Vas Vármegyei Szakképzési Centrum

Nádasdy Tamás Technikum és Kollégium

SZAKDOLGOZAT

Weblap/alkalmazás zenében érdeklődőknek

Bánó András, Fehér Tamás, Szalay Kristóf

Konzulens:
Varga Gábor

2024

Nyilatkozat

Alulírott, **Bánó András, Fehér Tamás, Szalay Kristóf** Szoftverfejlesztő és tesztelő szakos hallgató kijelentem, hogy a Weblap/alkalmazás zenében érdeklődőknek című szakdolgozat feladat kidolgozása a saját munkám, abban csak a megjelölt forrásokat, és a megjelölt mértékben használtam fel, az idézés szabályainak megfelelően, a hivatkozások pontos megjelölésével. Eredményeim saját munkán, számításokon, kutatáson, valós méréseken alapulnak, és a legjobb tudásom szerint hitelesek.

Csepreg, 2024.04.11 _____

tanuló

Csepreg, 2024.04.11 _____

tanuló

Csepreg, 2024.04.11 _____

tanuló

Kivonat

Weblap/alkalmazás zenében érdeklődőknek

A projektünk feltörekvő zenészek segítésére készült. A projektünk fő célja, hogy zenészek vagy zenében érdeklődő felhasználók tudjanak együtt kollaborálni. Az oldalunkon a felhasználók képesek kiposztolni az egyéni vagy a kollaborált zeneművet.

A weboldalunkon lehet saját profilt létrehozni regisztrálással, ahol a felhasználó létrehozhat magának egy felhasználónevet, a profiladatokhoz le tudja írni az érdeklődési köreit, egyéni személyes leírásait, van e tapasztalata a zeneszerzésben vagy a zenélésben.

A szakdolgozaton dolgozott: Bánó András, Fehér Tamás, Szalay Kristóf

Abstract

Website/application for those interested in music

Our project is designed to help aspiring musicians. The main goal of our project is for musicians or users interested in music to be able to collaborate together. On our site, users can post individual or collaborative music.

On our website, you can create your own profile by registering, where the user can create a username, describe their interests, individual personal descriptions, experience in composing or playing music.

Worked on the thesis: András Bánó, Tamás Fehér, Kristóf Szalay

Tartalomjegyzék

1. Bevezetés

A csapatunk 2024-es záróvizsgájának a témája egy olyan weboldal, amely a feltörekvő zenészek számára biztosít egy olyan platformot, ahol megtudják mutatni magukat a nagyvilágnak. Az oldalunkra feltudják tölteni a műveiket, bármilyen olyan zenével kapcsolatos alkotmányukat, amelyet ők készítettek és úgy érzik, hogy az figyelmet érdemel. Ezáltal figyelmet és nézettséget generálva saját maguk, valamint a weboldalunk számára is. Az oldalunkon lehetőségük lesz „barátok”, a közös szenvedéllyel rendelkező egyének megismerésére, akikkel a jövőben akár tudnak közösen is dolgozni. Ezáltal még egy lépéssel közelebb lépve az ismertség felé. A lehetőségek végtelen tárháza nyílik meg a művészek számára, amint regisztráltak az oldalra.

Miután regisztrált a felhasználó az oldalra, ezáltal létrehozva saját felhasználói profilját, onnantól kezdve az adatai mentésre kerülnek egy adatbázisban eltárolva. Ennek következtében a következő alkalommal már csak bejelentkeznie kell a már létező fiókjába. A felhasználónak miután regisztrált lehetősége nyílik a posztokra való reagálásra, valamint ha vannak saját posztjai, akkor értesítések érkezésére is számíthat. Ha a felhasználónak megtetszik egy esetleges feltöltés az oldalon nyugodtan rá tud menni a feltöltő profiljára további munkák felfedezésének a lehetőségéért. Valamint, ha az adott feltöltő profilján több adott munka is megtetszik a felhasználónak, akkor lehetősége nyílik arra, hogy akár a jövőben felvegyék a kapcsolatot, ezáltal egy lépéssel közelebb juthatnak, akár egy közös projekt megalkotásához. A mi projektünkön hárman dolgoztunk. Az oldal tervezését úgy osztottuk fel, hogy az oldal backend részén Fehér Tamás dolgozott, míg a frontend részen Szalay Kristóf és Bánó András, az oldal adatbázisának a kidolgozása és a megvalósítása közös munka volt.

Munkánkat szorgalmas kutatásokkal, utána nézéssel, kutatásokkal MySQL-ben, Laravelben, illetve nem utolsó sorban TypeScript-ben, VisualStudioCode-ban végeztük.

A projektünk Laragonon alapszik. A backend tesztelésekhez PostMan-t alkalmazva. A projektunk hosszú távú célja, még nem biztos. További fejlesztéseket tervezünk hozzá, hogy ki tudja nőni magát, hogy több emberhez eljuthasson. Akik még nem tudják, hogy hogyan és miként mutassák meg tudásuk, hogyan ismertessék meg magukat az emberekkel, az internet felhasználóival.

