1. feladat: IMPROTÁLT ADATBÁTZISHOZ USERNAME, PASSWORD:

localhost/phpmyadmin-ban import és adatbázis kiválasztása ott a navigációs sávon Jogok -> username jelszó kismásolása a connect.php alapján (mivel csatlakozzunk fel?) utána: ’Helyi gép’ kiválasztása nagyon fontos !!! (Közben gyakran frissítsük a weblapot eltűnnek-e a hibák ☺)

1. feladat:

PARAMÉTEREK MEGADÁSA:

FONTOS A SORREND: szerver címe, adatbázis neve, username, password, port !!!

Kiszeded külön változókba és azokat adod át a mysqli konstruktornak:

$host = 'localhost';

$username = 'esti\_shop\_user';

$password = 'esti\_shop\_user';

$database = 'esti\_shop';

$port = 3306;

KAPCSOLAT BEÁLLÍTÁSA:

$con = new mysqli($host, $username, $password, $database, $port);

LEHETSÉGES HIBÁK KEZELÉSE:

if ($con->errno) {

die("Hiba a kapcsolat létrehozásakor!");

}

if (!$con->set\_charset("utf8")) {

echo "Nem sikerült beállítani a karakterkódolást!";

}

1. feladat: ???
2. feladat:

(Részlet az esti\_shop0401 mappában lévő projekt html mappájának reg\_form.html)

<form> </form> tag-ek jelölik a szabvány HTML űrlapot minden esetben.

action="#" kinek küldünk a form-ról adatot

method="post" milyen http szabvánnyal küldjük?

egy tárolóba <div> egy darab <input> kerüljön annak a class-a legyen „form-group” (ez Bootstrap)

A divben legyen egy input, annak a Bootstrap class-a: „form-control”

inputok fajtái: (<input type=””>) NINCS ZÁRÓTAG-JE

„text” (szöveges mező)

„number” -> számos, csúszkás izé

„email” -> HTML vizsgálja alapból van-e @ karakter benne

„password” -> pöttyök

bármelyikhez required attribútomot beállítassz annak azinputnak a KÖTELEZŐ!

„submit” -> ő az aki a formból (ibnputokból) kiszedi az adatokat és küldi akinek megadtad, hogy küldöd (action attr.) és mivel (POST vagy GET értelem szerűen a method attr.ban!)

placeholder=””: jelzed a dfelhasználónak mit írjon m,elyik inputba (szürkés szöveg megjelenik az inputban)

ugyanez ha a type=”submit” akkor value=”IDE ÍRJ”, type=”submit”-nál a placeholder nem működik mert nem kitölthető mező a submit input ahnem GOMB!

ellenőrzi a postot:

akinek küldöm ott:

if ($\_SERVER['REQUEST\_METHOD'] == "POST") { … } // minden post-tal jön-e???

de ha akarom, akkor egyesével is ellenőrízhetem:

// formon van egy <input type=’text’ name=’username’ required>

input, ha a type text akkor: kitölthető szövegmező

name: ezzel a névvel keresi a szerveroldal a kóüldött adatot

required: kötelezően kitöltendő, amíg nincs beleírva valami, addig szinkronosan nem submitolja a submit gomb a formot (alapból a html5 ellenőrzi ki van-e töltve)

if (isset($\_POST[’NAME attr. értékét’]))

tehát:

if (isset($\_POST[’username’]))

ha pl. a formon van egy <input type=’password’ name=’pwd’ required>

ha a type=password akkor kipontozott lesz az írás (nem látszik mit adsz jelszónak)

if (isset($\_POST[’pwd’]))

aszinkronnál:

$.ajax({

method: ’AMIVEL KÜLDÖM’ (GET vagy POST),

url: ’AKINEKKULDOM.php’,

data:{

kismacska: itt van a jelszó értéke, pl $(’[name=pwd]).val() de ezt kiszedhetem változóba is

pl. let jelszo = $(’[name=pwd]).val();

akkor

kismacska: jelszo

}

});

if (isset($\_POST[’kismacska’]))

szinkron küldés: <form action=’akinekkudldom.php’ method=’amivel küldöm’>

action -> szerveroldali szkript aki várja, akinek ugye küldöm

method pedig amivel (GET/POST)

aszinkronnál van JS, ahol az ajax kérésben is megadhatod:

pl.

$.ajax({

method: ’AMIVEL KÜLDÖM’ (GET vagy POST),

url: ’AKINEKKULDOM.php’

});

sql lekérdezést elkészítette:

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

if (!empty($\_POST['email']) && (!empty($\_POST['password']))){

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

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

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

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

1. paraméterezett: (prepared statement)

$sql = "INSERT INTO user (email, password, name, address) VALUES (?, ?, ?, ?)";

$stmt = $con -> prepare($sql);

$stmt -> bind\_param("ssss", $email, $pwd, $name, $address);

if ($stmt -> execute()){

header("Location: index.php");

} else {

$\_SESSION['error'] = "Helytelen adatok!";

}

$stmt -> close();

$con -> close();

1. nem paraméterezett (create statement)

$sql = "INSERT INTO user (email, password, name, address) VALUES (’$email’, ’$pwd’, ’$name’, ’$address’)";

$res = $con -> query($sql);

if($res){

header("Location: index.php"); // ha sikeres akkor header() elírásnyít a szerveroldal vhova máshova

} else {

session\_start();

$\_SESSION['error'] = "Helytelen adatok!"; // user figyelmeztetés… (a reg formon akkor legyen kiírva a tartalma – ha létezik)

}

$con -> close();

