

# **DOKUMENTACIJA**

**Zenith – Music Library and Player API**

Web programiranje ASP

Ime i prezime:

**Nikola Đunisić 15/21**

## Sadržaj

Tema projekta.....	3
Dijagram baze podataka .....	4
Controllers .....	5
AlbumController .....	5
Ostali Lookup Controllers .....	6
AuthController .....	6
FilesController .....	6
PlaylistsController .....	7
TrackLikesController .....	7
TracksController .....	8
UsersController .....	8
UseCaseLogsController .....	9
AccessUseCasesController .....	9

# Tema projekta

- Ukoliko za admin korisnika nema neki od useCases, samo je potrebno u bazi dodati Id za tog korisnika, da bi mogao da izvrši

- PRE POKRETANJA PROJEKTA U wwwroot folderu, napraviti folder sa nazivom temp, ukoliko ne postoji.

- Ukoliko settings iz appSettings development nece da se Binduju, prebaciti settings u obican appSettings fajl ili namestiti sistemske promenljive.

- **Zenith API** je Music Library i Player projekat, gde korisnik može videti sve pesme u sistemu, može ih filtrirati i pretražiti po kriterijumima, može ih lajkovati i dodati u svoju plejlistu i praviti nove plejliste. Takođe, “administrator” ima mogućnosti svih CRUD operacija nad svim tabelama, što znači da može dodavati, menjati, brisati pesme i sve ostale entitete u sistemu.

- Koriscena je **Clean Architecture** sa 5 projekata unutar solutiona:

- Domain, DataAccess, Application, Implementation, API

# Dijagram baze podataka



# Controllers

- Mogućnost izvršavanja metoda iz controllera je definisano po jediničnom korisniku, što znači da nema klasičnih uloga, već se useCases koje korisnik može da izvršava dodeljuje posebno svakom korisniku. Za to je odgovorna tabela UserUseCase.
- Za svaki endpoint gde se vraćaju svi podaci postoji paginacija. Kroz Query Params se može proslediti Page I PerPage, da bi se prikazala odgovarajuća stranica I odgovarajući broj objekata.
- Takođe, za svaki endpoint gde se vraćaju svi podaci se koristi AutoMapper, da bi se mapirali podaci iz Domain klase u DTO klasu za prenos podataka.

## AlbumController

- Postoje validatori za svaku situaciju kada se šalju podaci sa Fronta.

### ENDPOINTS:

- GET /api/albums - Search za albums I dohvaćanje svih albuma, može se poslati Keyword kroz Query parametars da bi se pretražilo po nazivu albuma
- POST /api/albums - Dodavanje novog albuma, šalje se samo Name kroz body
- PUT /api/albums/{id} - Update postojećeg albuma, šalje se id u URL I Name kroz body, koji će se zameniti
- DELETE /api/albums/{id} - Soft delete postojećeg albuma, šalje se samo id kroz URL

## **Ostali Lookup Controllers**

- Isti su ovakvi controlleri i endpoints za sve ostale lookup tabele, samo umesto albums bi išlo nesto od ovih:

- artists
- fileTypes
- genres
- mediaTypes

- Takođe postoje Validatori za svaku od njih

## **AuthController**

### **ENDPOINTS:**

- POST /api/auth - Login, šalje se email i password, sistem proverava da li postoji takav korisnik i vraća token, koji se kasnije koristi za pristup useCases, token traje 600 sekundi i postavlja se u niz validnih tokena
- DELETE /api/auth - Logout, invalidira se postojeći token

## **FilesController**

- Postoje validatori za svaku situaciju kada se šalju podaci sa Fronta.

### **ENDPOINTS:**

- POST /api/files - Inicijalni upload fajla, kada korisnik na frontu uploaduje fajl i postavlja se u temp folder, odakle će se posle prebaciti u trajni folder. Sa fronta se kroz form-data šalje file za image i za audio pesme, jer pesma ima fajl jedne inicijalne slike i ima fajl same pesme.

