

Záródolgozat

készítette: Becsei Szabolcs Gábor

2024

Tartalomjegyzék

1. Bevezetés	4
1.1 Köszönetnyilvánítás	4
1.2 Téma választás	4
1.3 Tapasztalatok	4
2. Fejlesztői dokumentáció	5
2.1 Programnyelv kiválasztása előkészületek	5
2.2 Futtatási környezet és teszt fel használók	6
2.3 Az oldal adatbázisa	7
1. Kapcsolatok	8
2. Felhasználó adminisztrálására szolgáló táblák	9
3. Megosztás és üzenetek eltárolásáért felelős táblák:	12
4. Fájlok feltöltésért felelős táblák:	14
2.4 Algoritmusok	15
1. Regisztráció	15
2. Bejelentkeztetés	18
3. Kijelentkeztetés	19
4. Email küldés	19
5. Kommunikációs oldalak	22
6. Fájl feltöltés	29
2.5 Teszt dokumentáció	32
2.6 Fejlesztési javaslatok	34

3. Felhasználói dokumentáció	35
3.1 Futtatási környezet.....	35
3.1 Program	35
3.2 Regisztráció	35
3.3 Bejelentkezés.....	36
3.4 Jelszó visszaállítás	37
3.5 Kapcsolat	37
3.6 Barát keresés.....	38
3.7 Megosztás.....	39
3.8 Chatelés.....	40
 4. Összefoglalás	 41
4.1 Munkám értékelése.....	41
4.2 Programom későbbi hasznosulása	41
 5. Irodalom jegyzék	 42

1. Bevezetés

1.1 Köszönetnyilvánítás

Ezúton szeretném megköszönni Englert Ervin és Kovács László Bálint tanár úrnak, akik nélkül ez a szakdolgozat nem jöhetett volna létre.

1.2 Téma választás

Ezt a témát azért választottam, mert úgy gondaltam, hogy tanulmányaim során az alap tudást megszereztem hozzá és érdekes tapasztalat lesz. A záródolgozat alapját az nyújtotta, hogy nem láttam szépen kivitelezett és használható oldalt, amely a lóverseny kedvelőinek szól, így hát felbátorodtam és elkezdtem.

1.3 Tapasztalatok

A tapasztalatom a záródolgozattal kapcsolatban az, hogy az ehhez hasonló oldalak sokkal egyszerűbbnek tűnnek hogyha csak a kliens oldaláról nézzük, majd ahogy jobban belelátunk a feladatba, akkor már nem is olyan egyszerű több szempontból sem. Rájövünk arra, hogy a javascript kódot összekötni a PHP kóddal, hogy stílusosan nézzen ki, nem is mindig olyan egyszerű. Miközben írtam ezt a csodát, rájöttem, hogy sokszor nem mindig azt a legcélszerűbb alkalmazni amit az iskolában elsajátítottunk, mert előfordul, hogy inkább csak hátráltat és sokszor nem elegáns csak a legegyszerűbb megoldást alkalmazni egy feladat teljesítésére. További tapasztalatom, hogy amikor fejlesztünk, érdemes a legkorszerűbb elérhető eszközöket használni, mert sok időt megtakaríthatunk vele.

2. Fejlesztői dokumentáció

2.1 Programnyelv kiválasztása előkészületek

Amikor kiválasztottam a témát akkor viszonylag egyértelmű volt, mely nyelveket kell majd használnom és mely programot. Így hát Xampp és Visual studio code nevezetű programokat és PHP, HTML, CSS, JavaScript, MySQL nyelveket használtam, majd menet közben rájöttem, hogy vannak olyan problémák, amelyeket csak ezekkel nem tudok megoldani. Így találkoztam a jQuery-vel ami igaz, hogy a JavaScript csomagja, de szerintem megérdemli, hogy külön nyelvnek hívjuk. Sok esetben sokkal egyszerűbb és elegánsabb mintsem csak szimplán JavaScript-et alkalmazni, jelentősen csökkentette az elkészülési idejét a záródolgozatomnak, a weboldal elkészült, de még több lehetőség rejlik benne, mint például klasszifikáció, a chatszobák létrehozására.

2.2 Futtatási környezet és teszt fel használók

Böngésző:

Opera

Chrome

weboldal linkje: <https://lovifans.com>

email fiók link: <https://mail.websupport.hu/>

teszt felhasználók:

teszt1:

felhasználó: tesztsub1

jelszó: teszt1234

email fiók:

email cím: teszt1@lovifans.com

jelszó: Nq0QKLIICC

teszt2:

felhasználó: tesztsub1

jelszó: teszt1234

email fiók:

email cím: teszt2@lovifans.com

jelszó: Ph28Fq99II

2.3 Az oldal adatbázisa

Áttekintés

Az adatbázis egy szociális hálózatot modellez, amely lehetővé teszi a felhasználók közötti kapcsolattartást, tartalom megosztást és az interakciót. Az adatbázis különböző táblákat tartalmaz, amelyek leírják a felhasználók, csoportok, bejegyzések, kommentek, üzenetek és egyéb kapcsolódó entitások adatait.

Az adattáblák három részre oszthatóak:

1. Felhasználó adminisztrálására szolgáló táblák:

- users
- login
- note
- pass_reset
- friends

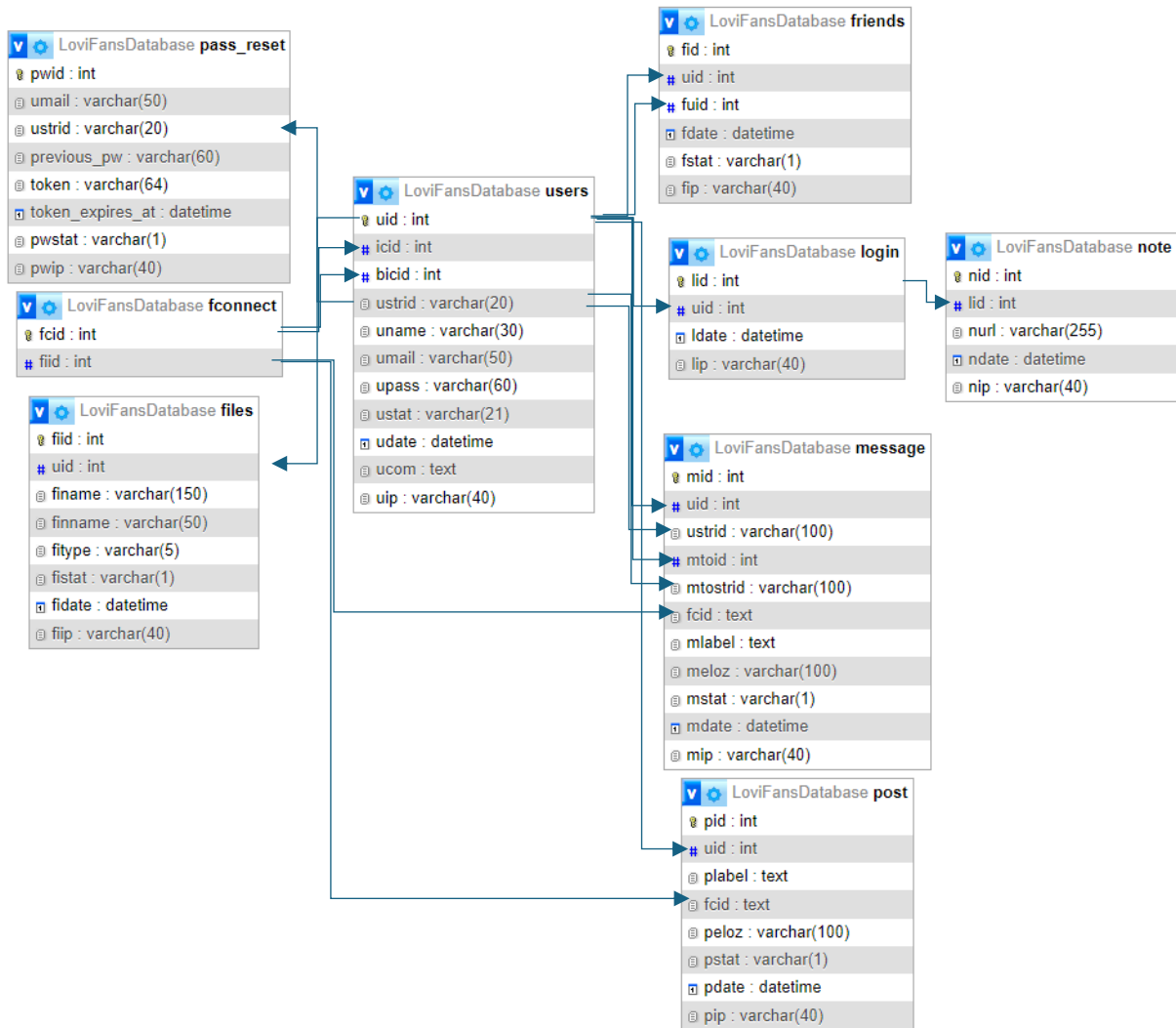
2. Megosztás és üzenetek eltárolásáért felelős táblák:

- message
- post

3. Képek feltöltésért felelős táblák:

- files
- fconnect

1. Kapcsolatok



2. Felhasználó adminisztrálására szolgáló táblák

users

A users adat tábla a felhasználók adminisztrálására készült, hogy minden szükséges mező definiálva legyen a felhasználók adatainak tárolására.

Az adattáblába több mezőt terveztem az esetleges későbbi bővítés érdekében.

leírás:

uid: Felhasználó azonosítója (Primary Key)

icid: Profilkép azonosítója

bicid: Borítókép azonosítója

ustrid: Felhasználó szöveges azonosítója

uname: Felhasználó neve

umail: Felhasználó email címe

upass: Felhasználó jelszava

ustat: Felhasználó státusza

amelyben eltároltam a fiók állapotát ideiglenesen, illetve, hogy mikor jelentkezett be utoljára, vagy éppen be van jelentkezve a fiókba vagy sem.

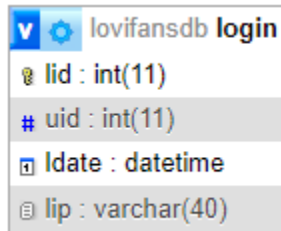
udate: Felhasználó regisztrációjának ideje

ucom: Felhasználó megjegyzése

uip: Felhasználó regisztrációjának IP címe

lovifansdb users	
🔑	uid : int(11)
#	icid : int(11)
#	bicid : int(11)
📄	ustrid : varchar(20)
📄	uname : varchar(30)
📄	umail : varchar(50)
📄	upass : varchar(60)
📄	ustat : varchar(21)
📅	udate : datetime
📄	ucom : text
📄	uip : varchar(40)

login



lovifansdb	login
🔑	lid : int(11)
#	uid : int(11)
📅	ldate : datetime
📄	lip : varchar(40)

A login adat tábla arra szolgál, hogy tároljam a felhasználó bejelentkezését. Ahogy utaltam már rá, még bővíteném azzal, hogy képes legyen tárolni a kijelentkezés idejét is, ezzel elő lehetne állítani egy statisztikát a felhasználókról be és kijelentkezéséről, hogy ki mennyit használja, illetve kiderüljön mennyire kedvelik magát az oldalt.

Leírás:

lid: Bejelentkezés azonosítója (Primary Key)

uid: Felhasználó azonosító

ldate: Bejelentkezés időpontja

lip: Bejelentkezés IP címe

note

A note adat tábla arra van, hogy naplózza azt, hogy mikor meddig melyik felhasználó melyik oldalra megy fel.

Leírás:

nid: Jegyzet azonosítója (Primary Key)

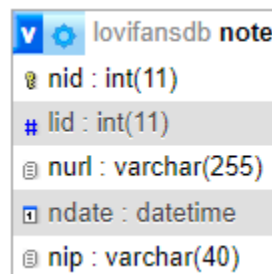
lid: Felhasználó azonosító

nurl: Jegyzet URL-je

ndate: Jegyzet létrehozásának ideje


nip: Jegyzet létrehozásának IP címe




mip: Üzenet létrehozásának IP címe



lovifansdb	note
🔑	nid : int(11)
#	lid : int(11)
📄	nurl : varchar(255)
📅	ndate : datetime
📄	nip : varchar(40)

pass_reset



lovifansdb pass_reset	
	pwid : int(11)
	umail : varchar(50)
	ustrid : varchar(20)
	previous_pw : varchar(60)
	token : varchar(64)
	token_expires_at : datetime
	pwstat : varchar(1)
	pwip : varchar(40)

Azzal a céllal jött létre ez a tábla, hogy ha valamelyik felhasználó esetleg elfelejtené a jelszavát, akkor eltárol egy egyszer használatos token egy dátumot és lejárat dátumot, mely segítségével új jelszót tud beállítani.

Leírás:

pwid: Jelszó visszaállítás azonosítója (Primary Key)

umail: Felhasználó email címe

ustrid: Felhasználó rövidített azonosítója

previous_pw: Előző jelszó

token: Visszaállító token

token_expires_at: Token lejárat ideje

pwstat: Jelszó visszaállítás státusza (aktív, használt)

pwip: Jelszó visszaállítás IP címe

3. Megosztás és üzenetek eltárolásáért felelős táblák:

post

A post publikus üzenet, amit bárki bármikor láthat, így hát a message táblával ellentétben csak egy uid-t kellett hozzárendelni.



lovifansdb post	
🔑	pid : int(11)
#	uid : int(11)
📄	plabel : text
📄	fcid : text
📄	peloz : varchar(100)
📄	pstat : varchar(1)
📅	pdate : datetime
📄	pip : varchar(40)

Leírás:

pid: Bejegyzés azonosítója (Primary Key)

uid: Felhasználó azonosító

plabel: Bejegyzés címkéje

fcid: Szülő bejegyzés azonosítója

peloz: Bejegyzés tartalma

pstat: Bejegyzés státusza (törölt, aktív, felfüggesztett)

pdate: Bejegyzés létrehozásának ideje

pip: Bejegyzés létrehozásának IP címe

message

A message tábla a privát kommunikáció létrehozására jött létre ezért két uid-t kapott.

