HALADÓ WEB PROGRAMOZÁS

Téma cím: Eseményszervező weboldal létrehozása

|  |  |
| --- | --- |
| Készítette: Pécsi Erik  Leckekönyv: 26221008  Szak: Informatika | Tanár: Zlatko Čović |

Tartalomjegyzék

[Projektleírás 3](#_Toc88554483)

[Adatbázis 4](#_Toc88554484)

[Csatlakozás az adatbázishoz 5](#_Toc88554485)

[Auth session 6](#_Toc88554486)

[Adatbevitel 7](#_Toc88554487)

[SQL-lekérdezések 8](#_Toc88554488)

[CSS 9](#_Toc88554489)

[Mobil applikáció 10](#_Toc88554490)

[Kutatás 14](#_Toc88554490)

[API 15](#_Toc88554490)

[Felhasznált irodalom 17](#_Toc88554490)

# Projektleírás

A projekt lényege egy eseményszervező weboldal elkészítése, amely lehetőségeket kínál szórakozni vágyó embereknek és rendezvény szervezőknek egyeránt. Az oldal célja, hogy ezt a két közönséget összekapcsolja. A weboldal hasznos információkat kínál egyes események értékeléseiről. Lehetőség nyílik publikus események böngészésére. Az oldal hírességeket is funkcióval lát el, miszerint nekik is létrehozható külön esemény. A projekt magába foglal egy mobil alkalmazást is, ami szorosan kapcsolódik a weboldalhoz és onnan is elérhető minden funkció. A projekt elérhető a <https://void.stud.vts.su.ac.rs/logins.php> oldalon. Extra funkciók amit az oldal kapott:

- Privát / Publikus / Híresség események

- Események értékelése

- Híresség rendszer

# Adatbázis

Az adatbázis a weboldal működésének alapját képezi, ahol tárolódnak és kezelésre kerülnek a felhasználói adatok. Az adatbázis lehetővé teszi a weboldal számára, hogy hatékonyan kezelje és rendszerezze az adatokat, és megfelelően válaszoljon a felhasználók kéréseire. Az adatbázis lehetővé teszi a felhasználók regisztrációját, bejelentkezését, események szervezését, stb.

# Csatlakozás az adatbázishoz

<?php  
  
$host = "host";  
$username = "username";  
$password = "password";  
$database = "database";  
  
$con = mysqli\_connect($host, $username, $password, $database);  
  
  
if (mysqli\_connect\_errno()) {  
 echo "Failed to connect to MySQL: " . mysqli\_connect\_error();  
}  
?>

Ábra 1: csatlakozás az adatbázishoz

Először is definiálódnak a kapcsolódáshoz szükséges változók:

- $host: Az adatbázis szerver neve vagy IP címe.

- $username: A felhasználónév, amellyel az adatbázishoz kapcsolódni szeretnénk.

- $password: A jelszó a megadott felhasználónévhez.

- $database: Az adatbázis neve, amelyhez kapcsolódni szeretnénk.

Ezt követően a mysqli\_connect() függvény segítségével próbálunk kapcsolódni az adatbázishoz a fent megadott információk alapján. A mysqli\_connect() függvény visszaad egy kapcsolatot az adatbázissal, vagy false értéket, ha a kapcsolódás sikertelen.

Ezt követően a mysqli\_connect() függvény segítségével próbálunk kapcsolódni az adatbázishoz a fent megadott információk alapján. A mysqli\_connect() függvény visszaad egy kapcsolatot az adatbázissal, vagy false értéket, ha a kapcsolódás sikertelen.

# Auth session

Csak bejelentkezett felhasználók:

<?php  
session\_start();  
if(!isset($\_SESSION["username"])) {  
 header("Location: login.php");  
 exit();  
}  
?>

Ábra 2: Bejelentkezett felhasználók

Vendégek is engedélyezve:

<?php  
session\_start();  
if(!isset($\_SESSION["username"]) || $\_SESSION["username"] == "Vendég") {  
 header("Location: login.php");  
 exit();  
}  
?>

Ábra 3: Csak vendégek

Az if utasítás vizsgálja meg, hogy van-e értéke a $\_SESSION["username"] változónak. A $\_SESSION munkamenet változóit tartalmazza. Ebben az esetben az ellenőrzés arra irányul, hogy a "username" kulcsú változó létezik-e a munkamenetben. A második esetben is ez történik, annyi kivétellel, hogy a “Vendég” is engedélyezve van az oldalon. Vendégként a bejelentkező felületen található gombbal tudunk belépni.

# Adatbevitel

Az oldalon lehetőségünk van adatokat bevinni az adatbázisba. Egy példa erről:

    $updateQuery = "

        UPDATE events

        SET name = '$name', event\_date = '$eventDate', place = '$place', type = '$type'

        WHERE id = '$eventId'

    ";

}

Ábra 4: Adatbevitel

Ez a kód egy SQL UPDATE lekérdezést végrehajt, amely módosítja az events táblában található esemény adatait. A lekérdezés az id oszlop alapján azonosítja a módosítandó eseményt, és frissíti az esemény nevét, időpontját, helyszínét és típusát az új értékekkel, amelyeket a változók ($name, $eventDate, $place, $type) tartalmaznak.