}

}

másik példa az érdemes művészeknél:

if (isset($\_POST['felhasznalo'])) { // POST VIZSGÁLAT

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

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

$sql = "SELECT az, nev FROM szemely WHERE nev = ? AND ev = ?";

$stmt = $link->prepare($sql);

$stmt->bind\_param('si', $felhasznalo, $jelszo);

$stmt->execute();

$stmt->store\_result();

if ($stmt->num\_rows == 1) {

//sikeres bejelentkezés

$stmt->bind\_result($id, $nev);

$stmt->fetch();

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

$\_SESSION['nevaz'] = $id;

} else {

$\_SESSION['hiba'] = "Helytelen felhasználónév vagy jelszó!";

}

SELECT-OPTION GENERÁLÁS:

require\_once '../config/connect.php';

$sql = "select distinct ev from szemely order by ev";

$res = $link->query($sql);

$html = '<select><option>Kérjük válasszon!</option>';

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

$html .= "<option>{$row['ev']}</option>";

}

$html .= '</select>';

echo $html;

printMessage() ????

ez valszeg Báró függvénxe, config/functions.php állományból hyívható függvény

ahol használni akarom, akkor require\_once ’../config/functions.php’; és utána hívhatom, hogy

printMessage(„üzenet”); // valszeg vmivel paraméterezhető

echo file\_get\_contents(’erőforrás neve’);

file\_get\_contents(’elérési út’);

egyik tartalomba behúzza a másikat és ha kiechozom akkor ki is írja a tartalmát

kiprinteli egy behúzott erőforrás tartalmát

1. feladat:
2. JQUERY-vel írj mindent…. $.ajax() ez jqueryben definiált metódus…

$.ajax({

method: ’AMIVEL KÜLDÖM’ (GET vagy POST),

url: ’AKINEK KULDOM.php’,

data:{ // adatok küldése – kulcs-érték páros

kismacska: itt van a jelszó értéke, pl $(’[name=pwd]).val() de ezt kiszedhetem változóba is

pl. let jelszo = $(’[name=pwd]).val();

akkor

kismacska: jelszo,

success: function(ha várok vissza valamit, akkor ide dekl. egy paramétert és lényegében elfogtam a választ…)

// itt meg a válasszal azt kezdek amit akarok… pl. alert(valasz);

$(’valahova’).html(valasz) // ha html-t kapok vissza azt a html() függvénnyel a domfának a kijelölt részébe gyógyíthatom… ’valahova’ a kijelölt elem….

// ha van header a szerveroldalon, akkor itt legyen:

location.href =’ahova a header visz szerveren, ide kell írni’;

},

error: function(xhr){

alert(xhr.status);

}

});

AJAXOS login:

$(document).on('click', '#submit', function (submitGomb) {

submitGomb.preventDefault(); // megállítottam a szerver felé futást,

// még itt pár sor JS-nek le kell fusson...

let username = $('[name=felhasznalo]').val().trim(); // leszedi az üres entereket

let jelszo = $('[name=jelszo]').val().trim(); // leszedi az üres entereket

// kliensoldali ellenőrzés:

// 1. példa: 6 karakternél több nem lehet a jelszó:

if(jelszo.length >6){

return; // ha a jelszó 6 hossznál hosszabb akkor kilépek az egész eseményből – sose fut az ajax kérés sorhoz ott lent:

}

regex:

tegyük fel van email cím mező és kiszedtük az értékét:

let email = $(’[name=email]’).val();

if(!email.test(EMAIL-REGEX)){

return;

}

$.ajax({

method: 'post',

url: 'php/login.php',

data: {

email: email,

jelszo: jelszo

},

success: function () {

location.href = 'main.php';

},

error: function () {

alert("Nem sikerült a bejelentkezés");

}

})

});

1. feladat:

AJAX kérés, lásd a fentieket

POST-ot használ: method: ’POST’;

felhasználót értesítette:

success: function(){

alert(’siker’);

töltödjön újra az oldal: location.reload();

}

error: function(xhr){

alert(xhr);

}

Bootstrap class-ok:

<input type=”text”>

<input type=”submit” class=”btn btn-primary”> ez egy gomb, ennek érdemes class=”btn btn-prmiary”

Sima szövegmező (ha a type = ”text vagy password vagy number vagy email”)

* form-control (100% hossz-t ad a szövegmezőnek + kékes árnyék + szebb beviteli mező)

Gombnál (<button> vagy <input type=”submit”>)

* btn

+

* btn-primary (kék)
* btn-danger (piros)
* btn-warning (sárga)
* btn-info (kékesszürkés)
* btn-primary (kék szín)

+

* form-control (100% hossz-t ad a gombnak)

Táblázat:

* table (alap táblázat szerkezet)

+

* table-hover (ha egy adott rekordot kijelölsz, az szürkés lesz)
* table-striped (csíkos)

<table class=”table table-striped”>

vagy

<table class=”table table-hover user-table”>

ha a classban, class=”table table-striped” van space, azaz szóköz, akkor az azt jelenti, hogy két class van ráaaggatva

jQuery-ben nem érdemes a formázó (table, table-striped), hanem érdemes a „technikai” (user-table) class-al kiszedni az elemet:

$(document).on(’esemény neve’, ’.user-table’, function() {…});

minden input beletehető form-group-ok közé.

<div class=”form-group”>

<input type=”text” placeholder=”felhasználónév” required>

</div>

<div class=”form-group”>

<input type=”email” placeholder=”jelszó” required>

</div>

minden egyes input menjen egy „form-group” class-al rendelkező <div>-be.