

WebTapQA

Online kvalitatív analízis

Készítette:

Szatmári Szabolcs

Szanyi Levente

Tóth Kristóf

Tartalomjegyzék

Bevezetés.....	3
Használat	3
Fejlesztés	6
Használt szoftver	6
xampp	6
apache	7
mariadb	7
phpmyadmin	7
VSCode.....	7
Használt nyelvek.....	8
html	8
php.....	8
css	8
javascript	8
Adatbázis felépítés	9
reagentreactions tábla	9
user_reagentreactions tábla	9
users tábla	10
temp_users tábla.....	10
posts tábla	11
Lényegesebb algoritmusok.....	11
A regisztrációs algoritmusunk	11
Az email igazolás algoritmus	12
A bejelentkeztető algoritmusunk	13
Publikus reakciók api algoritmus.....	14
Jövőbeli tervek	14

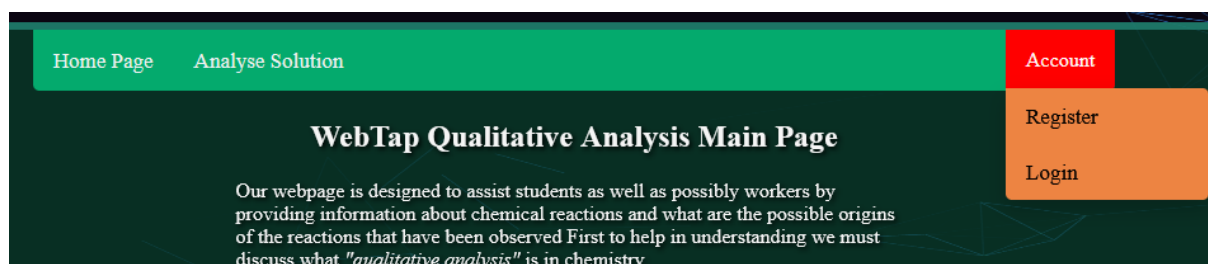
Bevezetés

A mi általunk készített webalkalmazás azzal a céllal jött létre, hogy kémiai tanuló diákoknak segítséget nyújthasson a kvalitatív analízisben.

Az oldal eredete egy előző iskolából hozott tapasztalat ahol több alkalommal is hasznos lett volna egy webes felület, ami segítséget nyújt a feladatok megoldásában.

Használat

Az oldal megnyitása után a főoldalon találjuk magunkat ahol átnavigálhatunk az analyze solution oldalra vagy az account menüpont alatt beléphetünk / regisztrálhatunk.



Az analyze solution oldalon ki tudjuk választani az általunk használt anyagokat és a viselkedésüket.

Miután kiválasztottunk egy anyagot a megadottak közül megtekinthetjük hogy mennyire valószínű hogy azzal az anyaggal dolgozunk.

A bejelentkezett felhasználók számára van 3 extra oldal amit használhatnak, egy komment fal ahol hagyhatnak visszajelzéseket, egy share reactions oldal ahol egyedi reakciókat adhatnak hozzá a rendszerhez, és egy load reactions oldal ahol mások által létrehozott reakciókra iratkozhatnak fel későbbi használatához.

A komment fal megnyitása után láthatunk egy szöveges dobozt egy post gombbal, ide tudjuk írni az üzenetünket és a gombra kattintással hozzáadódik az ez alatt lévő komment listához. A komment lista felett van egy keresősáv azok számára akik szeretnének keresni a kommentek között.

A share reactions oldal tartalmaz 11 bemeneti mezőt ahol megadhatjuk az adatokat a saját reakcióink létrehozásához, amiből az utolsó egy megosztási kód amit ha üresen hagyunk mindenki láthatja és használhatja a reakciónkat, de ha kitöltjük csak azok használhatják akik tudják a kódot és készítőjének nevét.

A load reactions oldalon láthatunk két bemeneti mezőt ahol tudjuk megadni a reakció kódját és feltöltőjének nevét hogy használni tudjuk, alatta az általunk követett vagy létrehozott reakciók listájával ami felett szintén van egy kereső.

