Szegedi Tudományegyetem

Informatikai Tanszékcsoport

Webalkalmazás Zene- és Podcast Streaming, Közösségi Funkciók és Személyre Szabott Ajánlások Szolgáltatásával

Szakdolgozat

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Készítette: |  | Témavezető: |  |
|  | Kovács Tamás |  | Hegedűs Péter |  |
|  | programtervező informatikus szakos hallgató |  | doktorandusz |  |

Szeged

2025

Feladatkiírás

A szakdolgozat célja egy olyan közösségi webalkalmazás készítése, ami a zenét, illetve podcast hallgatok igényeit elégíti ki. Az alkalmazás funkciói kiterjedjenek a közösségi portálok funkcionalitásaira, úgymint a regisztráció, üzenetküldés, felhasználó keresés. Ezen felül a zenét, illetve podcast hallgatók számára fontos funkciókat is tartalmazza, mint például hangfájl feltöltése, lejátszási listák létrehozása, audió letöltése, fórum, audió kereső. A webalkalmazás az interneten legyen elérhető akár online, illetve offline állapotban, megtekintéséhez csupán egy webböngésző legyen szükséges.

# Bevezetés

Az elmúlt években a digitális zenehallgatás és podcast-fogyasztás mindennapjaink szerves részévé vált, amiben az internet kiemelkedő szerepet játszott. Míg korábban fizikai adathordozókon vagy rádión keresztül hallgattuk kedvenceinket, ma már néhány kattintással elérheti.

Az internet nyújtotta lehetőségeket kihasználva egyre több előadó, podcaster, kiadó és független alkotó osztja meg műveit különböző platformokon. Ezek között, azonban könnyű elveszni, ezért jött létre a ZeeSharing, amely célja, hogy a felhasználók egy helyen, kényelmesen böngészhessenek a zenék és podcastek között, felfedezve új kedvenceket és visszatérve a már jól ismert tartalmakhoz.

A ZeeSharing elsődleges célja, hogy a felhasználó könnyen elérhesse az éppen trendi, népszerű vagy frissen megjelent zenéket, podcast epizódokat. A lejátszás mellett lehetőség van lejátszási listák létrehozására, követésére, sőt akár offline letöltésre is. Így a felhasználó bármikor visszatérhetnek a korábban elmentett tartalmakhoz anélkül, hogy internethez lennének csatlakozva.

A böngészés cím, kategória, előadó, podcaster alapján keresi meg a kívánt tartalmat, ami nagyban megkönnyíti a tartalom elérését. A regisztrált felhasználók személyes oldalukon nemcsak kedvenceiket követhetik nyomon, hanem lejátszási listákat hozhatnak létre, amit megoszthatnak más felhasználókkal is. Illetve a felhasználó nyomon követheti az eddig meghallgatott zenéket, illetve podcasteket grafikonok segítségével.

A ZeeSharing másik célja egy aktív közösség kiépítése is. Az előadok, valamint podcasterek a fórumon bejelenthetik új albumjaikat, illetve közelgő eseményeiket is. Lehetőség van egymást követni, illetve ismerősnek jelölni, privát üzeneteket küldeni, tartalmat ajánlani, és egymás lejátszási listáit is megtekinteni, ha azok nyilvánosak. A kapcsolati rendszer átlátható: követések, visszaigazolások, függő kérések külön szekcióban láthatóak.

Az előadók és podcastére számára külön kezelőfelület áll rendelkezésre, ahol új tartalmakat tölthetnek fel, illetve kiválasztják a feltölteni kíván elem kategóriáját is, illetve joguk van a fórum oldalra bejegyzéseket írni. Az adminisztrátorok külön felületen tudnak felhasználókat kezelni, illetve fórum bejegyzéseket törölni.

Összefoglalva, a ZeeSharing egy olyan modern, közösségi alapokra épülő zene- és podcast platform, amely nemcsak a hallgatás élményét biztosítja, hanem lehetőséget nyújt az alkotók és hallgatók közötti aktív, inspiráló kapcsolatteremtésre is.