Leírás:

mid: Üzenet azonosítója (Primary Key)

uid: Felhasználó azonosító

ustrid: Felhasználó szöveges azonosítója

mtoid: Címzett felhasználó azonosítója

mtostrid: Címzett felhasználó szöveges azonosítója

fcid: Szülő üzenet azonosítója

mlabel: Üzenet címkéje

meloz: Üzenet tartalma

mstat: Üzenet státusza (aktív, törölt, felfüggesztett)

mdate: Üzenet létrehozásának ideje

mip: Üzenet létrehozásának IP címe



lovifansdb message	
mid	int(11)
uid	int(11)
ustrid	varchar(100)
mtoid	int(11)
mtostrid	varchar(100)
fcid	text
mlabel	text
meloz	varchar(100)
mstat	varchar(1)
mdate	datetime
mip	varchar(40)

4. Fájlok feltöltésért felelős táblák:

fconnect

Az fconnect egy szimpla kapcsolótábla, amely arra szolgál, hogy több felhasználó és egy felhasználó több helyen is használhassa a fájlt.



lovifansdb fconnect	
	fcid : int(11)
	fiid : int(11)

files

A files adat tábla a fájlok eltárolását hivatott elvégezni. Tárolja a file feltöltési idejét, az új nevét, illetve az eredeti nevet. Az új név azt a célt szolgálja, hogy a weboldal számára könnyedén beazonosítható legyen a fájl, a régir pedig azért fontos megtartani, hogyha esetleg azt szeretnék, hogy letölthető legyen a fájl, akkor ajánlott az eredeti nevet használni, mert az sokkal felhasználóbarátabb.

leírás:

fiid: Fájl azonosítója (Primary Key)

uid: Felhasználó azonosító

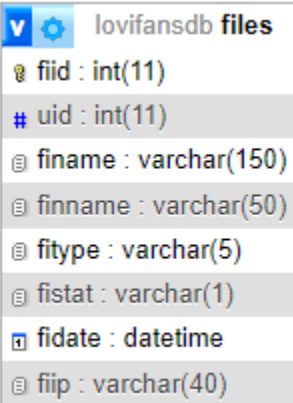
filename: Fájl neve









finname: Fájl belső neve

fitype: Fájl típusa (kép, video, hang, dokumentum)

fistat: Fájl státusza (törölt, aktív, felfüggesztett)

fidate: Fájl feltöltésének ideje



lovifansdb files	
	fiid : int(11)
	uid : int(11)
	filename : varchar(150)
	finname : varchar(50)
	fitype : varchar(5)
	fistat : varchar(1)
	fidate : datetime
	fiip : varchar(40)

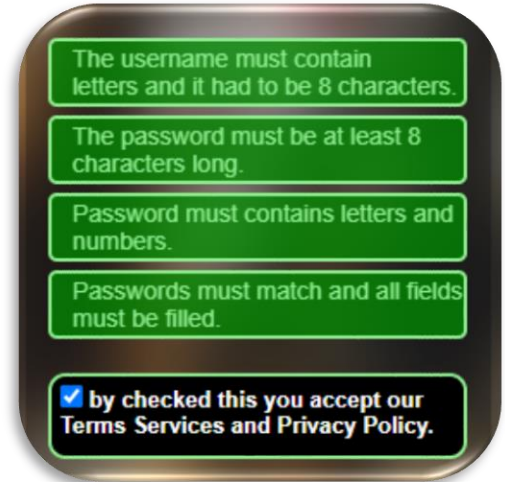
2.4 Algoritmusok

1. Regisztráció

Az algoritmusok regisztráció része annak ellenére, hogy talán ez a leghosszabb programrész, számomra az egyik legegyszerűbb volt.

Négy részből áll:

1. HTML
2. jQuery
3. PHP
4. MySQL



A HTML része rendkívül egyszerű, amiben text, submit és emial tagek vannak, utána következik a jQuery amivel leellenőrzöm, hogy az adatok megfelelnek a követelményeknek. A feltételek az oldalon is láthatóak, ezek egyenként zöldre változnak a feltételek teljesülésekor és elérhetővé válik a submit (elfogadás) gomb. A PHP részén pedig ezeket az adatokat megkapjuk `$_POST[]` globális változó segítségével és a PHP kóddal keretet adunk az MySQL kódnak, hogy feltöltsük az adatokat az adatbázisba.

HTML:

```
<form action="myfile.PHP" method="post">
    <input type="text" name="uname" id="uname" >
    <input type="email" name="umail" class="text" id="email">
    <input type="password" name="upass1" id="ptext1" >
    <input type="password" name="upass2" id="ptext2">
</form>
```

uname inputra jQuery reakció:

```
$("#uname").on('input', function(){
    var regex = /[a-zA-Z]/;
    var str = /[a-zA-Z]/;
    var num = /\d/;

    if($("#uname").val()!=" &
    $("#uname").val().length>7 & regex.test($("#uname").val())) {
        $("#1").css("color","lightgreen");
        $("#1").css("border-color","lightgreen");
        $("#1").css("background-color","rgba(0, 140, 0, 0.75)");
    }
    else{
        $("#1").css("color","red");
        $("#1").css("border-color","red");
        $("#1").css("background-color","rgba(140, 0, 0, 0.75)");
    }
    if((( $("#accepted").is(':checked') &
    $("#ptext1").val()!=" & $("#ptext2").val()!=" &
    $("#uname").val()!=" & $("#email").val()!=" &
    $("#ptext2").val() == $("#ptext1").val()) &
    (str.test($("#ptext1").val()) & num.test($("#ptext1").val())) &
    ($("#ptext1").val().length>7 ) &
    ($("#uname").val().length>7 & regex.test($("#uname").val()))){
        $("#signup").prop('disabled',false);
    } else $("#signup").prop('disabled',true);
});
```


PHP és sql:

```
$usern = $_POST["uname"];
$mail = $_POST["umail"];
$options = ["cost" => 12];
$pass =
password_hash($_POST["upass1"],PASSWORD_BCRYPT,$options);
$uids = mysqli_fetch_array(mysqli_query($dbase,"
select ustrid from users
"));

mysqli_query($dbase, "
INSERT INTO  users  ( uid ,  icid, bicid ,  ustrid ,  uname
,  umail ,  upass ,  ustat ,  udate ,  ucom ,  uip )
VALUES (NULL, NULL, NULL, '$usrtid', '$usern', '$umail',
'$pass', 'A, NLI', current_timestamp(), NULL,
'$_SERVER[REMOTE_ADDR]')
");
```

2. Bejelentkeztetés

A bejelentkeztetés tulajdonképpen két részből áll. Az egyik egy API a másik pedig egy HTML kód egy kis jQuery kóddal fűszerezve. Az api része felel azért, hogy a felhasználó a hibának megfelelő választ megkapja, illetve hogy a bejelentkezést végrehajtsa.

A felhasználó két három választ kaphat az API-ból:

1. Egy kulcsot, ami azt jelzi, hogy átmehet.
2. Egy hiba üzenetet arról, hogy nem töltötte ki az egyik mezőt.
3. Egy hiba üzenetet arról, hogy a felhasználónév vagy a jelszó hibás.

HTML:

```
<form id="loginForm">
    <span class="error d-done" id="error"></span>
    <input type="text" name="uname" class="text"><br>
    <input type="password" name="upass" >
    <input type="submit" value="Login" class="sub">
</div>
```

jQuery:

Itt két dolgot hajt végre a jQuery kód, továbbítja az adatokat a form tagból amelynél beállítjuk a method tulajdonságot post-ra és továbbítjuk az API-nak az adatokat, amelyet a HTML input tagek-ből kapunk, ezután kiolvassa az API válaszát, hogy lefrissítse az oldalt vagy hibaüzenetet dobjon.