Az oldal felhasználása teljesen ingyenes, a felhasználók nyugodtan pénz fizetése nélkül hozzáférhetnek az oldal minden specifikációjához, ami egy féle megnyugvást, illetve kényelem érzetet, komfortot biztosít a felhasználó számára. Az oldalon több féle műfajban tudnak egymásra találni a feltörekvő művészek, így gond nélkül meg tudják találni az ugyanabban a műfajban alkotó felhasználóink a jövőben. Valamint az oldalon nem muszáj feltölteni saját műveket. Egy felhasználó csak fel tud regisztrálni az oldalra olyan szándékkal, hogy az ő, saját maga által szívesen hallgatott műfajban tudjon olyan zenékre találni, melyeket szívesen hallgatna. Emellett lehetőség van akár csak zenei alapok feltöltésére is, valamint remixek kiadására is. A projektünk fő célja, hogy egy olyan biztos közeget teremtsünk a felhasználóknak, ahol szabadnak érezhetik magukat, hogy megmutathassák, hogy kik is ők valójában. Itt lehetőségük nyílik szabadjára engedni a személyiségüket így szárnyra kaphat a képzeletük a zene terén, hiszen számtalan helyről tudnak majd inspirációt nyerni, ami következtében akár egy átlag felhasználó is lehet, hogy ötletet kap ahhoz, hogy kipróbálja magát ezen a téren. A mi számunkra az lenne a legjobb, ha minél többen regisztrálnának az oldalunkra, ezáltal elismerve a munkánkat, melyet a weboldal elkészítésébe öltünk. Az oldalunk elkészítése előtt, illetve közben is rengeteg kutatást végeztünk, hogy hogyan is tudnánk a lehető legjobban létrehozni azt a weboldalt, amit elképzeltünk. Így tudjuk biztosítani a felhasználók számára a gördülékeny és könnyen megérthető működését az oldalunknak. Próbáltunk létrehozni egy letisztult, a mai világban minden téren elfogadott felhasználói felületet, amelyen nem számítanak a világ összetűzései csak a zene iránti szenvedély.

2. Fejlesztői dokumentáció

2.1. HASZNÁLT PROGRAMOZÁSI NYELVEK ÉS TECHNOLÓGIÁK

2.1.1. TypeScript

A TypeScript egy, a JavaScript kibővítéseként és továbbgondolásaként létrehozott, objektum-orientált programozási nyelv. A TypeScript legmeghatározóbb fejlesztése, hogy a JavaScript-et határozott típusdefiníció és típuskikötések használatának lehetőségével egészíti ki, ami lehetővé teszi a kód típusmegfelelőségre történő helyességellenőrzését már a futás megkezdése előtt egy statikus analízist alkalmazó fordító számára.

2.1.2. Php

A PHP egy szerveroldali szkriptnyelv, mely segítségével dinamikus weblapokat készíthetünk. A PHP nyelven írt kódokat a webszerver PHP feldolgozómodulja értelmezi. A PHP egy olyan programozási nyelv, mely segítségével képesek lehetünk elkészíteni egy adatbázisalapú weboldalt is.

2.1.3. Node.Js

A Node.js egy nyílt forráskódú, backend oldali JavaScript futtatókörnyezet. Használatával JavaScript alkalmazásokat és kódokat futtathatunk böngészőkön kívül, akár saját számítógépünkön a Google által fejlesztett V8 JavaScript motor segítségével. Mivel támogatja az összes főbb webes protokollt, beleértve a HTTP-t, a HTTPS-t és a WebSockets-t, így könnyedén használhatjuk webkiszolgálók fejlesztésére. A segítségével könnyedén készíthetünk webszolgáltatásokat, chat alkalmazásokat, játékokat és más valós idejű alkalmazásokat. Az összes főbb platformot támogatja, beleértve a Windows-t, Linux-t és a Mac OS X-t is. Rendelkezik egy fejlett modulrendszerrel, segítségével könnyedén hozzáadhatunk új funkciókat az alkalmazáshoz. Azért választottuk a Node.js-t, mert egy nyílt forráskódú platform, és nagyon aktív fejlesztői közössége van, ami jelentősen növeli a platform életképességét és gyors fejlődését.

2.1.4. GitHub

A GitHub egy olyan profitorientált vállalat, amely felhőalapú Git-tárház hosting szolgáltatást kínál. Lényegében nagyban megkönnyíti az egyének és a csapatok számára a Git használatát a verziókezeléshez és az együttműködéshez. A GitHub felülete elég felhasználóbarát, így még a kezdő kódolók is kihasználhatják a Git előnyeit. GitHub nélkül a Git használatához általában egy kicsit több technikai hozzáértésre és a parancssor használatára van szükség. A GitHub azonban annyira felhasználóbarát, hogy egyesek más típusú projektek kezelésére is használják a GitHubot. Ezen túlmenően bárki ingyenesen regisztrálhat és tárolhat egy nyilvános kódtárat, ami különösen népszerűvé teszi a GitHubot a nyílt forráskódú projektek körében. Vállalatként a GitHub pénzt keres azáltal, hogy eladja a tárolt privát kódtárakat, valamint más üzletközpontú terveket, amelyek megkönnyítik a szervezetek számára a csapattagok és a biztonság kezelését. A Githubot széles körben használjuk a Kinstánál belső projektek kezelésére és fejlesztésére.

2.1.5. React

A React (vagy ReactJS) egy nyílt forráskódú JavaScript keretrendszer, amely lehetővé teszi felhasználói felületek készítését webalkalmazásokhoz. A React alapvetően egy komponens alapú keretrendszer, ami azt jelenti, hogy az alkalmazásokat egymástól függetlenül épített, újra felhasználható komponensekre bontja. Ezek a komponensek tartalmazzák a logikát és az adatok megjelenítését is, és lehetővé teszik a fejlesztők számára, hogy az alkalmazásukat dinamikusan kezeljék. A React a virtuális DOM (Document Object Model) koncepcióját használja, amely lehetővé teszi a hatékonyabb és gyorsabb alkalmazásfrissítést a valódi DOM-hoz képest. Ez azt jelenti, hogy az alkalmazás állapotának megváltozása esetén a virtuális DOM-nál történik

2.2. FEJLESZTŐI KÖRNYEZETEK

2.2.1. Visual Studio Code

A Visual Studio Code (röviden VS Code) egy nyílt forráskódú, crossplatform szövegszerkesztő és fejlesztői környezet, amelyet a Microsoft fejlesztett ki. Az alkalmazás ingyenesen letölthető és használható Windows, macOS és Linux operációs rendszereken. A VS Code több programozási nyelvet támogat, és számos funkcióval rendelkezik, amelyek megkönnyítik a fejlesztők számára a kódírás folyamatát. Ezek közé tartozik például az intelligens kódkiegészítés, a szintaxis kiemelés, a debuggolás, a Git integráció, a szerveroldali fejlesztést támogató bővítmények. A VS Code rendkívül testre szabható, és lehetőséget biztosít a felhasználóknak, hogy testre szabják az alkalmazást a saját igényeik szerint. Az alkalmazás maga is bővíthető, és lehetőséget kínál a felhasználók számára, hogy telepítsenek különböző bővítményeket és kiterjesztéseket, amelyek további funkciókat adnak az alkalmazáshoz. Azért választottuk. mert az alkalmazás számos szolgáltatást kínál a fejlesztők számára, és lehetővé teszi a hatékony és produktív kódolást az egyszerű kódtól a nagyobb projektekig.