Below you can upload a few solutions to the database tied to your account that others may subscribe to and see when their analysis page is running.

Name of Uploader

Reaction's Code

Add Reaction

A reakció listában lévő reakciók tartalmaznak két gombot, az elsővel hozzá lehet adni / eltávolítani a jelenleg használt reakciókhoz, a másodikkal pedig ki lehet törölni az általunk létrehozott reakciókat.

Public Reactions

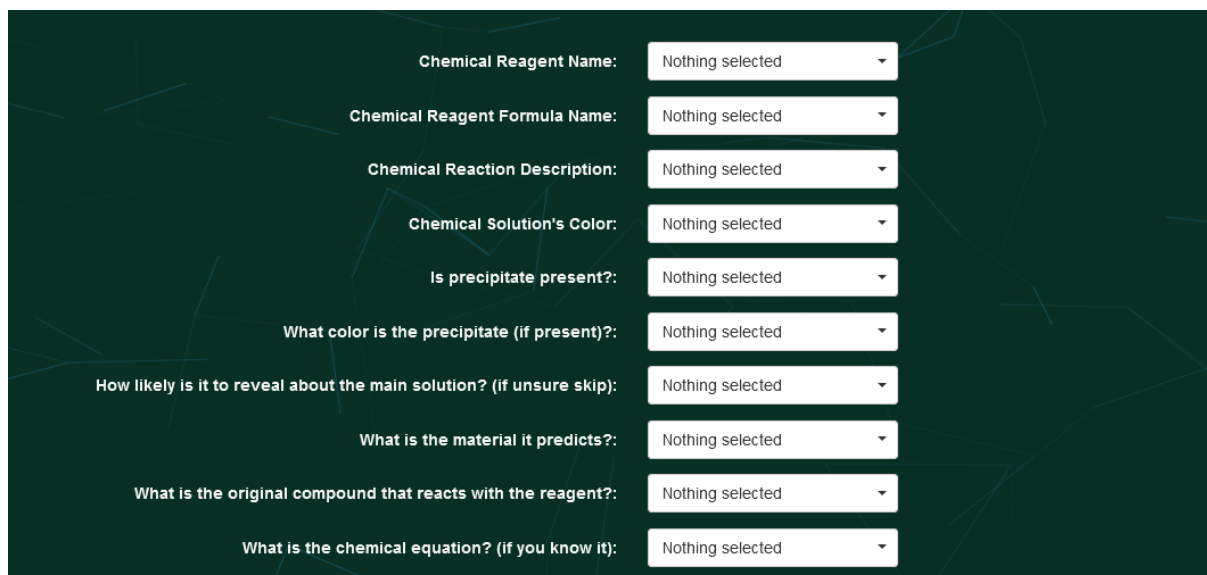
Search for keywords

<p>Reaction Code: Reagent Name: wa.mo Reagent Chemical Name: omae Reaction Description: baka Color Change: yellow Precipitate Present: 0 Precipitate Color: bluze Prediction Value: 50 Expected Atom/Ion: asd Expected Chemical: asdda Reaction Equation: adddasa</p> <p>Add to my reactions</p>	<p>Reaction Code: Reagent Name: wa.mo Reagent Chemical Name: omae Reaction Description: baka Color Change: yellow Precipitate Present: 0 Precipitate Color: bluze Prediction Value: 50 Expected Atom/Ion: asd Expected Chemical: asdda Reaction Equation: adddasa</p> <p>Add to my reactions</p>	<p>Reaction Code: Reagent Name: Hydrochloric acid Reagent Chemical Name: asassaa Reaction Description: gzzzz Color Change: ruaaaaa Precipitate Present: 1 Precipitate Color: shkeeeeee Prediction Value: 60 Expected Atom/Ion: asd Expected Chemical: das Reaction Equation: dasasd</p> <p>Add to my reactions</p>	<p>Reaction Code: Reagent Name: asd Reagent Chemical Name: asd Reaction Description: asd Color Change: asd Precipitate Present: 0 Precipitate Color: asd Prediction Value: 100 Expected Atom/Ion: asd Expected Chemical: asd Reaction Equation: asd</p> <p>Add to my reactions</p>
--	--	--	--

Ez alatt láthatjuk ugyanebben a formátumban a publikus reakciókat, most csak egy gombbal amivel bekövethetjük őket, ez a lista is tartalmaz egy keresősávot.