API Működése:

Egy if függvénnyel leellenőrzöm, hogy mindent kitöltött a kliens, hogy a kapott felhasználónév az adott jelszó párral megtalálható az adatbázisban. Amennyiben igen, küld egy üzenetet a jQuery kódnak, hogy lefrissítse az oldalt, illetve a felhasználók adatait beletegye session változókba, ha nem létezik az adott felhasználó a jelszó párral akkor hibaüzenetet küld.

3. Kijelentkeztetés

Ebben az algoritmusban tulajdonképpen nincs semmi különleges, amikor ezt az algoritmust meghívom, akkor az unset() függvénnyel megszüntetem egyesével a session változókat amelyeket a bejelentkezésnél létrehoztam.

4. Email küldés

Számomra érdekesebb része a kódnak, az email küldés. Amikor elkezdtem több utánajárást, tanulást igényelt, kihívás volt számomra. Az interneten kutatva néztem utána a dolgoknak, majd rájöttem, hogy ehhez telepíteni kell az oldalamra egy csomagot, ami a levél küldést le tudja kezelni. Több lehetőség közül a PHPMailer csomagot választottam. Tesztelés után rájöttem, hogy ez nem egészen elegendő, mert választanom kellett az smtp, a pop3 és az imap közül. Végül rájöttem, hogy az imap, a pop3 és az smtp valójában szerverek amelyek elengedhetetlen ahhoz, hogy bármilyen levél átvitelt megtudjak valósítani. Szükségem volt tehát egy smtp szerverre, amellyel leveleket tudok továbbítani, ezért felregisztráltam egy mailersend.com nevezetű oldalra. A regisztrációval kaptam egy korlátozott smtp szerveret amihez megkaptam a szerver domain nevét és a jelszavát, így már tudtam levelet küldeni a weboldalon. A használathoz az smtp szerverhez a kódban meg kellett adnom az smtp szerver jelszavát, így hát úgy döntöttem, két részre osztom a kódot, mivel két helyen is használom. Az email küldésnél, hogy a contact oldalon el tudják nekem egyenesen küldeni a visszajelzéseiket, illetve a jelszó visszaállításnál használom, hogy elküldje nekik azt a linket amellyel vissza tudják állítani a jelszavaikat.

Az email küldésre használatos függvény:

```
function mailing($mail){  
  
    /*  
  
    Itt beimportálom a levél elküldéshez szükséges PHP fájlokat amelyek  
    a csomagban találhatóak.  
  
    */  
  
    require "$_SESSION[priv]/PHPMailer/src/Exception.PHP";  
    require "$_SESSION[priv]/PHPMailer/src/PHPMailer.PHP";  
    require "$_SESSION[priv]/PHPMailer/src/SMTP.PHP";  
  
    $mail = new PHPMailer(true);  
  
    //$mail->SMTPDebug = SMTP::DEBUG_SERVER;  
  
    $mail->isSMTP();  
  
    $mail->SMTPAuth = true;  
  
    //itt pedig megadom az smtp szerver adatait  
  
    $mail->Host = "smtp domain neve";  
  
    $mail->SMTPSecure = PHPMailer::ENCRYPTION_STARTTLS;  
  
    $mail->Port = 587; //smtp port  
  
    $mail->Username = "smtp email címe";  
  
    $mail->Password = "smtp jelszava";  
  
    $mail->isHTML(true);  
  
    return $mail;  
  
}
```

Emailt küldő kód

```
$mail = mailing("support.team@lovifans.com"); //küldő email cím

//ezt az emailt címet látja az aki az emailt kapja mint küldő

$mail->setFrom("problem@lovifans.com");

//erre az email címre fog érkezni az email

$mail->addAddress("becsei.szabi@gmail.com");

$mail->Subject = ""; //email tárgya

$mail->Body = <<<END

    <!--itt adom meg hogy mi lesz az email tartalma-->

END;

try{

    $mail->send();

    $_SESSION["send"] = true;

}

catch(Exception $e){

    //mail hiba üzenet $mail->ErrorInfo

}
```

5. Kommunikációs oldalak

A kezdő oldal nem sokban különbözik a chat oldaltól, ezért egybe vettem őket, három részből áll:

1. Megjelenítés
2. Üzenet feltöltése
3. Üzenet lekérése

Megjelenítés:

```
<div class="home sites">

  <div class="form">

    <div class="textarea">

      <!-- szöveges üzenetekre az input -->

      <textarea id="message" name="message" ro><textarea>

    </div>

    <div class="buttons">

      <!-- fájlok feltöltésére -->

      <button id="photos">image</button>

      <input hidden type="file" name="file" id="file" multiple

      <button onclick="post()">send</button>

    </div>

  </div>

  <div id="posts" class="posts"></div>

</div>
```

Üzenet feltöltése jQuery:

```
//Ez a lista arra szolgál, hogy több fájlt is fel lehessen tölteni.  
  
window.selfiles = [];  
  
//ez a függvény megvárja amíg ténylegesen feltöltődik a fájl  
  
function waitfile(){  
    return new Promise(function(resolve,reject){  
        $("#file").on("change",function(){  
            var len= $("#file")[0].files.length  
            if(len>0) resolve($("#file"));  
        });  
    });  
}  
  
//Ez a függvény a photos azonosítóval rendelkező gomb hatására meghívódik,  
// majd feltölti egy listába a fájlokat.  
  
$("#photos").click(function(){  
    $("#file").click();  
    waitfile().then(function(){  
        filelist = $("#file")[0].files;  
        for(var file of filelist){  
            window.selfiles.push(file);  
        }  
    });  
});
```

```

//Ez a függvény a send gomb hatására fut le, hogy elküldje a fájlokat
//annak a PHP fájlnek, amely felküldi őket az adatbázisba.

function post(){

    var sendr = r1 + 'pages/loged/post.PHP';

    var filein = document.getElementById('file');

    var formData = new FormData(); // form objectum létrehozása

    var xhr = new XMLHttpRequest(); //fájl feltöltésre létrehozott objektum

    if(window.selfiles) {

        var filelist = window.selfiles;

        // feltöltjük a formba a fájlokat

        if(filelist.length>0) for(var file of filelist) formData.append("files[]",file);

    }

    // feltöltjük az üzeneteket a formba

    if($("#message").val()!=""){

        formData.append("message",$("#message").val()); //

    }

    //itt megadjuk a method és az action paramétereket

    xhr.open("post",sendr);

    xhr.send(formData); // itt pedig elküldjük a form adatait

    window.selfiles = []; // kiürítjük alistát

    $("#message").val(""); // kiürítjük a textarea

}

```



```
post_fetch(); // megjeleníti a postokat  
  
setInterval(post_fetch,6000); // megfutattja amikor letelik a 6s
```

Üzenet feltöltése PHP:

```
session_start(); // session változók használatához elindítjuk a session-t  
  
//Betesszük connect.PHP fájlt, hogy megkapjuk az file_upload() függvényt  
  
//és az kapcsolatot teremt az adatbázissal.  
  
include("$ _SESSION[priv]/connect.PHP");  
  
//feltöltjük a fájlokat és a $fcid listába betesszük a fájlok fconnect táblában  
  
//lévő fcid nevezetű azonosítóját.  
  
if(isset($_FILES["files"])){  
  
    $len = count($_FILES["files"]["name"]);  
  
    for($i=0;$i<$len;$i+=1) $fcid[] = file_upload($dbase,$_FILES["files"],$i);  
  
}  
  
//Ha létezik a $_POST['message'] akkor betölti a $mes változóba, ha nem  
  
//akkor egy üres szöveget tölt bele.  
  
$mes = isset($_POST['message']) ? $_POST['message'] : "";  
  
//Ha létezik a $fcid akkor betölti a $files változóba , összefűzve ha nem  
  
//akkor egy üres szöveget tölt bele.  
  
$files = isset($fcid) ? implode(",",$fcid) : "";  
  
// ha csak az egyik változó nem üres, akkor szépen feltölti az adatbázisba.  
  
if($files!=" | $mes!=") mysqli_query($dbase,"  
  
INSERT INTO post (pid, uid, plabel, fcid, peloz, pstat, pdate, pip)
```

```
VALUES      (NULL,      $_SESSION[uid], '$mes', '$files', '', 'A', current_timestamp()),
'$_SERVER[REMOTE_ADDR]')

");

mysqli_close($dbase); //lezárom az adatbázist.
```

Üzenet megjelenítés PHP:

Készíték egy lekérdezést, amellyel megkapom a posztokat az adatbázisból,
majd elindítok egy while ciklust, amely végig megy a lekérdezésen.