2.2.2. Postman

A Postman egy szoftver, amely lehetővé teszi az API-k (Application Programming Interface) tesztelését, dokumentálását és fejlesztését. Az API-k olyan programozási felületek, amelyek könnyebb kommunikációt biztosítanak az alkalmazások és a szolgáltatások között A Postman egy grafikus felhasználói felületet (GUI) biztosít, segítségével a fejlesztők tesztkéréseket küldhetnek az API-k felé, és ellenőrizhetik a válaszokat. A Postman a HTTP, HTTPS és más protokollok használatára is alkalmas, és támogatja a REST (Representational State Transfer) API-kat és más elterjedt API típusokat. A Postman funkciói közé tartozik a kérések automatizálása, a tesztek és adatok mentése, a dokumentáció és az API-keresések megosztása a csapatokkal

2.3. AZ ADATBÁZIS

Az projektünk lényege a könnyű áttekinthetőség, átláthatóság megvalósítása, ami egy jól megtervezett adatbázisnál kezdődik. Az adatbázis készítésénél az átláthatóságot és dinamikus adatbevitelt figyelembe véve a csapatunk a MySQL adatbázis-kezelő mellett döntött. A MySQL, ahogy a neve is mutatja, egy SQL-alapú adatbázis-kezelő rendszer vagy DBMS(Database Management Systems). A MySQL egy nyílt forráskódú adatbázis kezelő szoftver, amely segít a felhasználóknak adataik tárolásában, elrendezésében és visszakeresésében. Ez egy rendkívül erős és rugalmas program. A nyílt forráskódú szoftvert jelenleg az Oracle, a Java programozási nyelvet is kifejlesztő vállalat tartja karban. A MySQL képes az adatok táblázatokban történő tárolására, kezelésére és megjelenítésére. Amely így egy sorokat és oszlopokat kezelő relációs adatbázissá teszi. A MySQL-ben tábláknak hívják az adatstruktúra felépítését, az alapértelmezett elsődleges kulcs értéke szám.

- Adattípusok:

POSTS tábla:

id: BIGINT
title: VARCHAR(255)
text: TEXT
url: VARCHAR(255)
user_id: BIGINT

USERS tábla:

id: BIGINT
name: VARCHAR(255)
email: VARCHAR(255)
description: TEXT
password: VARCHAR(255) (hash-elve tárolt)
is_admin: TINYINT

Az adatbázist migrációkkal generáltuk laravelben:

```
php artisan make:migration createPostsTable
public function up(): void
    {
        Schema::create('posts', function (Blueprint $table) {
            $table->id();
            $table->string('title');
```

```
            $table->text('text');
            $table->string('url')->nullable();
            $table->foreignId('user_id');
            $table->timestamps();

            $table->foreign('user_id')
                ->references('id')
                ->on('users');
        });
    }
```

Az API kéréseknél a with() segítségével tudtuk lekérni a posztokhoz tartozó felhasználókat, amikhez a modellekben használtuk a hasMany() és belongsTo() függvényeket az adatbáziskapcsolatok létrehozására a táblákhoz tartozó model fájlokban.
Az adatbázisban szereplő táblák adatainak lekéréséhez az elérési utakat hasonló módon csoportosítottuk, a CRUD elvét követik (create, read, update, delete). A felhasználók és a posztok kaptak keresés függvényeket is, így egyszerűbb szűrni közöttük.

```
Route::group(['prefix' => 'users'], function() {
    Route::get("/", [UserController::class, "get"]);
    Route::post("/create", [UserController::class, "create"]);
    Route::put("/update/{user}", [UserController::class, "update"]);
    Route::delete("/delete/{user}", [UserController::class, "delete"]);
    Route::get("/search", [UserController::class, "search"]);
});
```

2.4. FEJLESZTÉSI LETETŐSÉGEK

2.4.1. Üzenetküldés az oldalon

A jövőben szeretnénk lehetőséget, teret szabni a felhasználók egymás között történő, elérhetésére, egymás felkeresésére, felkérésére. Akár egy külön szekció létrehozásával.

2.4.2. Értesítés

A Backend oldalon elő van készítve már az értesítési központ, de még további fejlesztési lehetőségnek számítana, hogy minden egyes értesítést e-mailben is kiküldenénk.

2.4.3. Többnyelvűség

A többnyelvű weboldal lehetővé teszi a cégeknek, hogy elérjék azokat a felhasználókat, akik különböző nyelveken keresnek információkat vagy termékeket szolgáltatásokat. Ezáltal növelhetik a látogatók számát és a potenciális ügyfelek körét. Jelenleg csak angol nyelven érhető el a webalkalmazás, de a jövőben a magyar és a német nyelvel szeretnénk bővíteni a nyelvtárát az oldalnak. Azért lenne pozitív számunkra a többnyelvűség mert egy vállalat vagy márka, amely többnyelvű weboldalt kínál, azt sugallja, hogy globális szemlélettel rendelkezik és elkötelezett az ügyfelek kiszolgálása iránt. Ez növeli a vállalat hitelességét és professzionalizmusát.

2.4.4. Natív alkalmazás

A végső célunk, hogy ne csak böngészőn keresztül lehessen elérni az oldalt, hanem fejlesztenénk egy telefonos és egy asztali alkalmazást, amivel még könnyebben lehetne használni a programunkat.