Az analyze solution oldalon láthatunk 10 lenyitható menüt ahol bevihetünk adatokat a keresett vegyülethez hozzáadott anyagunkról mint például az anyag nevét teljes név és kémiai formulája is elfogadott, a reakció leírása pl.: „az anyag színe sárgás lett”, a reakció eredményének színét, hogy a reakciónk tartalmaz-e szilárd anyagot és a színét ha tartalmaz és hasonló adatokat.

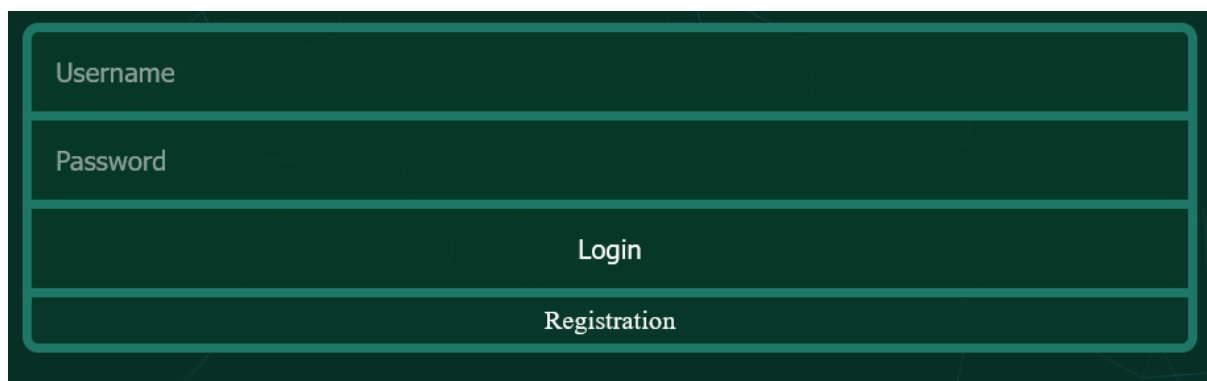
Miután kitöltöttük rákereshetünk, hogy milyen anyagokra igazak ezek a változások, ezek után kapunk egy anyag listát amiből kiválaszthatjuk hogy miket adunk az anyagunkhoz és megnézhetjük hogy hogyan változik az anyagunk a reakció miatt és kizárásos alapon ki tudjuk találni hogy mi is valójában az anyagunk a viselkedése alapján.



A screenshot of a web form with a dark green background and a faint molecular structure pattern. The form contains ten rows, each with a label on the left and a white dropdown menu on the right. All dropdown menus currently display 'Nothing selected'. The labels are: 'Chemical Reagent Name:', 'Chemical Reagent Formula Name:', 'Chemical Reaction Description:', 'Chemical Solution's Color:', 'Is precipitate present?:', 'What color is the precipitate (if present)?:', 'How likely is it to reveal about the main solution? (if unsure skip):', 'What is the material it predicts?:', 'What is the original compound that reacts with the reagent?:', and 'What is the chemical equation? (if you know it):'.

Emellett még van a regisztrációs és bejelentkezése oldalak ahol be lehet jelentkezni és regisztrálni.

A bejelentkezési oldalon van kettő bemeneti mező ahova beírjuk a felhasználónevünket és jelszavunkat, ezután a login gombra kattintva be fogunk jelentkezni ami hozzáférést ad a másik 3 oldalhoz.