- GET /api/files/{fileName} - Provera da li fajl postoji, salje se naziv fajla sa fronta.

## **PlaylistsController**

- Postoje validatori za svaku situaciju kada se salju podaci sa Fronta.

### **ENDPOINTS:**

- GET /api/playlists - Endpoint za dohvaćanje svih plejlista koje postoje u sistemu, zajedno sa pretragom po Name.
- GET /api/playlists/mine - Endpoint za dohvaćanje svih plejlista ulogovanog korisnika.
- POST /api/playlists - Endpoint za dodavanje nove plejliste. Salje se Name sa Fronta.
- PUT /api/playlists/{id} - Endpoint za update plejliste. Salje se Id kroz URL I Name.
- DELETE /api/playlists/{id} - Endpoint za Soft delete plejliste. Salje se id kroz URL.
- POST /api/playlists/{id}/track - Dodavanje Track u Plejlistu. Id koji se salje kroz URL je vezan za Plejlistu, a kroz Body se salje TrackId.
- DELETE /api/playlists/{id}/track - Brisanje Track iz Plejliste. Id koji se salje kroz URL je vezan za Plejlistu, a kroz Body se salje TrackId.

## **TrackLikesController**

- Postoje validatori za svaku situaciju kada se salju podaci sa Fronta.

### **ENDPOINTS:**

- POST /api/trackLikes - Endpoint za lajkovanje pesme od strane korisnika. Kroz body se salje TrackId pesme koja treba biti lajkovana.

- DELETE /api/trackLikes/{id} - Endpoint za brisanje lajka sa pesme za ulogovanog korisnika. TrackId se dobija kroz URL.

## TracksController

- Postoje validatori za svaku situaciju kada se salju podaci sa Fronta, za svaki DTO.

### ENDPOINTS:

- GET /api/tracks - Endpoint za dohvatanje svih dostupnih pesama. Postoji filtriranje, sortiranje i paginacija po razlicitim kriterijumima, koji se salju kroz Query Params sa Fronta.
- POST /api/tracks - Endpoint za dodavanje nove pesme. Kroz body se salje Name, Description, Duration, Price, trackFiles.ImagePath, trackFiles.SongPath, ArtistId, AlbumId, GenreId, MediaTypeId
- PUT /api/tracks/{id} - Endpoint za izmenu postojece pesme. Kroz url se salje Id pesme i ostala polja kroz body.
- DELETE /api/tracks/{id} - Endpoint za Soft delete pesme. Kroz url se salje id pesme.

## UsersController

- Postoje validatori za svaku situaciju kada se salju podaci sa Fronta, za svaki DTO.

### ENDPOINTS:

- GET /api/users- Endpoint za dohvatanje svih dostupnih usera. Postoji filtriranje i paginacija po razlicitim kriterijumima, koji se salju kroz Query Params sa Fronta.
- POST /api/users- Endpoint za registrovanje usera. Kroz body se salje Username, FirstName, LastName, Email, Password



- PUT /api/users/{id} - Endpoint za izmenu postojećeg usera. Kroz url se šalje Id usera I ostala polja kroz body.
- DELETE /api/users/{id} - Endpoint za Soft delete usera. Kroz url se šalje id usera.

## **UseCaseLogsController**

### **ENDPOINTS:**

- GET /api/useCaseLogs - Endpoint za dohvaćanje UseCaseLogs. Postoji filtriranje, sortiranje i paginacija po različitim kriterijumima, koji se šalju kroz Query Params sa Fronta.

## **AccessUseCasesController**

- Postoji validator za sve podatke sa Fronta koji bi mogli biti neispravni.

### **ENDPOINTS:**

- POST /api/accessUseCases - Endpoint za postavljanje dozvola za use cases, odnosno koji user sme da izvršava koji use case. Kroz body se šalje UserId I niz UseCaseIds (koji se useCases dozvoljavaju tom korisniku).