# 1. Technológiák bemutatása

## 1.1. Programnyelvek, technológiák

### 1.1.1. Angular

Az Angular egy modern, nyílt forráskódú webalkalmazás-fejlesztési keretrendszer, amelyet eredetileg a Google fejlesztett ki, és 2016-ban jelent meg az első stabil változata Angular 2 néven. Bár a fejlesztés kezdete az AngularJS-hez köthető, az új Angular gyökeresen eltér elődjétől: TypeScript nyelven íródott, és a legmodernebb komponensalapó architektúrát követi. A keretrendszert a Google folyamatosan frissíti és karbantartja, így biztosítva a hosszú távú támogatást, valamint a technológiai fejlődéssel való lépéstartást.

Az Angular célja, hogy egységes megközelítést kínáljon az összetett, nagy volumenű frontend alkalmazások fejlesztésére. Már a kezdetektől beépített eszköztárat biztosít a komponensalapú struktúrához, űrlapkezeléshez HTTP-kommunikációhoz, oldalnavigációhoz, teszteléshez és dependency injection-höz – vagyis olyan funkciókhoz, amelyek más keretrendszerekben csak külső osztálytárakkal oldhatók meg.

Az Angular választása számomra egyértelmű volt, mert professzionális szintű, jól dokumentált, és hosszú távon is fenntartható keretrendszert kínál. A hivatalos dokumentáció minden fogalomhoz világos példákat és részletes leírást biztosít. Emellett az aktív fejlesztői közösség folyamatosan bővíti a tudásbázist fórumokon, blogokon és GitHubon keresztűl.

Az Angular teljesen platformfüggetlen, így asztali, mobil és webes alkalmazások készítésére egyaránt alkalmas. Kiválóan együttműködik a Firebase-szel, REST API-kkal, és bármilyen backend technológiával. A szintaxisa egységes, TypeScript alapú, ami segíti az automatikus típusellenőrzést, ezáltal csökkenti a hibák számát és növeli a kód megbízhatóságát. A projekt buildelése és futtatása egy egyszerű böngészővel megoldható, fordítás csak a fejlesztés során szükséges. Támogatja a PWA (Progresszív Webalkalmazás) fejlesztést, offline használatot, és lehetőséget kínál natív asztali alkalmazás készítésére Electronnal.

### 1.1.2. CSS

A weboldal megjelenítéséhez a CSS (Cascading Style Sheets) stílusleíró nyelvet használtam, amely lehetővé teszi a tartalom és a forma szétválasztását. Különös figyelmet fordítottam a színek összehangjára, az elemek rendezett elhelyezésére, valamint az űrlapok és navigációs elemek esztétikus kialakítására. A stílusokat külső CSS fájlokba szerveztem, külön az egyes felületek számára, így a módosításokat egyszerűen, központilag lehetett elvégezni. A reszponzív megjelenítéssel biztosítottam, hogy a weboldal különböző eszközökön – mobilon, tableten, és asztali gépen – is megfelelően jelenjen meg. A modern CSS lehetőségei, mint a flexbox és az animációk, tovább javították a felhasználói élményt és a vizuális megjelenítést.

### 1.1.3. Firebase és Firebase Storage

A Firebase a Google által fejlesztett, felhőalapú fejlesztői platform, amely modern webes és mobilalkalmazások háttérrendszerének kialakításához nyújt komplett megoldást. Számos szolgáltatást biztosít, például valós idejű adatbázist, felhasználói hitelesítést, fájltárolást, üzenetküldést és analitikát, ezzel jelentősen megkönnyítve a fejlesztők munkáját. A Firebase erőssége a skálázhatóság, a biztonságos és gyors működés, valamint az, hogy a szolgáltatások szorosan integrálódnak egymással és könnyedén használhatók modern frontend keretrendszerekkel, mint például az Angular.

A projekt során elsősorban két Firebase szolgáltatást alkalmaztam: a Cloud Firestore NoSQL adatbázist és a Firebase Storage fájltárolót.

A Firestore egy dokumentum alapú adatbázis, amely lehetővé teszi strukturált adatok valós idejű tárolását és lekérdezését. Az adatokat gyűjteményekbe és dokumentumokba szervezve tárolja, ahol minden dokumentum kulcs-érték párokat tartalmaz. Előnye, hogy valós idejű szinkronizációt biztosít. Ez különösen hasznos volt a projekt dinamikus részeinél, például a fórum vagy a chat funkciók esetén. Emellett lehetőség van összetett lekérdezésekre, rendezésre, szűrésre, illetve egyéni jogosultsági szabályok beállítására is.