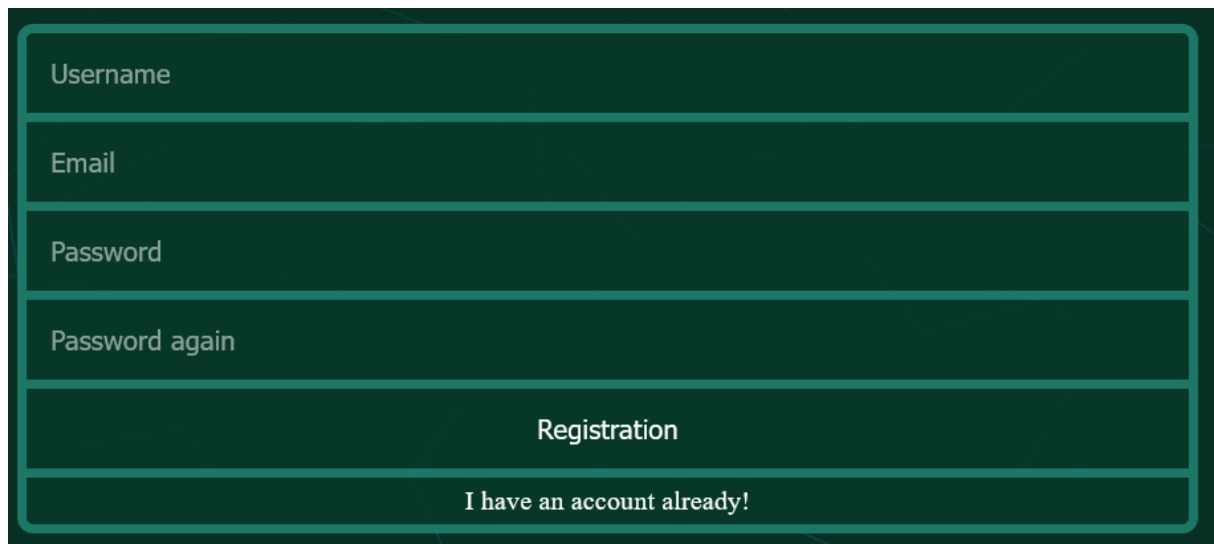


A screenshot of a login and registration form with a dark green background. It features two input fields at the top labeled 'Username' and 'Password'. Below these fields are two buttons: 'Login' and 'Registration', both with a light green gradient and rounded corners.

A regisztrációs oldal egy fokkal bonyolultabb, itt kell a felhasználónév, egy email cím, és a jelszavunk kétszer.

A felhasználónevünknek a minimum hossza 8 karakter és maximum hossza pedig 30 karakter.

A jelszónak szintén 8 karakter hosszúnak kell lennie, emellett tartalmaznia kell egy számot és nagy betűt is, ha minden adatot sikeresen adtunk meg akkor rákattintunk a register gombra, ami küld egy email az email címünkre, ez tartalmaz egy linket, amire van 12 óránk rákattintani hogy sikeres legyen a regisztráció, ha nem sikerül a link megnyitása a megadott időn belül akkor a felhasználó automatikusan törlődik.

A registration form with a dark teal background and rounded corners. It contains five input fields: 'Username', 'Email', 'Password', 'Password again', and a 'Registration' button. Below the button is a link that says 'I have an account already!'.

Fejlesztés

Használt szoftver

- xampp
- apache
- mariadb
- phpmyadmin
- visual studio code

xampp

A weboldal fejlesztéséhez használt szoftvercsomag, ami nagyon hasznos tud lenni mivel egyben tartalmazza az apache2 webservert, a mariadb adatbázist, a php scriptnyelvet és a phpmyadmin programot is, mind előre bekonfigurálva hogy ne kelljen konfigurációval foglalkozni a legelejétől.

A xampp onnan ered hogy valaki csinálni akart egy cross-platform programcsomagot a lamp/wamp stack minden elemét tartalmazza egy alkalmazáson belül.

xampp -> cross-platform apache, mariadb, php, perl.

