All rights reserved. Project on <a href="https://github.com/Ddudde/Kursach-HTML" style="color: #ff9700;">github</a>.

</div>

`;

ttheme.innerHTML = `

<input type="checkbox">

<span class="checkbox-green-switch" data-label-on="Light" data-label-off="Dark"></span>

`;

}

ini(null);

}

function kon(){

window.scrollTo(0, window.innerHeight);

}

function gen\_pas(){

var password = "";

var symbols = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789";

for (var i = 0; i < 15; i++){

password += symbols.charAt(Math.floor(Math.random() \* symbols.length));

}

pasr.value = password;

//navigator.clipboard.writeText(password);

var text = tlang.firstElementChild.checked ? 'Сгенерирован пароль: ' : 'Password generated: ';

text += password;

text += tlang.firstElementChild.checked ? '. Он скопирован в буфер обмена' : '. It is copied to the clipboard'

wt.innerHTML = text;

warner.style.display = "inline";

setTimeout(function (){

warner.style.display = "none";

wt.innerHTML = tlang.firstElementChild.checked ? 'Допустимы только латинница и цифры' : 'Only Latin letters and numbers are allowed';

}, 10000);

}

function inpchv(event){

var dat = event.target;

if (dat.validity.patternMismatch || dat.value.length == 0) {

dat.style.animation = "but 1s ease infinite";

setTimeout(function () {dat.style.animation = 'none'}, 1000)

dat.style.outline = "solid red";

warnev.style.display = "inline-block";

} else {

dat.style.outline = "none black";

warnev.style.display = "none";

}

}

function vxo(){

var xhr = new XMLHttpRequest();

xhr.open('GET', `/php/start.php?log=${logv.value}&pas=${pasv.value}`, true);

xhr.send(); // (1)

xhr.onreadystatechange = function() { // (3)

if (xhr.readyState != 4) return;

if (xhr.status == 200) {

var bul = xhr.responseText == "1";

console.log(bul);

if(bul)

{

warnew.style.display = 'none';

console.log('vxod');

window.location.href = 'home.html';

}

else

warnew.style.display = 'inline';

}

}

}

function rego(){

if(regb && pasr.value && logr.value){

let ch = ch1.checked ? 1 : ch2.checked ? 2 : 3;

var xhr = new XMLHttpRequest();

xhr.open('GET', `/php/start.php?reset=true&log=${logr.value}&pas=${pasr.value}&ch=${ch}`, true);

xhr.send(); // (1)

xhr.onreadystatechange = function() { // (3)

if (xhr.readyState != 4) return;

if (xhr.status == 200) {

console.log(xhr.responseText);

}

}

onvxod();

}

}

function inpchr(event){

var dat = event.target;

regb = !(dat.validity.patternMismatch || dat.value.length == 0);

if (dat.validity.patternMismatch || dat.value.length == 0) {

dat.style.animation = "but 1s ease infinite";

setTimeout(function () {dat.style.animation = "none"}, 1000)

dat.style.outline = "solid red";

warner.style.display = "inline";

} else {

dat.style.outline = "none black";

warner.style.display = "none";

}

}

function checkCaps(event) {

var caps = event.getModifierState && event.getModifierState('CapsLock');

if (caps){

warnc.style.display = "inline";

warncr.style.display = "inline";

}else{

warnc.style.display = "none";

warncr.style.display = "none";

}

}

function onreg(){

console.log('sdffds');

vxod.style.transform = "rotateX(90deg)";

vxod.addEventListener('transitionend', chvxod);

}

function onvxod(){

reg.style.transform = "rotateX(90deg)";

reg.addEventListener('transitionend', chreg);

}

function chvxod(){

vxod.style.position = "absolute";

reg.style.transform = "rotateX(0deg)";

reg.style.position = "relative";

r.style.display = "none";

v.style.display = "inline";

vxod.removeEventListener('transitionend', chvxod);

}

function chreg(){

reg.style.position = "absolute";

vxod.style.transform = "rotateX(0deg)";

vxod.style.position = "relative";

r.style.display = "inline";

v.style.display = "none";

reg.removeEventListener('transitionend', chreg);

}

## Вывод:

Все цели практической работы были успешно достигнуты. Новый опыт получен, а также приобретены некоторые, дополнительные знания.