A ciklusban eltárolom soronként egy listába azokat az adatokat amelyeket a felhasználók láthatnak:

```
$posts=[

    "name"          => $name,

    "profil_picture" => $profil_pic,

    "text"          => $label,

    'files'         => $files,

    'eloz'          => $eloz,

    'date'          => $date

];
```

Ezután megfuttatok két parancs sort.

```
header('Content-Type: application/json'); // a PHP kód létrehoz egy API -t

//kiíratja a kapott listát, mintha egy json api lenne

echo json_encode($posts);
```

Üzenet megjelenítés jQuery:

Ez a része két részből áll:

1. API lekérő függvény
2. Megjelenítő függvény

API lekérő függvény:

```
function post_fetch(){  
    var chatr = r1+'pages/loged/post_fetch.PHP'  
    $.ajax({  
        url: chatr,    //API link  
        method: 'GET',  
        dataType: 'json', //API típusa  
        success: function(data) {  
            displaypost(data); // a kapott adatok megjelenítése  
        },  
        error: function(jqXHR, textStatus, errorThrown) {  
            console.error('Error fetching messages:', errorThrown);  
        }  
    });  
}
```

Megjelenítő függvény:

```
function displaypost(posts){  
    $("#posts").html("");  
    posts.forEach(function(f){  
        var post_blueprint =`  
  
        ide írjuk hogyan szeretnénk megjeleníteni a kapott  
        adatokat, amikre a szövegben következőképpen hivatkozunk:  
        ${post.name}  
        `;  
        // itt hozzáfűzzük ahhoz az objektumhoz, amibe  
        // bele szeretnénk tenni a példányt  
        $("#posts").append(post_blueprint);  
    });  
}
```

6. Fájl feltöltés

```
function file_upload($dbase,$file,$index=0) {  
    // megengedett fájl kiterjesztések  
    $supfiles = [  
        ["jpg","jpeg","png","svg","gif"],  
        ["mp4","mov","avi","wmv","AVCHD"],  
        ["mp3","aac","aiff","alac","m4a","dsd"],  
        ["pdf","txt","pptx","xlsx","docx"]  
    ];  
  
    $ftypes = ["image","video","audio","docs"]; //fájltípusok mappái  
    $support = false;  
  
    // megvizsgáljuk, hogy a fájl létezik-e  
    if(!empty($file["name"][$index])) {  
        //változókba tesszük a fájl adatait  
        $fname = $file["name"][$index];  
        $ftype = $file["type"][$index];  
        $fext = explode(".", $fname)[1];  
        $stype = "";
```

```

//megvizsgáljuk, hogy a fájlt támogatja a szoftverünk

for($i = 0; $i < 4; $i++) {

    foreach($supfiles[$i] as $s) {

        if($s == $fext) {

            $support = true;

            $stype = $ftypes[$i];

            break;

        }

    }

}

// Ha támogatja, illetve nem lépi át a mérete 209715200 byte-ot, akkor

//elkezd a fájl tárolását.

if($support && $file["size"][$index] <= 209715200) {

    $path = "$_SESSION[priv]/uploads/" . $stype . "/" . date("Y-m-d")."/";

    if(!is_dir($path)) mkdir($path); // létrehozzuk a fájl mappáját

    //megadjuk a fájl új nevét

    $new_fname = date("His") . $_SESSION["ustrid"] . "$index" . "." . $fext;

    $uploadedFile = $path . $new_fname ;

    // ha a fájl eltárolása sikeres volt, akkor az adtbázisba is felkerül

    if(move_uploaded_file($file['tmp_name'][$index], $uploadedFile)) {

```

```
//betesszük a files adattáblába
```

```
mysqli_query($dbase,"INSERT INTO files
```

```
( fiid, uid, fname, finame, fitype, fistat ,fideate, fiip )
```

```
VALUES
```

```
(NULL, $_SESSION[uid], '$fname', '$new_fname',
```

```
'$stype','A' ,current_timestamp(), '$_SERVER[REMOTE_ADDR]')
```

```
");
```

```
$fid = mysqli_insert_id($dbase); // itt megkapjuk a fájl fiid
```

```
// majd itt a fájl azonosítóját az fconnect táblában is eltároljuk.
```

```
mysqli_query($dbase,"
```

```
INSERT INTO fconnect ( fcid , fiid )
```

```
VALUES (NULL, $fid)
```

```
");
```

```
$fcid = mysqli_insert_id($dbase);
```

```
return "$fcid";
```

```
// ha sikeresek voltunk visszatérünk az fconnect azonosítójával.
```

```
} else return ";
```

```
} else return ";
```

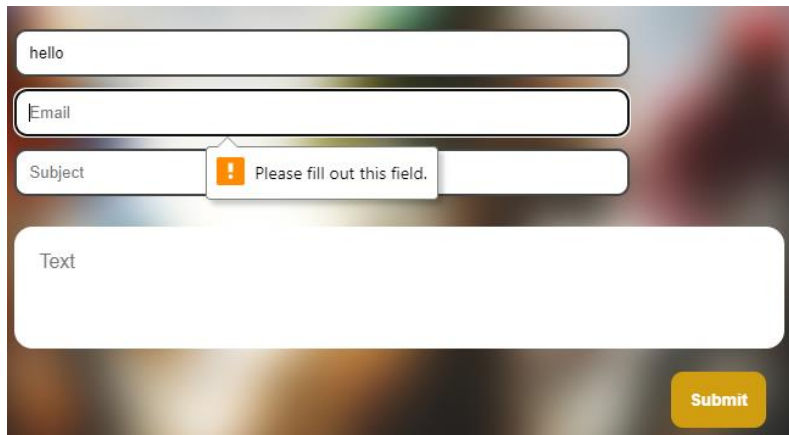
```
} else return ";
```

```
}
```

2.5 Teszt dokumentáció

Kapcsolat oldal kitöltetlen részek

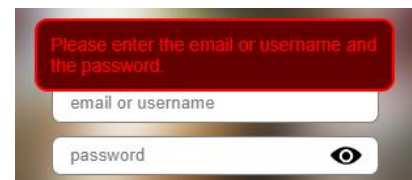
A címkék required paraméter hozzáadásával kezeltem a hibát



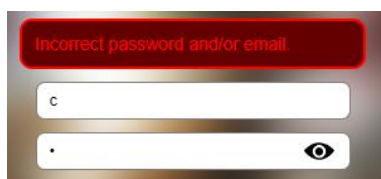
A screenshot of a contact form on a website. The form has four input fields: a text field containing 'hello', an email field, a subject field, and a large text area. A yellow tooltip with an exclamation mark icon points to the subject field, displaying the message 'Please fill out this field.' A yellow 'Submit' button is located at the bottom right of the form.