Mivel a xampp gyors fejlesztéshez lett kitalálva nagyon nem ajánlott hogy erről fusson egy publikus oldal, az összes konfiguráció arra van állítva hogy könnyű hozzáférése legyen mindenhez ami egy publikus szerveren okozhat problémákat.

apache

A webszerver amit használtunk, az apache egy viszonylag régi program, de a mai napig a második helyet tartja világszerte 1995 óta.

Az egyetlen webszerver ami megelőzi az az újabb Nginx ami az egyszerű használat és bővíthetőség helyett a sebességre fókuszál.

Ismert arról, hogy nagyon egyszerűen bővíthető kiegészítőkkal, amik sok esetben megkönnyíthetik a munkát.

mariadb

Az adatbázisunk a MariaDB, ami egy nyílt forráskódú adatbázis amit az eredeti mysql fejlesztők csináltak miután a mysql-t megvette az oracle és fizetőssé tette a használatát.

A mariadb egy sokkal modernebb és biztonságosabb verziója a mysql adatbázisnak amellest hogy még ingyen is elérhető.

Egy pár extra funkció amit a mariadb tud és a mysql nem:

json adattípus aminek a tartalmára is lehet szűrni lekérdezésekben (ez viszonylag lassú de lehetséges)

dinamikus oszlopok ahol egy táblán belül két sornak lehetnek különböző oszlopai, ez lehetséges egy json mezővel is de ez a módszer sokkal effektívebb

phpmyadmin

A phpmyadmin egy php-ban írt grafikus adatbáziskezelő program.

Nagyon hasznos tud lenni mivel az adatbázist alpból csak parancsok begépelésével lehet irányítani és egy vizuális felület ahol egyszerre látsz mindent sokat tud segíteni.

VSCode

A visual studio code a szövegszerkesztő amit mi használtunk ehhez a webalkalmazáshoz.

A visual studio code a legismertebb szövegszerkesztő program mivel egyszerűen bővíthető kiegészítőkkal és ingyenesen elérhető.

Használt nyelvek

- php
- html
- css
- javascript

html

A html minden weboldalhoz kötelező mivel ez az egyetlen dolog amit a böngésző meg tud jeleníteni (médiafájlokon kívül).

A mi oldalunk html-je php által van legenerálva, css-el kidekorálva hogy modern kinézete legyen és reszponzív legyen.

php

A php egy szerver oldali scriptnyelv amit mi az oldal html kódjának legenerálására használunk.

A php adatbázisban lévő adatok és felhasználótól kapott adatok alapján generál le egy sima html weboldalt.

A php egy dinamikus nyelv amit sokan nem szeretnek az egyszerűsége miatt.

A php előnyei számunkra azok voltak hogy gyárilag bele van építve minden amire szükségünk van

Hátrányai viszont hogy phpban szinte lehetetlen követni a clean code-ot.

CSS

A css a weblapok kinézetért felelő nyelv.

A mi weboldalunk tartalmazza a bootstrap keretrendszert, ami egy előre elkészített css csomag gyakran használt elemekkel hogy felgyorsítsa a weboldal készítését és a reszponzivitást is megoldja hogy ne kelljen külön oldal csinálni ahoz hogy mobilon is jól működjön.

javascript

A javascript a fő webes programozási nyelv ami segít interaktívabbá tenni a weboldalunkat.

Jquery: a javascript keretrendszer amit arra használunk hogy kis változások esetén ne kelljen újratölteni az egész oldalt csak egy-egy elemét aminek változnia kell.

Wantajs + threejs: az animált háttérképért felelő javascript csomag, ezekkel lehetséges bármiféle 2 vagy 3ds animációkat csinálni a böngészőben.

Adatbázis felépítés

reagentreactions tábla

Az oldalhoz előre elkészített reakciókat tartalmazza.