# SQL-lekérdezések

Az oldal több helyen is használ SQL lekérdezést, adatok kiírására. Egy példa erről:

    $eventsQuery = "SELECT \* FROM events WHERE user\_id IN (SELECT id FROM users WHERE username = '$username')";

$eventsResult = mysqli\_query($con, $eventsQuery);

            echo "        <span class='label'>ID:</span> $eventId<br>";

            echo "        <span class='label'>Esemény neve:</span> $name<br>";

            echo "        <span class='label'>Időpont:</span> $date<br>";

            echo "        <span class='label'>Helyszín:</span> $place<br>";

            echo "        <span class='label'>Típus:</span> $type<br>";

            echo "        <span class='label'>Megjegyzés:</span> $comment<br>";

Ábra 5: SQL lekérdezés

Ez a kód először lekérdezi az events táblát, hogy az aktuális felhasználó által létrehozott eseményeket megkapja, a users tábla alapján az username változóval. A lekérdezés eredményeit ($eventsResult) feldolgozva, a kód az események adatait (ID, név, időpont, helyszín, típus és megjegyzés) HTML-ben formázott szövegekként jeleníti meg.

# CSS

A Cascading Style Sheets (CSS) egy stíluslap nyelv, amelyet a weboldalak kinézetének és elrendezésének irányítására használnak. A CSS segítségével meg lehet határozni, hogy az elemek hol helyezkedjenek el a weboldalon. A CSS segítségével animációkat és átmeneteket lehet létrehozni. Ezek az animációk lehetnek például gombnyomásra történő mozgások vagy színes változások. A CSS segítségével a weboldalak mérete csökkenthető, ami gyorsabb betöltést eredményezhet. A kódot összevonhatjuk, felesleges ismétlődő szabályokat eliminálhatjuk, és minimálisra csökkenthetjük a képek és egyéb erőforrások mennyiségét. CSS segítségével az oldalt reszponzívvá tehetjük, ami azt eredményezi, hogy minden készülékről könnyen olvasható lesz az oldalunk.

@media (max-width: 768px) {  
 .navbar {  
 width: 100%;  
 position: static;  
 justify-content: center;  
 align-items: center;  
 }  
  
 .content {  
 margin-left: 0;  
 }  
}

Ábra 6: CSS kód

Ez a CSS kód egy media lekérdezést definiál, amely a képernyő szélességét vizsgálja. Ha a képernyő szélessége 768 pixeltől kisebb vagy egyenlő, akkor az itt megadott stílusok alkalmazódnak. A media lekérdezések lehetővé teszik a stílusok testreszabását és alkalmazkodását különböző kijelzőméretekhez és eszközökön.

# Mobil applikáció

A mobil applikáció létrehozására react nativet használtam.

 export default function CreateEvent({ navigation }) {

    const [event, setEvent] = useState({

        name: '',

        event\_date: '',

        place: '',

        type: '',

        comment: '',

    });

    const handleCreate = async () => {

        const response = await fetch('https://void.stud.vts.su.ac.rs/create\_event\_api.php', {

            method: 'POST',

            headers: {

                'Content-Type': 'application/json',

            },

            body: JSON.stringify(event),

        });

        const data = await response.json();

        if (data.success) {

            Alert.alert('Success', data.message);

            navigation.goBack();

        } else {

            Alert.alert('Error', data.message);

        }

    };

Ábra 7: Esemény létrehozása

const fetchEvents = async () => {

            try {

                const response = await fetch('https://void.stud.vts.su.ac.rs/events\_api.php');

                const data = await response.json();

                if (data.success) {

                    setEvents(data.events);

                } else {

                    setError(data.message);

                }

            } catch (err) {

                setError('Hiba t�rt�nt az esem�nyek lek�r�sekor.');

            } finally {

                setLoading(false);

            }

        };

  );

}

Ábra 8: Esemény beolvasása

const handleUpdate = async () => {

        const response = await fetch('https://void.stud.vts.su.ac.rs/modify\_event\_api.php', {

            method: 'POST',

            headers: {

                'Content-Type': 'application/json',

            },

            body: JSON.stringify({

                id: eventId,

                ...event

            }),

        });

        const data = await response.json();

        if (data.success) {

            Alert.alert('Success', data.message);

            navigation.goBack();

        } else {

            Alert.alert('Error', data.message);

        }

    };

Ábra 9: Esemény módosítása

            const response = await fetch('https://void.stud.vts.su.ac.rs/delete\_event\_api.php', {

                method: 'POST',

                headers: {

                    'Content-Type': 'application/json',

                },

                body: JSON.stringify({ id: eventId }),

            });

            const data = await response.json();

            if (data.success) {

                Alert.alert('Success', data.message);

                setEvents(events.filter(event => event.id !== eventId));

            } else {

                Alert.alert('Error', data.message);

            }

Ábra 10: Esemény törlése

# Kutatás

A kutatás az eventbrite oldalán történt. Az eventbrite számos funkciót kínál, események létrehozása, jegyértékesítés, nyilvántartás és ezekhez kapcsolódó dolgok például: jegy vásárlás.