A Firebase Storage egy biztonságos, Google Cloud Storage alapú fájltároló megoldás, amelyet képek, zenék, videók és egyéb fájlok hatékony kezelésére fejlesztettek ki. A projekt során ezt a szolgáltatást használtam a felhasználók által feltöltött médiafájlok tárolására. A Storage SDK segítségével egyszerűen lehet feltölteni, letölteni vagy törölni a fájlokat közvetlenül a kliensoldali alkalmazásból. A két szolgáltatás közötti integráció lehetővé tette, hogy a feltöltött fájlok elérési útját elmentsem az adatbázisba, így az alkalmazás részein könnyedén hivatkozhattam ezekre.

### 1.1.4. IndexedDB

Az IndexedDB egy beépített, kliensoldali adatbázis, amelyet a modern böngészők biztosítanak, és lehetővé teszi nagy mennyiségű strukturált adat tárolását offline módban is. Az IndexedDB segítségével objektumalapú kulcs-érték párokat menthetünk, ami ideálissá teszi összetettebb webalkalmazások számára, ahol a gyors elérés és az offline működés is elvárás. Az adatok aszinkron módon kerülnek elérésre, így nem blokkolják a fő szál működését, ami különösen fontos felhasználói élmény szempontjából.

A technológiát elsősorban olyan funkciókhoz használtam, ahol fontos volt a gyors helyi adattárolás és elérés, például letöltött médiafájlok, kezelése során. Az IndexedDB használatával a felhasználó akkor is hozzáférhet a korábban letöltött adatokhoz, ha éppen nincs internetkapcsolat. A strukturált tárolás és a tranzakcióalapú működés biztonságos adatkezelést tesz lehetővé. Támogatása széles körű, minden modern böngészőben elérhető, így PWA (Progresszív Webalkalmazás) fejlesztés során is nélkülözhetetlen eszközként szolgál.

## 1.2. Fejlesztői környezet

Fejlesztés és tesztelés során fontos, hogy mindig a számunkra megfelelő eszközzel dolgozzunk.

Fejlesztői környezet:

* Visual Studio Code 1.100

A projekt fejlesztéséhez a Microsoft által fejlesztett Visual Studio Code nevű, platformfüggetlen, nyílt forráskódú szerkesztőt használtam. A program kiválóan alkalmas webes fejlesztésre, mivel rendkívül rugalmas, bővíthető, és számos hasznos funkciót kínál, mint például az automatikus kódkiegészítés (IntelliSense), beépített terminál, Git integráció és szintaktikai kiemelés különböző nyelvekhez, köztük HTML, CSS, JavaScript, TypeScript, Angular, PHP és JSON formátumhoz. A fejlesztés során nagy előnyt jelentett, hogy könnyen kezelhettem benne a teljes projektstruktúrát, és az Angular alkalmazás moduláris felépítését átlátható módon tudtam kezelni.

Teszteléshez:

* Google Chrome 136
* Mozilla Firefox 138
* Microsoft Edge 136

A fejlesztés során elengedhetetlen volt az alkalmazás több böngészőben történő tesztelése, hogy biztosítsam a kompatibilitást és egységes megjelenést. A Chrome-hoz elérhető Angular DevTools segített az egyes komponensek és adatok valós idejű elemzésében. A különböző böngészők beépített fejlesztői eszközeivel (DevTools) lehetőségem volt JavaScript és TypeScript hibák nyomkövetésére, CSS vizsgálatra, hálózati forgalom elemzésére, valamint a reszponzív nézetek tesztelésére is.

Egyéb eszközök:

* GIMP 3: saját logó szerkesztése
* Firebase console: az adatbázis és tárolók kezelésére
* Git + GitHub: verziókezeléshez és távoli tároláshoz

# 2. Specifikáció

Évek óta foglalkozom webes alkalmazások fejlesztésével hobbiként, és régóta megfogalmazódott bennem az igény egy olyan komplex közösségi platform létrehozására, amely a zenehallgatás élményét ötvözi a közösségi funkciókkal. Tapasztalataim alapján úgy gondolom, hogy a legfontosabb követelmény egy ilyen alkalmazással szemben az, hogy gyors, megbízható és felhasználóbarát legyen, emellett pedig kínáljon olyan egyedi szolgáltatásokat, amelyek miatt a felhasználók rendszeresen visszatérnek, és aktívan használják azt. Ennek megfelelően részletesen megterveztem azokat a funkciókat, amelyeket egy modern zenelejátszó és közösségi platformnak nyújtania kell.