## Вопросы:

1. Сессии являются механизмом, который использует для отслеживания "состояния" между сайтом и каким-либо браузером. Сессии позволяют вам хранить произвольные данные браузера и получать их в тот момент, когда между данным браузером и сайтом устанавливается соединение. Данные получаются и сохраняются в сессии при помощи соответствующего "ключа".

2. это небольшой фрагмент данных, отправляемый сервером на браузер пользователя, который тот может сохранить и отсылать обратно с новым запросом к данному серверу. Это, в частности, позволяет узнать, с одного ли браузера пришли оба запроса (например, для аутентификации пользователя). Они запоминают информацию о состоянии для протокола HTTP, который сам по себе этого делать не умеет.

3. Перевод веб-страниц между сервером и браузером происходит посредством протокола передачи гипертекста (http). Когда пользователь вводит URL в адресную строку браузера, браузер берет ее и отправляет запрос на сервер, запрашивая веб-страницы, заданные пользователем. Далее, сервер посылает страницу, запрашиваемую браузером, в виде http-ответа. Ответ передается в виде пакетов из текста, который может содержать заявление с просьбой, чтобы браузер сохранил куки. Это делается посредством заявления, "настройка cookie: имя = значение". Браузер спрашивает, сохранить значение-строка 'имя' и вернуть его на сервер при каком-либо дальнейших к нему обращений. В любой последующий запрос к одному серверу, даже при запросе другой веб-страницы с использованием данного сервера, браузер отправляет серверу значение куки. Сервер идентифицирует эту информацию и выполняет запрос, без необходимости пользователю выполнять процесс аутентификации еще раз.

4. Опишите простой пример работы сессий в PHP.

Сессии являются простым способом хранения информации для отдельных пользователей с уникальным идентификатором сессии. Это может использоваться для сохранения состояния между запросами страниц. Идентификаторы сессий обычно отправляются браузеру через сессионный cookie и используются для получения имеющихся данных сессии. Отсутствие идентификатора сессии или сессионного cookie сообщает PHP о том, что необходимо создать новую сессию и сгенерировать новый идентификатор сессии.

5. Опишите способы защиты сессии пользователя.

По умолчанию вся информация о сессии, включая ID, передается в cookie. Но так бывает не всегда. Некоторые пользователи отключают cookie в своих браузерах. В таком случае браузер будет передавать идентификатор сессии в URL.

Здесь ID передается в открытом виде, в отличие от сессии через cookie, когда информация скрыта в HTTP-заголовке. Самым простым способом защиты от этого будет запрет передачи идентификатора сессии через адресную строку. Сделать это можно, прописав следующее в конфигурационном файле Apache-сервера .htaccess:

php\_flag session.use\_only\_cookies on

6. Верно ли, что можно хранить данные сессии в БД?

Сессия как правило - это необходимые горячие данные пользователя, что нужно сохранить между запросами. БД - одно из самых медленных хранилищ этих данных. В файлах кстати тоже не стоит хранить. При большой нагрузке io будет подтормаживать. Длительный период сессии обычно не хранятся, вместо этого на клиент задается токен, по которому человек через много времени может автоматически авторизироваться 7. Что такое Web API?

Это интерфейс прикладного программирования для веб-сервера или веб-браузера. Это концепция веб-разработки, обычно ограниченная клиентской стороной веб-приложения, и поэтому обычно не включает детали реализации веб-сервера или браузера, такие как SAPI или API, если они не доступны для общего доступа через удаленное веб-приложение.

7) Жизненный цикл сессии проходит несколько этапов :

1) Абсолютно для каждого нового запроса на сервер (неважно, разные это клиенты или один) ASP.NET генерирует уникальный идентификатор сессии. Идентификатор сессии представляет собой случайно сгенерированное число, закодированное с помощью специального алгоритма в строку длиной 24 символа. Строка состоит из литералов от A до Z в нижнем регистре, а также чисел от 0 до 5. Пример идентификатора - hjnyuijl1pam3vox2h5i41in

2)Если в течение текущего запроса данные клиента НЕ сохраняются для дальнейшей работы с ним, то и время жизни сессии этого клиента заканчивается (фактически не начавшись). При этом ранее сгенерированный идентификатор сессии становится недействительным (так как не был использован). В ответ на такой запрос клиент не получает ничего, чтобы связало его с новой сессией.