2.4.5. Oldal design

Maga a design sok változtatáson ment keresztül, de még mindig sok változtatás fér rá. A színkombinációk, a háttér, és a gombok kialakítására is sok változtatás fér még. A célunk egy kellemes érzetű, felhasználóbarát oldal létrehozása.Felhasználói dokumentáció

2.5. Rendszerkövetelmények

Projektmunkánk egy webes felület, ezért csak egy eszközt igényel, amivel az internethez lehet kapcsolódni. Ez lehet laptop, asztali számítógép, telefon vagy akár tablet is, mert a felületet optimalizáltuk az összes méretre. Az eszközön internetelérésnek és a felsorolt böngészők egyikének kell lennie: Google Chrome, Brave, Microsoft Edge, Safari, Firefox.

2.6. Jogosultsági szintek

- Felhasználó:

A felhasználói jogosultságot minden egyes regisztrált felhasználó megkapja. A felhasználó ranggal hozzáadhat saját posztokat és ezeket kezelheti. Ezen kívül a felhasználó a saját adatait módosíthatja.

- Adminisztrátor:

Az adminisztrátori jogosultsággal az admin szinten lévő felhasználó betekintést nyer egy külön erre a szintre kifejlesztett kezelői felületre, amelyen tudja kezelni az egész adatbázist.

2.7. Felhasználói felület

2.7.1. Regisztráció

Amennyiben még nincs felhasználónk az alkalmazásban, akkor regisztrálni kell egyet. A regisztráláshoz 4 mezőt kell kitölteniük a felhasználóknak. A felhasználónevüket, az E-mail címüket, a jelszavukat kell megadni, és a jelszavukat mégegyszer. A regisztrációs oldalt React-ban készítettük, a stílushoz CSS-t használtunk.

- React kódrészlet:

```
import { Col, Container, Form, Row } from "react-bootstrap"
import 'bootstrap/dist/css/bootstrap.min.css'
export default function Register() {
   return(
      <Form className="loginBody">
         <Form className="wrapper"  >
         <h1 className="h1">Register</h1>
   <Form.Group className="input-box" as={Row}  >
      <Col>
      <input type="text" placeholder="Username" required/>
        <i className='bx bxs-user'></i>
      </Col>
   </Form.Group>
   <Form.Group className="input-box" as={Row}  >
```

```
      <Col>
      <input type="text" placeholder="E-mail" required/>
        <i className='bx bxs-user'></i>
      </Col>
    </Form.Group>

    <Form.Group className="input-box" as={Row}  >
      <Col >
        <input type="password" placeholder="Password" required/>
        <i className='bx bxs-locsk-alt'></i>
      </Col>
    </Form.Group>
    <Form.Group className="input-box" as={Row}  >
      <Col >
        <input type="password" placeholder="Password" required/>
        <i className='bx bxs-locsk-alt'></i>
      </Col>
    </Form.Group>
    <button type="submit" className="btn">Register</button>
    <div className="register-link">
    <p>Have an account? <a href={`/login`}>Log in</a></p>
          </div>
     </Form>
      </Form>
    )
}
```

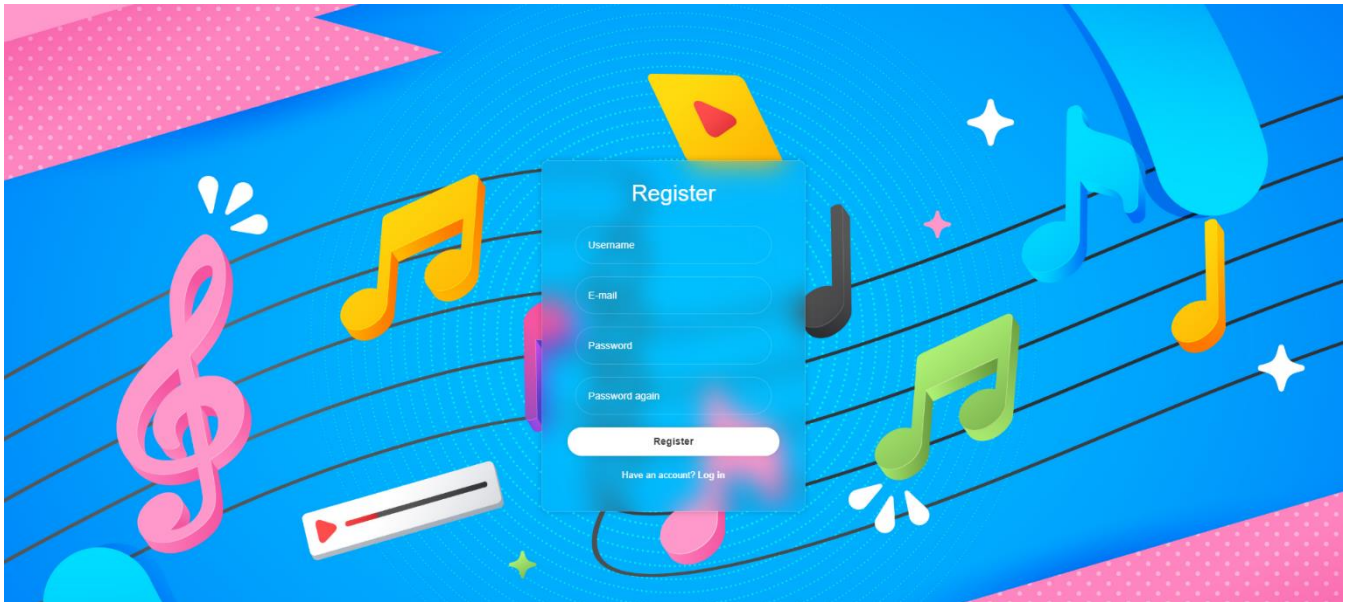
- CSS kódrészlet:

```
.loginBody{
  display: flex;
  justify-content: center;
  align-items: center;
  min-height: 100vh;
  background-image: url('/images/img.jpg');
  background-repeat: no-repeat;
  background-size: cover;
  background-position: center;
}
.wrapper{
  width: 420px;
  background: transparent;
  border: 2px solid rgba(255, 255, 255, .2);
  backdrop-filter: blur(20px) ;
  box-shadow: 0 0 10px rgba(0, 0, 0, .2);
  color: #fff;
```

```
  border-radius: 10px;
  padding: 30px 40px;
}
.wrapper .h1 {
  font-size: 36px;
  text-align: center;
}
.wrapper .input-box{
  position: relative;
  width:100%;
  height: 50px;
  margin: 30px 0;
}
.input-box input{
  width: 100%;
  height: 100%;
  background: transparent;
  border: none;
  outline: none;
  border: 2px solid rgba(255, 255, 255 , .2);
  border-radius: 40px;
  font-size: 16px;
  color: #fff;
  padding:20px 45px 20px 20px;
}
.input-box input::placeholder {
  color: #fff;
}
.input-box i{
  position: absolute;
  right: 20px;
  top: 50%;
  transform: translateY(-50%);
  font-size: 20px;
}
.wrapper .remember-forgot{
  display: flex;
  justify-content: space-between;
  font-size: 14.5px;
  margin:-15px 0 15px ;
}
.wrapper .btn {
  width: 100%;
  height: 45px;
  background: #fff;
  border: none;
  outline: none;
```

```
  border-radius: 40px;
  box-shadow: 0 0 10px rgba(0, 0, 0, .1);
  cursor: pointer;
  font-size: 16px;
  color: #333;
  font-weight: 600;
}
.wrapper .register-link{
  font-size: 14.5px;
  text-align: center;
  margin: 20px 0 15px;
}
.register-link p a {
  color: #fff;
  text-decoration: none;
  font-weight: 600;
}
.register-link p a:hover  {
  text-decoration: underline;
}
```