Login

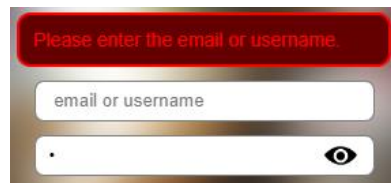
A hiba kezelést úgy oldottam meg, hogy egy API ellenőrzi az input feldolgozása során megfelelnek a követelményeknek. Ki van-e töltve az összes beviteli mező, illetve hogy a felhasználónév a jelszó párral megtalálható az adatbázisban.



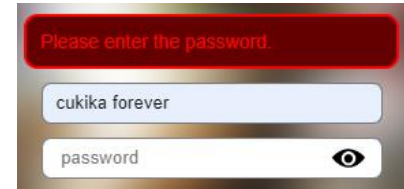
A screenshot of a login form. It features two input fields: 'email or username' and 'password'. A red error message box at the top reads 'Please enter the email or username and the password.' The password field has an eye icon for toggling visibility.



A screenshot of a login form showing a validation error. A red error message box at the top reads 'Incorrect password and/or email.' The 'email or username' field contains the letter 'c', and the 'password' field contains a single dot. Both fields have eye icons for toggling visibility.



A screenshot of a login form showing a validation error. A red error message box at the top reads 'Please enter the email or username.' The 'email or username' field is empty, and the 'password' field contains a single dot. Both fields have eye icons for toggling visibility.

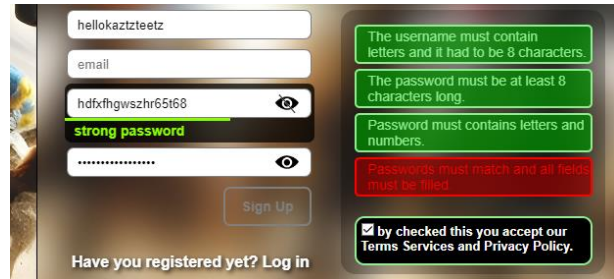


A screenshot of a login form showing a validation error. A red error message box at the top reads 'Please enter the password.' The 'email or username' field contains the text 'cukika forever', and the 'password' field is empty. Both fields have eye icons for toggling visibility.

Sign Up

jQuery segítségével írtam egy részletes feltétel sort, amely ellenőrzi, hogy az adott sorok megfelelnek a követelményeknek. Amennyiben nem fellel meg

addig nem enged belépni. PHP kód segítségével ellenőriztem, hogy használták-e már azt az email címet.



The Sign Up form includes the following fields and validation rules:

- Username:** hellokaztzeetz (Valid: The username must contain letters and it had to be 8 characters.)
- Email:** hdxftgwszhr65t68 (Valid: The password must be at least 8 characters long.)
- Password:** strong password (Valid: Password must contain letters and numbers.)
- Confirm Password:** (Valid: Passwords must match and all fields must be filled.)

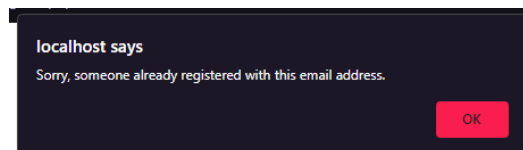
Buttons: Sign Up, Have you registered yet? Log in. A checkbox for Terms Services and Privacy Policy is checked.



The Sign Up form shows error messages for the email and password fields:

- Email:** Please include an '@' in the email address. 'hxfjgfgfghj' is missing an '@'.
- Password:** The password must be at least 8 long.

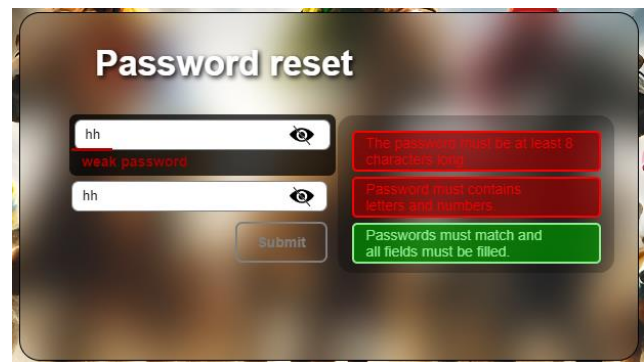
Buttons: Sign Up, Have you registered yet? Log in. A checkbox for Terms Services and Privacy Policy is checked.



localhost says
Sorry, someone already registered with this email address.
OK

Password-reset

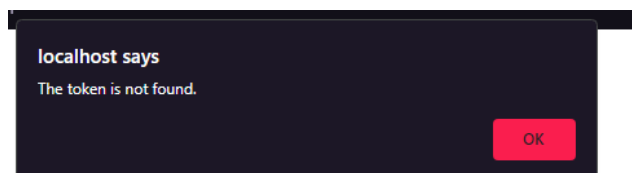
Ennél az oldalnál ismételten használtam amit már a Sign Up oldalnál is. A jQuery kódban átírtam az összes feltételt és a sajátjára vonatkozókat külön, amelyben megadtam a háttérszínt, a szélek színét, és a betűk színét.



The Password reset form includes the following fields and validation rules:

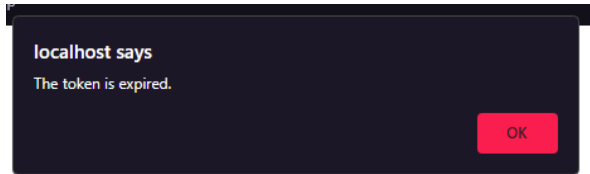
- Current Password:** hh (Valid: The password must be at least 8 characters long.)
- New Password:** hh (Valid: Password must contain letters and numbers.)
- Confirm Password:** (Valid: Passwords must match and all fields must be filled.)

Buttons: Submit.



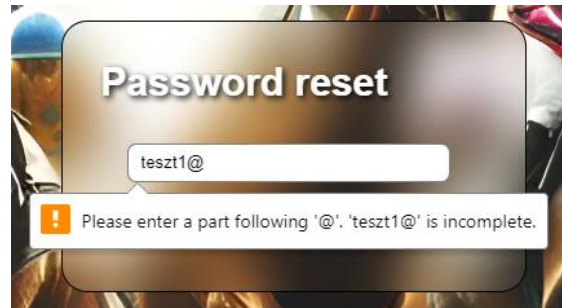
localhost says
The token is not found.
OK

A PHP fájlban lekezeltem, hogy lejárt, nem is létezett, vagy már használták a token.



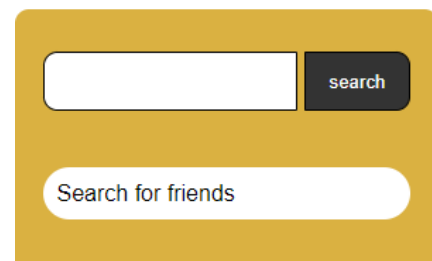
send- send-password-reset

Ez szintén egy szimpla required-es kezelés melyhez kapcsoltam egy PHP feltételt is, mely abból áll hogy megnézi. Az email létezik-e az adatbázisban. Amennyiben nem akkor szimplán lefrissíti az oldalt.