3)Если же данные клиента (например, имя, адрес доставки товара) сохраняются на сервере, ASP.NET связывает сохраненные данные с ранее сгенерированным идентификатором сессии. Далее создается специальная сессионная куки, и в нее записывается также этот идентификатор. Эта куки добавляется в ответ на запрос и сохраняется в браузере клиента. Таким образом, создается связь клиента и его персонализированной информации на сервере. Новая сессия для данного клиента создана.

4)При каждом следующем запросе клиент передает на сервер персональный идентификатор сессии через куки. Сервер сопоставляет идентификаторы и «узнает» клиента в рамках текущей сессии.

5)До тех пор пока клиент передает свой персональный ключ, сессия считается активной. Сессия может закончиться по разным причинам, например, вручную на стороне сервера или по истечении какого-то установленного времени (таймаут).

8. Обработка сессии это ключевой приём в PHP, что позволяет хранить данные пользователя на всех страницах веб-сайта или приложения, так что нет.

9. Система управления сессиями поддерживает ряд опций, которые могут быть указаны в файле php.ini. Ниже приводится краткий обзор.

session.save\_handler string

session.save\_handler определяет имя обработчика, который используется для хранения и извлечения данных, связанных с сессией. По умолчанию имеет значение files. Следует обратить внимание, что некоторые модули могут зарегистрировать собственные обработчики (save\_handler). Текущие зарегистрированные обработчики отображаются в phpinfo(). Смотрите также session\_set\_save\_handler().

session.save\_path string

session.save\_path определяет аргумент, который передаётся в обработчик сохранения. При установленном по умолчанию обработчике files, аргумент содержит путь, где будут создаваться файлы. Смотрите также session\_save\_path().

У этой директивы также существует дополнительный аргумент N, определяющий глубину размещения файлов сессии относительно указанной директории. Например, указание '5;/tmp' может в конечном итоге привести к такому размещению файла сессии: /tmp/4/b/1/e/3/sess\_4b1e384ad74619bd212e236e52a5a174If . Для того, чтобы использовать аргумент N, необходимо предварительно создать все эти директории. Помочь в этом может небольшой скрипт, расположенный в ext/session. Версия для bash называется mod\_files.sh, а Windows-версия - mod\_files.bat. Также следует учитывать, что если N определён и больше 0, то автоматическая сборка мусора не выполняется, подробнее смотрите информацию в файле php.ini. Кроме того, если используется N, необходимо удостовериться, что значение session.save\_path указано в кавычках, поскольку разделитель (;) в php.ini используется как знак комментария.

Модуль хранения файлов создаёт файлы с правами 600 по умолчанию. Это можно изменить с помощью необязательного аргумента MODE: N;MODE;/path, где MODE - восьмеричное представление режима доступа к файлу. Установка MODE не затрагивает umask.

10 Опишите директивы конфигурации файловой системы и потоков в PHP

Имя Значение по умолчанию Область изменения

allow\_url\_fopen "1" PHP\_INI\_SYSTEM

user\_agent NULL PHP\_INI\_ALL

default\_socket\_timeout "60" PHP\_INI\_ALL

from NULL ??

auto\_detect\_line\_endings "Off" PHP\_INI\_ALL

allow\_url\_fopen boolean

Данная директива включает поддержку упаковщиков URL (URL wrappers), которые позволяют работать с объектами URL, как с обычными файлами. Упаковщики, доступные по умолчанию, служат для работы с удаленными файлами с использованием протокола ftp или http. Некоторые расширения, например, zlib, могут регистрировать собственные упаковщики.

ser\_agent string

Устанавливает строку "User-Agent" для использования ее PHP при запросах к удаленным серверам.

default\_socket\_timeout integer

Значение таймаута (в секундах) для потоков, использующих сокеты.

Замечание: Данная директива стала доступна с версии PHP 4.3.0

from="joe@example.com" string

Устанавливает пароль для анонимного доступа к серверу ftp (ваш адрес электронной почты).

auto\_detect\_line\_endings boolean

Когда данная директива включена, PHP проверяет данные, получаемые функциями fgets() и file() с тем, чтобы определить способ завершения строк (Unix, MS-Dos или Macintosh).