- A regisztrációs oldal kinézete:



1. ábra: Regisztrációs oldal

2.7.2. Bejelentkezés

A bejelentkezéshez a megfelelő e-mail-cím és jelszó párost kell beírnunk, ezután a Login gombra kattintva máris beléptünk az oldalra. Bármilyen hibás adatot adunk meg, az oldalon egy hibaüzenet jelenik meg. Ezen a felületen láthatjuk a felhasználók posztjait, tevékenységeit. Ezen felül láthatjuk a navigációs részt, ahol az oldalak között válthatunk. A felület jobb felső részén láthatjuk azokat a gombokat, melyek különböző oldalakra vezetnek át. Ott láthatjuk a kijelentkezés/bejelentkezés, főoldal, keresés gombokat, melyek különböző oldalakra irányítanak minket, valamint külön a posztoknak is van helye. Többek között a saját profilunknak is, amelyet tetszésünk szerint lehet szerkeszteni. A Log in oldalt PHP-ban, React-ban készítettük, a stílushoz CSS-t használtunk. A bejelentkezéshez JWT tokent használtunk. JWT (JSON Web Token) egy népszerű módszer az autentikációra és az információk biztonságos átvitelére. Amikor egy felhasználó sikeresen bejelentkezik egy alkalmazásba, a szerver általában egy JWT-t generál, amely tartalmazza a felhasználó azonosítóját és/vagy más információkat. Ezután a JWT-t a kliens oldalra küldik vissza, és az alkalmazás a továbbiakban ezt a tokent használja a felhasználó azonosítására és az engedélyek kezelésére. A bejelentkezéshez használtuk a Laravelben alapértelmezetten elérhető modelleket (pl.: User), a tokeneket a personal_access_tokens táblában tárolja a szerver, ami szintén a keretrendszer beépített funkciója

- React kódrészlet:

```
import { Col, Container, Form, Row } from "react-bootstrap"
import 'bootstrap/dist/css/bootstrap.min.css'
export default function Login() {
   return(

      <Form className="loginBody">
         <Form className="wrapper"  >
         <h1 className="h1">Login</h1>
   <Form.Group className="input-box" as={Row}  >
      <Col>
      <input type="text" placeholder="Username" required/>
      </Col>
   </Form.Group>

   <Form.Group className="input-box" as={Row}  >
      <Col >
        <input type="password" placeholder="Password" required/>
```

```
        <i className='bx bxs-lock-alt'></i>
      </Col>
    </Form.Group>
    <button type="submit" className="btn">Login</button>
          <div className="register-link">
             <p>Don't have an account? <a href={`/register`}>Register</a></p>
          </div>
     </Form>
      </Form>
    )
}
```

- PHP kódrészlet:

```
class AuthController extends Controller
{
    public function register(Request $request) {
        return $user = User::create([
            'name' => $request->input('name'),
            'email' => $request->input('email'),
            'password' => $request->input('password')
        ]);
    }

    public function login(Request $request) {
        if(!Auth::attempt($request->only('email', 'password'))) {
            return response([
                'message' => 'Invalid credentials'
            ], 401);
        }
        $user = Auth::user();
        $token = $user->createToken('token')->plainTextToken;
        $cookie = cookie('jwt', $token, 60 * 24);
        return response([
            'message' => 'success'
        ])->withCookie($cookie);
    }
    public function user() {
        return Auth::user();
    }
    public function logout(Request $request) {
        $cookie = Cookie::forget('jwt');

        return response([
            'message' => 'success'
```