- ChemId – int(11)
- ReagentName – varchar(255)
- ReagentChemicalName – varchar(255)
- ReactionDesc – varchar(255)
- ColorChange – varchar(255)
- PrecipitatePresent – bool
- PrecipitateColor – varchar(255)
- PrecipitateValue – int(11)
- MaterialOne – varchar(255)
- MaterialTwo – varchar(255)
- ReactionEquation – varchar(255)

ReagentName: a reagens neve, a reakcióban ez okozza a változást.

ReagentChemicalName: a reagens kémiai vegyülete.

ReactionDesc: a reakció leírása.

ColorChange: a reakció által kiváltott színváltozás.

PrecipitatePresent: szilárd anyag jelenléte a vegyületben.

PrecipitateColor: a szilárd anyag színe.

PrecipitateValue: a szilárd anyag befolyásának mértéke százalékban

MaterialOne: az anyag, amire indirekten utal a reakció.

MaterialTwo: az anyag, amire direktén utal a reakció.

ReactionEquation: a reakció leírása.

user_reagentreactions tábla

Tartalmaz minden adatot, amit a reagentreactions tábla is tartalmaz csak egy pár extrával.

A felhasználók által létrehozott reakciókat tartalmazza.

- uploaderID – int(11)
- reactionCode – varchar(100)

uploaderID: a létrehozó felhasználó fiókazonosítója.

reactionCode: egy egyedi azonosító a reakcióhoz, ennek segítségével lehet megosztani másokkal.

users tábla

A felhasználói fiók adatait tartalmazza.

- id – int(32)
- username – varchar(80)
- password – varchar(60)
- email – varchar(100)

id: felhasználó egyedi azonosítója.

username: a felhasználó felhasználóneve.

password: a felhasználó jelszavának hashe.

email: a felhasználó email címe.

temp_users tábla

A frissen regisztrált felhasználók adatait tartalmazza, ameddig nem erősítik meg a fiókjukat vagy nem jár le a megadott idő, miután törölve lesz.

- id – int(32)
- username – varchar(80)
- password – varchar(60)
- email – varchar(100)
- code – varchar(100)
- time – int(32)

id: egyedi azonosító.

username: felhasználónév.

password: hashelt jelszó.

email: email cím.

code: azonosító kód.

time: a regisztráció ideje.

posts tábla

Felhasználók által hagyott kommenteket tartalmazza amiket egy komment falon lehet megtekinteni.

- id – int(11)
- post – varchar(1000)
- upload_time – timestamp
- uploaderID – int(11)

id: egyedi azonosító.

post: a komment tartalma.

upload_time: a komment írásának ideje.

uploaderID: a komment írójának felhasználói azonosítója.

Lényegesebb algoritmusok

A regisztrációs algoritmusunk

```
function register($conn, $uname, $pass, $pass2, $email) {
    if(strlen($pass) < 8) {
        return false;
    }
    if($pass != $pass2) {
        return false;
    }
    if(!preg_match('/^.+@.+[.].+$/ ', $email)) {
        return false;
    }

    if(strlen($uname) > 30 || strlen($uname) < 8) {
        return false;
    }
    if(!preg_match('/\d/', $pass) && !preg_match('/[A-Z]/', $pass)) {
        return false;
    }

    $existing_users = $conn->execute_query("SELECT * FROM users WHERE username=? OR email=?", [$uname, $email]);
    $existing_users_tmp = $conn->execute_query("SELECT * FROM temp_usr WHERE username=? OR email=?", [$uname, $email]);
    if($existing_users->num_rows != 0 || $existing_users_tmp->num_rows != 0) {
        echo "this username or email already taken ";
        return false;
    }else{
        $hash = password_hash($pass, PASSWORD_BCRYPT);
        $code = bin2hex(random_bytes(30));
        $conn->execute_query("INSERT INTO temp_usr VALUES(NULL, ?, ?, ?, ?, UNIX_TIMESTAMP())", [$uname, $hash, $email, $code]);
        return true;
    }
}
```

Kezdeként ellenőrizzük a jelszó hosszát, hogy egyezik-e a két megadott jelszó, az email valóságát, a felhasználónév minimum és maximum hosszát, és végül a jelszó nagybetű és szám tartalmát, ezek után megnézzük, hogy az adatbázis tartalmazza e már a felhasználónevet vagy email címet, és ha igen elutasítjuk a regisztrációt.