Данная директива позволяет PHP взаимодействовать с системами Macintosh, однако, по умолчанию эта директива выключена, поскольку при ее использовании возникает (несущественная) потребность в дополнительных ресурсах для определения символа окончания первой строки, а также потому, что программисты, использующие в системах Unix символы перевода строки в качестве разделителей, столкнутся с обратно-несовместимым поведением PHP.

11 Какой тип ресурса использует файловая система. Опишите данный тип

Файловая система. На каждом носителе информации (гибком, жестком или лазерном диске) может храниться большое количество файлов. Порядок хранения файлов на диске определяется используемой файловой системой.

Каждый диск разбивается на две области: область хранения файлов и каталог. Каталог содержит имя файла и указание на начало его размещения на диске. Если провести аналогию диска с книгой, то область хранения файлов соответствует ее содержанию, а каталог - оглавлению. Причем книга состоит из страниц, а диск - из секторов.

Для дисков с небольшим количеством файлов (до нескольких десятков) может использоваться одноуровневая файловая система, когда каталог (оглавление диска) представляет собой линейную последовательность имен файлов. Такой каталог можно сравнить с оглавлением детской книжки, которое содержит только названия отдельных рассказов.

Если на диске хранятся сотни и тысячи файлов, то для удобства поиска используется многоуровневая иерархическая файловая система, которая имеет древовидную структуру. Такую иерархическую систему можно сравнить, например, с оглавлением учебника, которое представляет собой иерархическую систему разделов, глав, параграфов и пунктов.

Начальный, корневой каталог содержит вложенные каталоги 1-го уровня, в свою очередь, каждый из последних может содержать вложенные

каталоги 2-го уровня и так далее. Необходимо отметить, что в каталогах всех уровней могут храниться и файлы.

12 Как открыть и закрыть файл с помощью PHP

Для открытия файла используется функция fopen().

Ее синтаксис: int fopen(string filename, string mode [, int use\_include\_path])

Принимаемые аргументы:

string filename – имя файла или абсолютный путь к нему. Если путь к файлу не будет указан, то будет произведен его поиск в текущем каталоге. При отсутствии искомого файла система выведет сообщение об ошибке.

string mode – указывает режим открытия файла. Принимаемые аргументом значения:

r – файл открыт только для чтения, файловый указатель устанавливается в начале;

r+ – файл открыт для чтения и записи;

w – создается новый файл только для записи. Если файл с таким именем уже существует, в нем происходит автоматическое удаление всех данных;

w+ — создается новый файл для записи и чтения. При существовании такого файла происходит полная перезапись его данных на новые;

a – файл открыт для записи. Указатель устанавливается в конце. То есть запись в файл php начнется не с начала, а с конца;

a+ – открытие файла в режиме чтения и записи. Запись начнется с конца;

b – режим работы с файлом, содержащим в себе двоичные данные (в двоичной системе исчисления). Этот режим доступен только в операционной системе Windows.

Для закрытия доступа к файлу служит функция fclose ().

Ее синтаксис: int fclose (int file), где int file – дескриптор сайта, который нужно закрыть.

После каждого чтения или записи файл нужно закрывать этой функцией. Иначе остается открытым поток, созданный для файла. А это ведет к лишнему расходу серверных мощностей.

13. Как производится чтение и запись файлов в PHP

//Открытие тестового файла $file = fopen('test.txt', 'wt');

//Запись строки в файл fwrite($file, 'Текущая дата и время: '. date('d.m.y H:i:s')); //Закрытие файла fclose($file).

14. Опишите как считать только часть файла, как считывать файл последовательно и считать весь файл целиком.