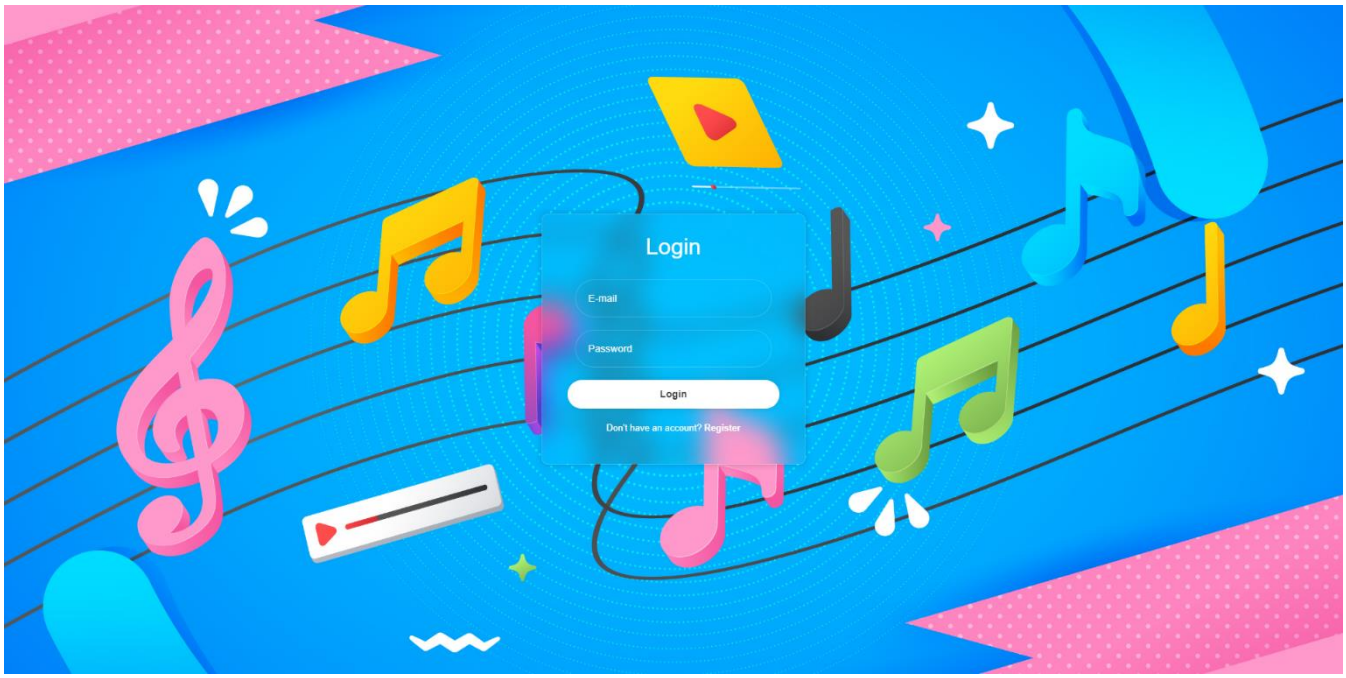
```
        ])->withCookie($cookie);
    }
}
```

- CSS kódrészlet:

```
.loginBody{
  display: flex;
  justify-content: center;
  align-items: center;
  min-height: 100vh;
  background-image: url('/images/img.jpg');
  background-repeat: no-repeat;
  background-size: cover;
  background-position: center;
}
.wrapper{
  width: 420px;
  background: transparent;
  border: 2px solid rgba(255, 255, 255, .2);
  backdrop-filter: blur(20px) ;
  box-shadow: 0 0 10px rgba(0, 0, 0, .2);
  color: #fff;
  border-radius: 10px;
  padding: 30px 40px;
}
.wrapper .h1 {
  font-size: 36px;
  text-align: center;
}
.wrapper .input-box{
  position: relative;
  width:100%;
  height: 50px;
  margin: 30px 0;
}
.input-box input{
  width: 100%;
  height: 100%;
  background: transparent;
  border: none;
  outline: none;
  border: 2px solid rgba(255, 255, 255 , .2);
  border-radius: 40px;
  font-size: 16px;
```

```
  color: #fff;
  padding:20px 45px 20px 20px;
}
.input-box input::placeholder {
  color: #fff;
}
.input-box i{
  position: absolute;
  right: 20px;
  top: 50%;
  transform: translateY(-50%);
  font-size: 20px;
}
.wrapper .remember-forgot{
  display: flex;
  justify-content: space-between;
  font-size: 14.5px;
  margin:-15px 0 15px ;
}
.wrapper .btn {
  width: 100%;
  height: 45px;
  background: #fff;
  border: none;
  outline: none;
  border-radius: 40px;
  box-shadow: 0 0 10px rgba(0, 0, 0, .1);
  cursor: pointer;
  font-size: 16px;
  color: #333;
  font-weight: 600;
}
.wrapper .register-link{
  font-size: 14.5px;
  text-align: center;
  margin: 20px 0 15px;
}
.register-link p a {
  color: #fff;
  text-decoration: none;
  font-weight: 600;
}
.register-link p a:hover  {
  text-decoration: underline;
```

- A bejelentkezés kinézete:



2. ábra: Bejelentkezési oldal

2.7.3. Főoldal

A sikeres bejelentkezés után a rendszer a főoldalra irányít minket. Ezen az oldalon láthatjuk a posztokat, amiket le lehet játszani. Jobb felül találhatók a keresés, posztok, profil, login/logout gombok.

- React kódrészlet

```
export default function Home() {
   const [data, setData] = useState([]);

   useEffect(() => {
      const fetchData = async () => {
         const result = await axios.get("http://localhost/project_1/server/public/api/posts");
         setData(result.data);
      };
      fetchData();
   }, []);

   return(
      <div className="background">
         <title>NOTEHOUSE - Posts</title>
         <Container>
```

```
            <Menu/>
            <Col>
               <Button href='/posts/create'>Create Post</Button>
               <div className="cardouter">
                  {data.map(post => (
                     <PostCard key={post.id}
                        title={post.title}
                        text={post.text}
                        url={post.url}
                        username={post.user.name}
                     />
                  ))}
               </div>
            </Col>
         </Container>
      </div>
   )
}
```