Ha sikeres, akkor le hasheljük a jelszót a bcrypt algoritmussal, hogy ne tároljuk el az eredetit, emellett generálunk egy random kódot, amit elküldünk a felhasználó email címére és a felhasználó belekerül az ideiglenes felhasználók adatbázisba.

Az email igazolás algoritmus

```
require("../shared/php/connection.php");
$code = $_GET['code'];

if(isset($code)) {
    $users = $conn->execute_query("SELECT * FROM temp_usr WHERE code=?", [$code]);
    if($users->num_rows != 1) {
        echo "user doesnt exist";
        return;
    }
    $user = $users->fetch_assoc();
    if($user["time"] > time() + 3600) {
        echo "registration request timed out";
        return;
    }
    $conn->execute_query("INSERT INTO users VALUES(NULL, ?, ?, ?)", [$user["username"], $user["password"], $user["email"]]);
    $conn->execute_query("DELETE FROM temp_usr WHERE code=?", [$code]);
    header("Location: ../main/index.php");
}
```

Az emailben kapott link megnyitásakor megnézzük, hogy létezik-e a kód, ha létezik, akkor ellenőrizzük, hogy nem e járt le, és ha még mindig érvényes, akkor áthelyezzük a felhasználót az ideiglenes adatbázisból a fő adatbázisba és átirányítjuk a kezdőlapra.

A bejelentkeztető algoritmusunk

```
function login($conn, $uname, $pass) {  
    if(strlen($uname) == 0 || strlen($pass) == 0) {  
        return false;  
    }  
  
    $users = null;  
    if(str_contains($uname, "@")) {  
        $users = $conn->execute_query("SELECT * FROM users WHERE email=?", [$uname]);  
    }else{  
        $users = $conn->execute_query("SELECT * FROM users WHERE username=?", [$uname]);  
    }  
  
    if($users->num_rows == 1) {  
        $user = $users->fetch_assoc();  
        if(password_verify($pass, $user["password"])) {  
            return $user["id"];  
        }  
    }  
    return false;  
}
```

Először ellenőrizzük hogy nem e lett üresen hagyva a felhasználónév vagy a jelszó sem.

Ezek után megnézzük hogy esetleg felhasználónév helyett emaillel próbálnak e bejelentkezni és az alapján keressük ki a felhasználót, ha létező a felhasználó akkor sikeresen bejelentkeztetjük.

Publikus reakciók api algoritmus

```
$keys = [
    "ReagentName",
    "ReagentChemicalName",
    "ReactionDesc",
    "ColorChange",
    "PrecipitatePresent",
    "PrecipitateColor",
    "PredictionValue",
    "MaterialOne",
    "MaterialTwo",
    "ReactionEquation"
];

function hide_private(&$out, $db, &$keys) {
    while($record = $db->fetch_assoc()) {
        $rr = [];
        foreach ($keys as $key) {
            $rr[$key] = $record[$key];
        }
        array_push($out, $rr);
    }
}

$response = [];

$databse = $conn->query("SELECT * FROM user_reagentreactions WHERE reactionCode != ''");
hide_private($response, $databse, $keys);

$databse = $conn->query("SELECT * FROM reagentreactions;");
hide_private($response, $databse, $keys);

echo(json_encode($response));
```

Ez az algoritmus visszaadja json formátumban az összes publikusan elérhető reakciót.

A teljes adatbázisból lekér minden adatot ami publikusan elérhető és az előre megadott \$keys változó alapján elrejt azokat az adatokat amikre a felhasználónak valószínűleg nincs haszna, például az egyedi azonosítót.

Jövőbeli tervek

A jelenlegi kommentfalat átalakítani egy fórumra mivel az hasznosabb lehet a felhasználók számára.