Friends

Ha nincsenek még a felhasználónak barátai az oldalon, akkor megjeleníti a barátok keresését.



2.6 Fejlesztési javaslatok

Post oldalt esztétikusabbá tenni, videófájlok küldését kijavítani, a chat és a post oldal fájl küldésnél, fájlok megjelenítése amikor kiválasztásra kerülnek, profil oldal fejlesztése, a fiók beállításoknál funkciókat bővíteni, illetve a fiók törlés opciót létre hozni.

3. Felhasználói dokumentáció

3.1 Futtatási környezet

Böngésző megnyitása után weboldal linkjét írjuk be a keresőbe.

támogatott böngészők:



Opera

<https://www.opera.com/>



Chrome

<https://www.google.com/chrome>

weboldal linkje:

<https://lovifans.com>

3.1 Program

Ez a program a felhasználók közti kommunikációt teszi lehetővé.

3.2 Regisztráció

A regisztráció a bejelentkezéshez szükséges, amellyel elérhetővé válnak az oldal funkciói. A regisztráció pofon egyszerű,

be kell írni egy tetszőleges nevet

ide

pl. Kovács József

Az email címet pedig ide

pl. jozsef.kovacs@example.com

és találjunk ki egy jelszót ide

pl. takacs1234

majd elolvasás után pipálja ki itt

majd miután mindent rendben
talált, fogadja el itt

Kovács József

jozsef.kovacs@example.com

takacs1234

medium password

takacs1234

Sign Up

Have you registered yet? Log in

The username must contains letters and it had to be 8 characters.

The password must be at least 8 characters long.

Password must contains letters and numbers.

Passwords must match and all fields must be filled.

☒ by checked this you accept our Terms Services and Privacy Policy.

3.3 Bejelentkezés

a nevet, amelyet megadott írja be ide

a megadott jelszót írja be ide

majd fogadja el

Login

email or username

password

Login

Don't have an account? Sign Up

Forgot password?

3.4 Jelszó visszaállítás

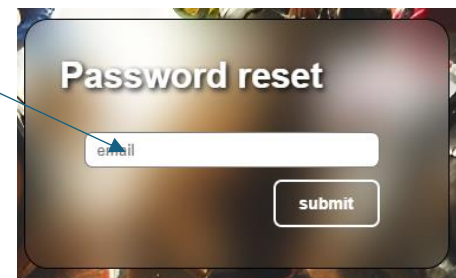
A jelszó visszaállítást akkor kell használni amikor a jelszót elfelejtettük, ilyenkor nyomjunk rá a Forgot password? linkre.



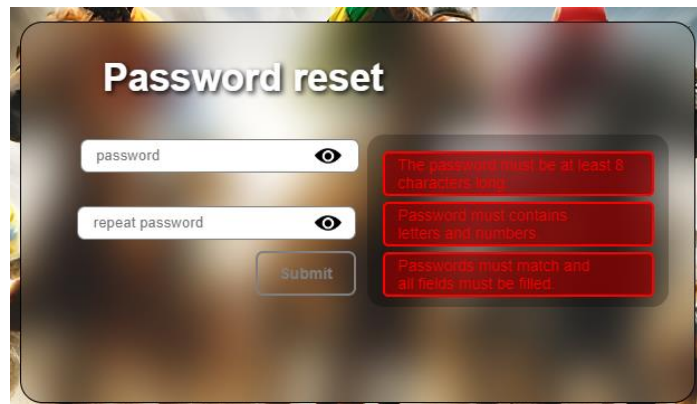
Amit megadtunk email címet az oldalra regisztrálása során írjuk be ide.

Ezután lépünk be az email

fiókunkba majd nyomjunk a here linkre



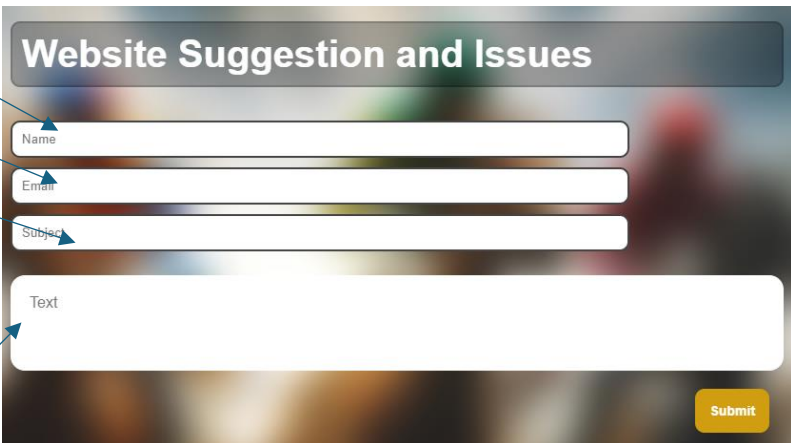
A link átirányítja erre az oldalra, ahol majd meg kell adni az új jelszót, ahogy a regisztrációnál.



3.5 Kapcsolat

Ezt az oldalt akkor kell használni, amikor valamilyen hibát észleltünk, vagy javaslat jut az eszünkbe, itt lehet elérni a fejlesztőt.

írja be a nevét
az email címét
írja be a tárgyat
itt megfogalmazhatja
javaslatait és üzeneteit



The image shows a web form titled "Website Suggestion and Issues". It has four input fields: "Name", "Email", "Subject", and "Text". The "Text" field is a larger text area. A yellow "Submit" button is located at the bottom right of the form. Blue arrows point from the Hungarian labels on the left to each of the four input fields.

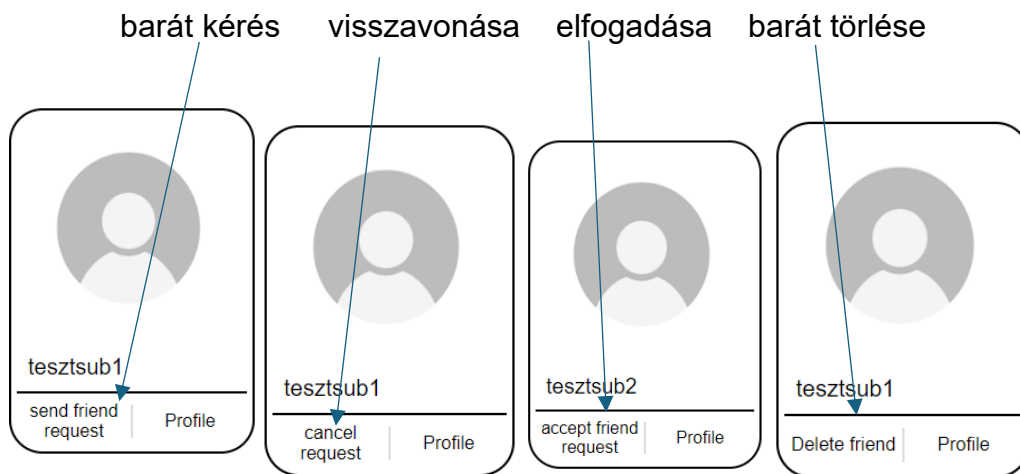
3.6 Barát keresés

Ezen az oldalon tudunk barátokat keresni, hozzá adni és törölni, mely lehetővé teszi az oldalon az egymás közötti kommunikációt, anélkül hogy bárki más látná.