Если нам надо прочитать файл полностью, то мы можем облегчить себе жизнь, применив функцию **file\_get\_contents()**:

<?php

$str = htmlentities(file\_get\_contents("form.php"));

echo $str;

?>

Также можно провести поблочное считывание, то есть считывать определенное количество байт из файла с помощью функции **fread()**:

<?php

$fd = fopen("form.php", 'r') or die("не удалось открыть файл");

while(!feof($fd))

{

    $str = htmlentities(fread($fd, 600));

    echo $str;

}

fclose($fd);

?>

15. Как производится создание и удаление файлов с помощью PHP

Для создания и открытия файла на PHP используют функцию fopen():

|  |  |
| --- | --- |
| 1 | fopen(имя\_файла, режим\_файла); |

- имя\_файла – здесь нужно указать название и расширение файла, которое нужно создать или открыть. Например, «bloggood-ru.txt».  
- режим\_файла – здесь нужно указать режим, другими словами параметры. Например, что вы хотите сделать с этим файлом: дописать текст или вставить новый и т.д

Для **удаления** **файла** в **PHP** предусмотрена функция unlink(f). Она принимает единственный строковой параметр, который указывает директорию и имя **файла**, который следует **удалить**. Пример: if (unlink(«D:\documents\copy.txt»)) echo «**Файл** успешно **удален**»; elseecho «Ошибка!

16. С помощью каких функций и какую информацию о файле можно получить с помощью PHP?

fstat — Получает информацию о файле, используя открытый файловый указатель. fsync — Синхронизирует изменения в файле (включая метаданные). ftell — Возвращает текущую позицию указателя чтения/записи файла.

17. Что такое DOM?

DOM означает объектную модель документа. Это программный интерфейс, который позволяет нам создавать, изменять или удалять элементы из документа. Мы также можем добавлять события к этим элементам, чтобы сделать нашу страницу более динамичной. Модель DOM рассматривает документ HTML как дерево узлов.

18. Как создать документ и работать с ним с помощью модуля DOM?

Можно представить HTML как набор вложенных коробок. Теги вроде <body> и </body> включают в себя другие теги, которые в свою очередь включают теги, или текст. Структура данных, использующаяся браузером для представления документа, отражает его форму. Для каждой коробки есть объект, с которым мы можем взаимодействовать и узнавать про него разные данные – какой тег он представляет, какие коробки и текст содержит. Это представление называется Document Object Model (объектная модель документа), или сокращённо DOM.  
  
Мы можем получить доступ к этим объектам через глобальную переменную document. Её свойство documentElement ссылается на объект, представляющий тег . Он также предоставляет свойства head и body, в которых содержатся объекты для соответствующих элементов.

19. Что такое JSON?

JSON (JavaScript Object Notation) — это формат для хранения и обмена информацией, доступной для чтения человеком. Файл содержит только текст и использует расширение. json.

20. Как декодировать строку JSON и вернуть JSON-представление данных?

Метод JSONDecoder.raw\_decode() декодирует JSON-документ из строки s в формате JSON и возвращает двойной кортеж представление данной строки в Python и индекс в строки s, где документ закончился. Метод JSONDecoder.raw\_decode() может быть использован для декодирования документа JSON из строки, которая в конце содержит посторонние данные.

21. Как проанализировать и выявить ошибки при кодировании и декодировании JSON?

В некоторых случаях вам может не потребоваться Codable для поддержки двунаправленного кодирования и декодирования.  
Например, некоторым приложениям требуется только вызывать удаленные сетевые API, и им не нужно декодировать ответы, содержащие тот же тип.  
Encodable Если вам нужна только поддержка кодирования данных Encode, заявите о соответствии. И наоборот, Decodable заявляет о соответствии, если вам нужно только читать данные определенного типа.

22)  
  
можно создать XML-документ и считывать его функцией simplexml\_load\_file().  
simplexml\_load\_file — Интерпретирует XML-файл в объект  
  
simplexml\_load\_file(  
string $filename,  
?string $class\_name = SimpleXMLElement::class,  
int $options = 0,  
string $namespace\_or\_prefix = "",  
bool $is\_prefix = false  
): SimpleXMLElement|false  
  
Преобразует правильно сформированный XML-документ в указанном файле в объект.  
  
23)  
  
драйверы на основе субд используются с такими источниками данных, как Oracle или SQL Server, которые предоставляют автономное ядро субд для использования драйвером. Эти драйверы обращаются к физическим данным через автономный модуль. то есть они отправляют SQLные инструкции и получают результаты из подсистемы.  
  
  
24)  
  
SELECT x,y,z FROM table1  
  
UPDATE table1 SET x = 'b' WHERE x = 'a'

25. Постоянное HTTP-соединение — использование одного TCP-соединения для отправки и получения множественных HTTP-запросов и ответов вместо открытия нового соединения для каждой пары запрос-ответ.