Az oldal felhasználóbarát, könnyen eligazodtam rajta. Sok funkció található meg rajta, különbőző jegyek (pl: VIP, Sima) létrehozása. Számos platformon elérhető ez az oldal.

Hátrányként hoznám fel, hogy az oldal néhol fizetős és korlátozza az ingyenes felhasználókat.

További utánajárás után kiderült, hogy sok felhasználó panaszkodik az oldalra olyan jegyvásárlásnál ami nagyon kelendő, ezt sajnos nem tudtam kipróbálni.

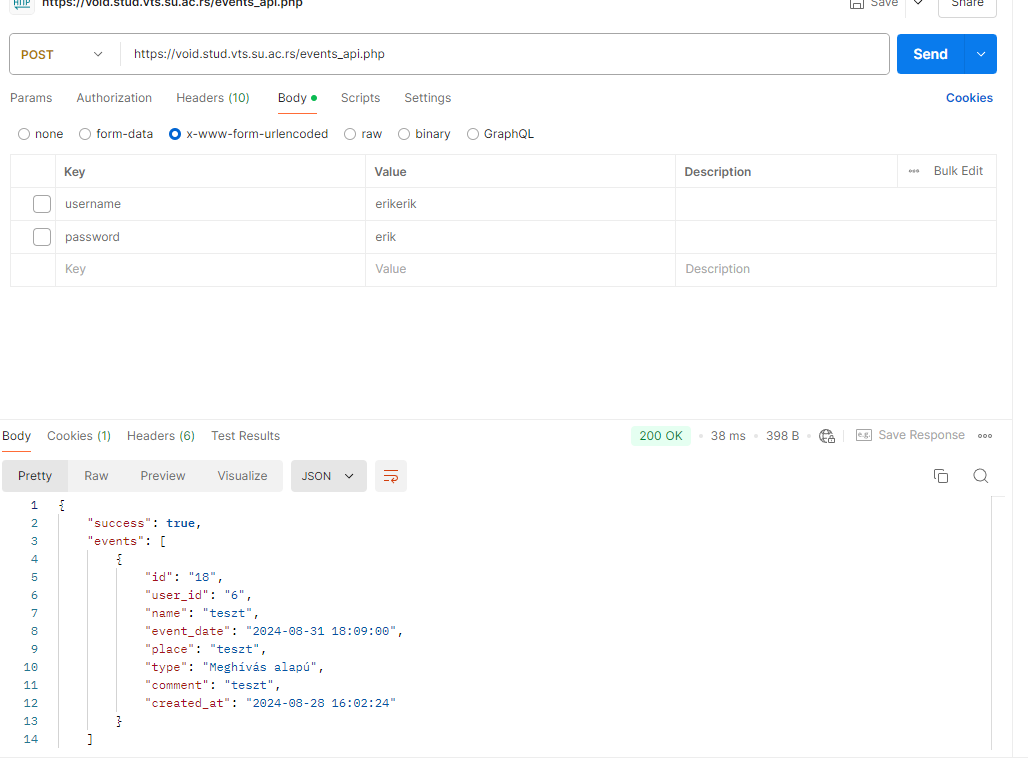
# API

Az oldalon API-kat használunk a mobilos alkalmazással való összeköttetésre. Ezek például:

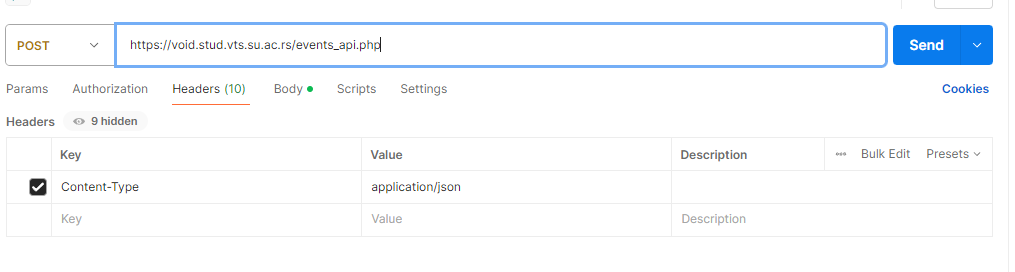
- events\_api az események lekérdezésére

- modify\_event\_api az esemény módosítására

**Postman lekérdezés**



Ábra 11: events\_api postman lekérdezése



Ábra 11: postman header beállítás

# Felhasznált irodalom

1. PHP oldala - <https://www.php.net>
2. React Native Guide - <https://reactnative.dev/docs/getting-started>

[3] stackoverflow - https://stackoverflow.com