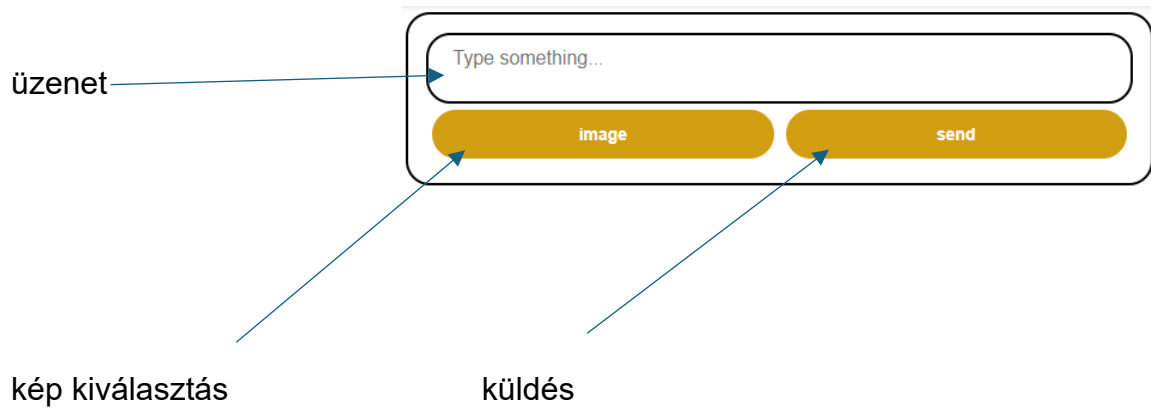
itt lehet keresni



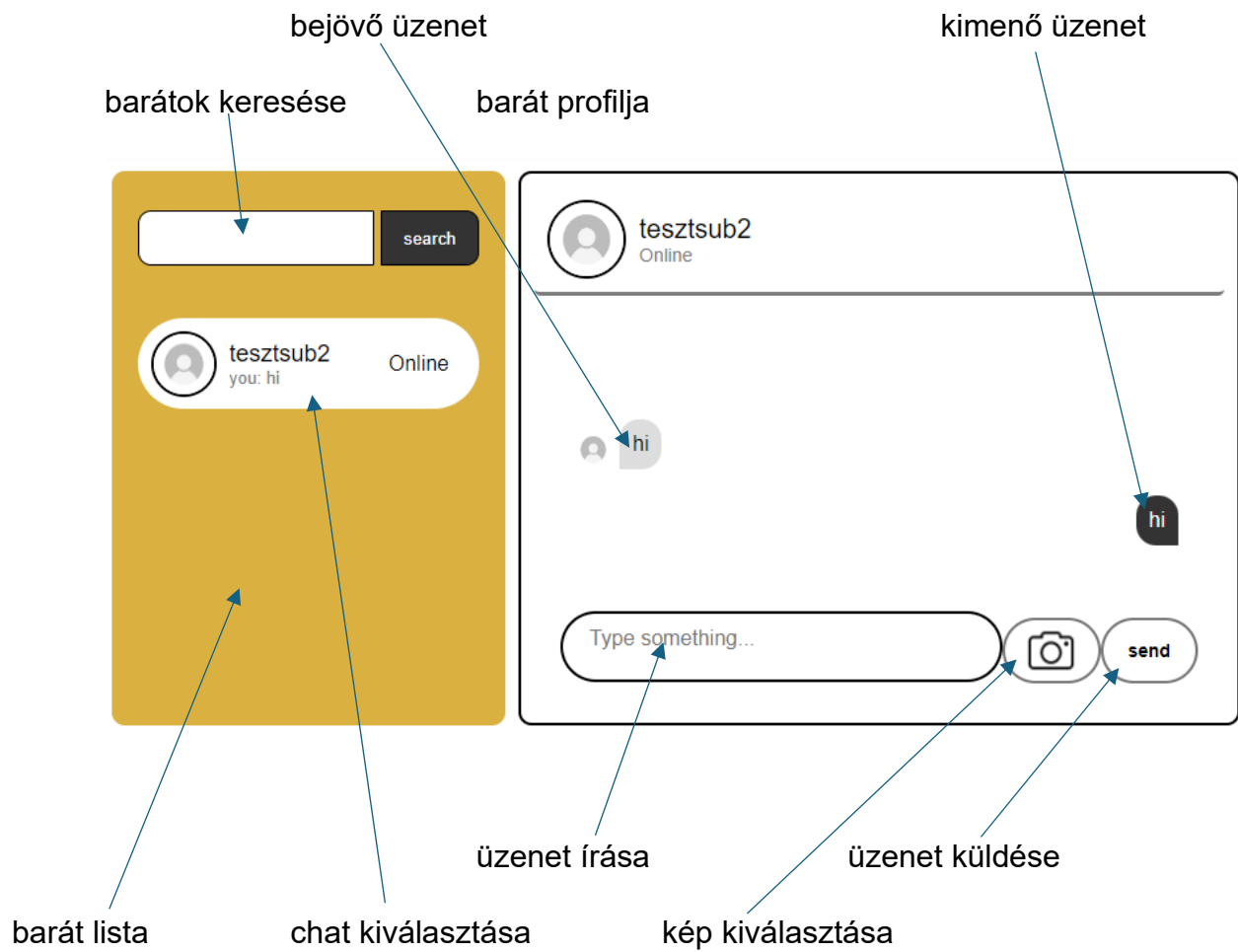
The image shows a friend search interface. It features a search bar with a "submit" button. Below the search bar are six buttons: "All", "Friend requests", "Friends of my Friends", "Are still unknown?", and "My friend's profiles". A blue arrow points from the Hungarian label "itt lehet keresni" to the search bar.



3.7 Megosztás



3.8 Chatelés



4.Összefoglalás

4.1 Munkám értékelése

Kihívásnak éreztem, kezdetben sok kérdés merült fel, sok újdonsággal szembesültem, fejlesztettem magam, kerestem a hasznosítható parancsokat a feladatok megoldásához. Úgy érzem a webfejlesztés területén fejlődtem és sokkal komplexebbnek látom, mint elsőre gondoltam volna. Sokat tanultam belőle, a nehézségeken sikerült túllendülni. Összességében úgy érzem, elégedett vagyok, az idő arányában amit lehetett befektettem a munkámba, a programozás, kidolgozás, tanulás és idő terén is.

4.2 Programom későbbi hasznosulása

Miután ennek a félévnek vége lesz, úgy tervezem a fejlesztői javaslatokban feltüntetett ötletekre megoldást keresek. A programot felteszem a web-re, reményeim szerint használatából is sok tapasztalatot szerzek, bővíthető és kombinálható más felületekkel. Mindenféleképpen sokat nyertem a program megírásával, mert sok tapasztalatot és tudást szereztem, illetve referenciaként is fel szeretném használni.

5. Irodalom jegyzék

Források:

szakmai jegyzékek:

PHP: <https://www.PHP.net>

W3schools: <https://www.w3schools.com>

infojegyzet.hu: <https://infojegyzet.hu>

stack overflow: <https://stackoverflow.com>

képek forrása:

Clipdrop: <https://clipdrop.co/uncrop>

DALL-E: <https://labs.openai.com>

Designer image-creator: <https://designer.microsoft.com/image-creator>

Leonardo AI: <https://app.leonardo.ai/ai-generations>

Szoftver forrás kódja:

<https://github.com/becseiszabolcs/lovifans.com>