2.7.3.1. Poszt készítés:

- React kódrészlet:

```
export default function CreateForm() {
   return (
      <div className="background">
         <Container>
         <Menu/>
            <Form onSubmit={(event) => {
               event.preventDefault();

               const form = event.target;

               axios.post("http://localhost/project_1/server/public/api/posts/create", {
                  title: form.title.value,
                  text: form.text.value,
                  url: form.url.value
               }).then((res) => {
                  alert("Posted successfully!");
               }).catch((error) => {
                  alert("Error");
               })
            }}>
```

```
            <Card>
              <Container className="createcard">
                <Form.Group className="mb-3">
                  <Form.Label>Title</Form.Label>
                  <Form.Control type="text" name="title"/>
                </Form.Group>
                <Form.Group className="mb-3">
                  <Form.Label>Text</Form.Label>
                  <Form.Control className="textinput" as="textarea"
name="text"/>
                </Form.Group>
                <Form.Group className="mb-3">
                  <Form.Label>URL</Form.Label>
                  <Form.Control type="text" name="url"/>
                </Form.Group>
                <Form.Group className="mb-3">
                  <Button className="createbutton buttonwrapper"
type="submit">Post</Button>
                </Form.Group>
              </Container>
            </Card>
          </Form>
        </Container>
      </div>
    )
}
```

2.7.3.2. Navbar:

Egy "navbar" egy olyan rész az egy weboldalon vagy alkalmazáson belül, amely általában egy vízszintes sáv vagy menü, és tartalmazza a weboldal vagy alkalmazás legfontosabb linkeit vagy funkcióit. A navbar gyakran a fejléc része, és lehetővé teszi a felhasználók számára, hogy könnyen navigáljanak az oldalon vagy az alkalmazásban.

A navbar többféleképpen segítheti az oldalt:

Navigáció: A legfontosabb linkek vagy funkciók könnyen elérhetővé válnak a felhasználók számára a navbaron keresztül. Ez lehetővé teszi, hogy a felhasználók gyorsan eljuthassanak a kívánt helyre az oldalon.

Áttekinthetőség: A navbar lehetővé teszi az oldal struktúrájának és a rendelkezésre álló lehetőségeknek az áttekintését. Segít megérteni, hogy milyen szolgáltatások vagy információk érhetők el az oldalon.

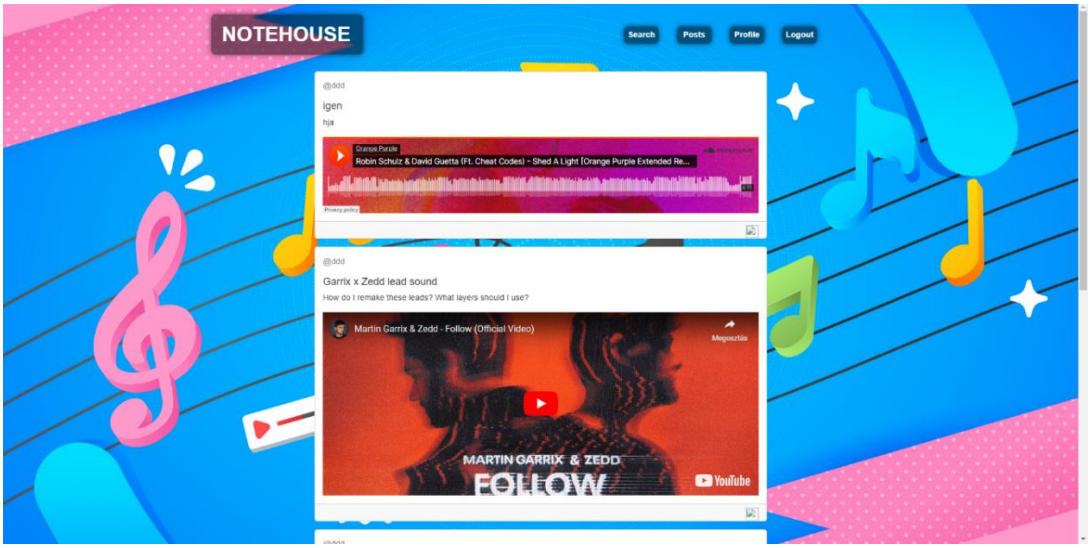
Felhasználói élmény javítása: A megfelelően kialakított navbar segíthet a felhasználók számára az oldalon való tájékozódásban és a keresett tartalom gyors megtalálásában, ezáltal javítva a felhasználói élményt. Összességében a navbar fontos elem egy weboldalon vagy alkalmazásban, mivel segít a felhasználóknak navigálni és könnyen megtalálni a keresett információkat vagy funkciókat.

React kódrészlet:

```
export default function Menu() {
   const router = useRouter();

   return(
      <>
         <br/>
         <h1 className="pagetitle
titlewrapper"><b>NOTEHOUSE</b></h1>
         <div className="navbuttons">

            <Nav className="navBar">

               <Nav.Item className="button">
                  <Button type="submit"
href={"/search"}><b>Search</b></Button>
               </Nav.Item>
               <Nav.Item className="button">
                  <Button href={`/`}><b>Posts</b></Button>
```

```
                </Nav.Item>
                <Nav.Item className="button">
                   <Button color="#FFFFFF"
href={`/profile`}><b>Profile</b></Button>
                </Nav.Item>
                <Button color="#FFFFFF" className="button"
onClick={(event) => {

axios.post("http://localhost/project_1/server/public/api/logout").then
((res) => {
                      router.push("/login")
                   }).catch((error) => {
                      alert(error.message)
                   })
                }}><b>Logout</b></Button>
             </Nav>
          </div>
          <br/>
       </>
    )
}
```

- A főoldal kinézete:



3. ábra: Főoldal

2.7.4. Profil

A profil gomb elérhető lesz a főoldalon. Itt a felhasználó tudja szerkeszteni az adatait. A felhasználó tud önmagáról információkat megosztani. Le tudja írni a zenei érdeklődési köreit, korábbi tapasztalatait.

- React kódrészlet:

```
export default function Profile() {
    const [data, setData] = useState([]);

    useEffect(() => {
        const fetchData = async () => {
            const result = await
axios.get("http://localhost/project_1/server/public/api/user");
            setData(result.data);
        };
        fetchData();
    }, []);

    return(
        <Container>
```

```
            <Menu/>
            <ProfilePage
               username={data.name}
               email={data.email}
               description={data.description}
            />
         </Container>
      )
}
```

2.7.5. Posztok

A posztrendszer a weboldalunkon úgy működik, hogy linkeken keresztül lehet elérni a különböző feltöltéseket. A felhasználó tud saját posztokat feltölteni az oldalra, valamint, ha posztolni nem is szeretne, akkor tud csak más felhasználók által feltöltött posztokat végig lapozni.