Идея постоянных подключений состоит в том, чтобы соединение между клиентским процессом и базой данных можно было использовать повторно, особенно когда требуется создавать и закрывать соединения множество раз. Это бы позволило снизить накладные расходы на создание новых подключений каждый раз, когда они требуются, за счёт использования существующих кешированных подключений, свободных для повторного использования.

В модуле есть встроенный функционал, осуществляющий очистку соединений и переводящий их в состояние пригодное для использования. Код очистки, реализованный в mysqli включает следующие операции:

· Откат активных транзакций

· Закрытие и удаление временных таблиц

· Снятие блокировки с таблиц

· Сброс переменных сессии

· Закрытие подготовленных запросов (всегда происходит в PHP)

· Закрытие обработчиков

· Снятие блокировок, установленных функцией GET\_LOCK()

Модуль mysqli делает очистку соединений автоматически путём вызова C-API функции mysql\_change\_user().

26. MongoDB — это ориентированная на документы база данных NoSQL с открытым исходным кодом, которая использует для хранения структуру JSON. Модель данных MongoDB позволяет представлять иерархические отношения, проще хранить массивы и другие более сложные структуры.

MongoDB отлично подходит для:

· кэширование данных;

· электронная коммерция, каталоги товаров, соцсети, новостные форумы и другие похожие сценарии, где много контента, в том числе видео и изображений;

· новый проект или стартап, если неизвестна итоговая структура данных или же вы точно знаете, что у вас будут слабо связанные данные без четкой схемы хранения;

· геоаналитика (обработка геопространственных данных — данных на основе местоположения);

· хранение данных с датчиков и устройств, собранных с решений интернета вещей, в том числе промышленных;

· работа с большими данными в машинном обучении;

· исследования в ритейле и других отраслях.

27. Процесс добавления новой записи в СУБД MongoBD

Все данные хранятся в бд в формате BSON, который близок к JSON, поэтому нам надо также вводить данные в этом формате. При добавлении в нее данных она автоматически создается.

Для добавления в коллекцию могут использоваться три ее метода:

· insertOne(): добавляет один документ

Все поля в документе представляют из себя набор пар ключ, значение.

Для хранения публикаций коллекция будет содержать поля title, description, rate, languages. Таким образом title, description, rate, languages - будут являться ключами.

Добавим публикацию с заголовком, описанием, рейтингом и списками языков, относящимися к данной публикации:

db.posts.insertOne({" title": "Заголовок публикации", "description": "Какое то описание публикации", rate: 20, languages: ["Английский", "Русский", "Итальянский"]})

· insertMany(): добавляет несколько документов

Добавим еще 2 демонстративные публикации.

Для этого передадим в insertMany массив, в котором перечислены добавляемые записи через запятую.

db.posts.insertMany([{" title": "Заголовок второй публикации", "description": "Описание второй публикации", rate: 1, languages: ["Английский", "Португальский", "Итальянский"]},{" title": "Заголовок третьей публикации", "description": "Описание третьей публикации", rate: 22, languages: ["Русский"]}])

· insert(): может добавлять как один, так и несколько документов.

Добавление одной записи:

b.posts.insert({" title": "Заголовок для insert метода", "description": "Описание для публикации с insert методом", rate: 20, languages: ["Английский", "Русский", "Португальский", "Польский"]})

Добавление нескольких записей:

db.posts.insert([{" title": "Заголовок для insert метода", "description": "Описание для публикации с insert методом", rate: 44, languages: ["Английский", "Русский", "Португальский", "Польский"]},{" title": "Новый заголовок для insert метода", "description": "Новое описание для публикации с insert методом", rate: 30, languages: ["Французский", "Немецкий"]}])

## Список использованной литературы:

1. Конспект лекций по дисциплине «Разработка сервисных частей интернет ресурсов», РТУ МИРЭА, лектор –преподаватель Куликов А.А.

2. Docker – URL: https://www.docker.com/ (Дата обращения – 02.09.2021)

3. Настройка стека ламп в контейнерах Docker– URL: https://www.centlinux.com/2020/03/configure-lamp-stack-in-docker-containers.html (Дата обращения – 02.09.2021)

4. Полное практическое руководство Docker – URL: https://habr.com/ru/post/310460/ (Дата обращения – 02.09.2021)