- React kódrészlet:

```
export default function PostCard({
   title,
   text,
   url,
   username
} : {
   title: string,
   text: string,
   url: string,
   username: string
}) {
   return(
      <Container>
         <Card className="card mb-3">
            <Card.Body>
               <Card.Text><a                                    className="username"
href={`/profile/${username}`}>@{username}</a></Card.Text>
               <Card.Title>{title}</Card.Title>
               <Card.Text>{text}</Card.Text>
               <div className="url">
                  {URLRender()}
               </div>
```

```
            </Card.Body>
            <Card.Footer className="foot">
              <Card.Text>
              </Card.Text>
              {ShareRender()}
            </Card.Footer>
          </Card>
        </Container>
    )
    function URLRender() {
        if (url.includes("soundcloud.com/")) {
            return <ReactPlayer url={url} width={900} height={150}/>
        } else if (url === null || url === "") {
            return null
        } else return <ReactPlayer url={url} width={900} />
    }
    function ShareRender() {
        if (url === "" || url === null) {
            return <></>
        } else return <a href={url} className="openlink" color="rgb(0, 184, 107)"
target="_blank"><Button>Open Link</Button></a>
    }
}
```

2.7.6. Keresősáv:

Az oldalunkon található keresősáv azért jött létre, hogy az oldal felhasználói könnyen rá tudjanak keresni arra a műfajra, akár előadóra, producerre akitől szívesen hallgatnának valamit, illetve ha csak böngészni szeretnének a profilján, akkor lehetőségük nyílik a feltöltő posztjai közötti böngészésre.

- React kódrészlet:

```
export default function SearchPage() {
    const [data, setData] = useState('');

    function Data(adat: string) {
        useEffect(() => {
            const fetchData = async () => {
                const result = await
axios.get(`http://localhost/project_1/server/public/api/posts/search`, {
                    title: adat
                });
```

```
            setData(result.data);
        };
        fetchData();
    }, []);
  }
  return(
    <Form className="searchform" onSubmit={(event) => {
        const form = event.target;

        Data(form.searchbar.value);
    }}>
        <Row>
            <Col className="col-3"/>
            <Col className="col-5">
                <Form.Group>
                    <Form.Control type="text" name="searchbar"></Form.Control>
                </Form.Group>
            </Col>
            <Col className="col-1">
                <Button type="submit">Search</Button>
            </Col>
            <Col className="col-3"/>
        </Row>
    </Form>
  )
}
```

3. Tesztelés

Az API útvonalak teszteléséhez Postmant használtunk. A Postman egy sokoldalú és könnyen használható eszköz, amely segít az API-k tesztelésében, dokumentálásában és automatizálásában. Ideális fejlesztők, tesztelők és DevOps mérnökök számára. Különféle HTTP kéréseket küldhetünk a szervernek (get, post, delete, put).

4. Összefoglalás

A program megírása során rengeteget fejlődtünk. Nagyon sok új készség alakult ki bennünk, a legfontosabb a csapatmunka és a munkamegosztás. Természetesen az elején sok hibát követtünk el az alkalmazás megtervezésében, de a projektfelosztással ezeket közösen hamar javítottuk. A program elkészítése során rengeteg szakmai tapasztalattal bővült a tudásunk, és megismerkedtünk egy komplex szoftver megírásának elemeivel, annak minden nehézségével együtt. Szakmai ismereteink legfőképpen a webfejlesztés irányába növekedtek. Szerencsénk volt megismerkedni a ReactJs-el, ami hatalmas szerepet töltött be a projektünk megalkotásához, hiszen rengeteget dolgoztunk a React keretein belül. Ugyanez igaz a NodeJs-re, valamint a Laravel-re is, hiszen amikor elkezdtük a projektünk tervezését még nem igazán voltunk biztosak magunkban, nem ismertük még ezeket a keretrendszereket. Mára ez többnyire változott. A projektünk megalkotása közben hol több, hol kevesebb vitával, de annál is több kitartással és elkötelezettséggel dolgoztunk. Így juthattunk el erre a pontra, hogy azt mondhatjuk, hogy megalkottuk ezt a weboldalt. A közös munkát összességében nagyon élveztük, általában közös hullámhosszon voltunk így könnyen ment a munka, viszont néha volt pár vitánk, hogy hogyan is haladjunk tovább hiszen mindhármunknak különböző ötletei voltak, de ezeket megbeszéltük, átléptünk rajtuk és mentünk tovább. Van pár funkció, amit nem tudtunk belerakni a projektbe, de a jövőben szeretnénk ezen változtatni és további fejlesztésekkel bővíteni az oldalt, valamint újdonságokkal felruházni, hogy minél több felhasználó találjon komfortot az oldal használata közben.

5. Irodalomjegyzék

1. **TypeScript**: https://prog.hu/szotar/TypeScript Letöltés ideje: 2024.04.16
2. **Php**: https://webiskola.hu/php-ismeretek/mi-az-a-php-fogalma-bemutatasa/ Letöltés ideje: 2024.03.1
3. **GitHub**: https://kinsta.com/knowledgebase/what-is-github/ (magyarra fordítva) Letöltés ideje: 2024.04.21
4. **React**: https://www.w3schools.com/whatis/whatis_react.asp (magyarra fordítva) Letöltés ideje: 2023.10.13 16.
5. **NodeJS**: https://nodejs.org/en/blog/announcements/foundation-v4-announce/ (magyarra fordítva) Letöltés ideje: 2023.11.23
6. **Postman**: https://www.javatpoint.com/postman (magyarra fordítva) Letöltés ideje: 2024.01.09
7. **Visual Studio Code**: https://code.visualstudio.com/docs/editor/whyvscode (magyarra fordítva) Letöltés ideje: 2023.12.20

6. Ábrajegyzék