

**KAUNO TECHNOLOGIJOS UNIVERSITETAS**

**INFORMATIKOS FAKULTETAS**

**SAITYNO TAIKOMŲJŲ PROGRAMŲ PROJEKTAVIMAS**

**Projekto Ataskaita**

**Atliko:**

IFF-4/3 grupės studentas

Žilvinas Abromavičius

**Priėmė:**

dėst. P. Tamošiūnas

dėst T. Blažauskas

**KAUNAS, 2017**

Turinys

[1. Sprendžiamo uždavinio aprašymas 3](#_Toc500964424)

[1.1. Sistemos paskirtis 3](#_Toc500964425)

[1.2. Funkciniai reikalavimai 3](#_Toc500964426)

[2. Sistemos architektūra 3](#_Toc500964427)

[3. Vartotojo sąsajos projektas 4](#_Toc500964428)

[4. API Specifikacija ir panaudojimo pavyzdžiai 11](#_Toc500964429)

[4.1. Events 12](#_Toc500964430)

[4.2. Files 13](#_Toc500964431)

[4.3. Genres 15](#_Toc500964432)

[4.4. Likes 16](#_Toc500964433)

[4.5. Playlists 19](#_Toc500964434)

[4.6. Songs 27](#_Toc500964435)

[4.7. Users 31](#_Toc500964436)

[5. Išvados 37](#_Toc500964437)

# Sprendžiamo uždavinio aprašymas

# Sistemos paskirtis

Kursinio projekto tikslas – sukurti muzikos klausimo ir talpinimo platformą.

Veikimo principas – panašus kaip kitų muzikos platformų (Soundcloud, Mixcloud). Registruoti vartotojai gauna galimybę kelti savo muziką, klausyti kitų platformoje įkeltos muzikos, kurti grojaraščius, pamėgti tam tikrus muzikos kūrinius.

# Funkciniai reikalavimai

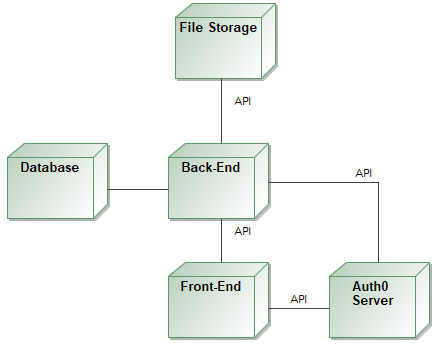
Žemiau nurodytas funkcinių reikalavimų sąrašas:

* Bet kokių platformoje esančių muzikos kūrinių klausymas.
* Muzikos paieška
* Galimybė pamėgti muzikos kūrinius, peržiūrėti pamėgtus kūrinius, klausytis jų kaip grojaraščio.
* Grojaraščių klausymas, kūrimas, redagavimas, šalinimas. Galimybė juos padaryti viešus, privačius.
* Galimybė sekti tam tikrus platformos vartotojus-kūrėjus, matyti jų veiklą naujienų sraute.
* Registracija ir prisijungimas, naudojant OAUTH2 (Google).
* Muzikos skirstymas į tam tikrus žanrus, tų žanrų peržiūra.

# Sistemos architektūra

Sistema susideda iš šių pagrindinių dalių:

* Front-End – React + Redux + Babel + Webpack. Grafinei sąsajai kurti naudota Semantic UI React biblioteka.
* Back-End – ASP.NET Core 2.0 + Entity Framework.
* Failų saugojimo serveris – čia bus saugojami ir pasiekiami visi muzikos failai. Programos įgyvendinimo metu buvo nuspręsta naudoti Dropbox talpyklą.
* Duomenų bazė – saugojami platformos duomenys. Naudojamas Microsoft SQL Server.
* Auth0 autorizacijos serveris – OAUTH2 įgyvendinimui panaudotas Auth0 autentifikacijos bei autorizacijos serveris.

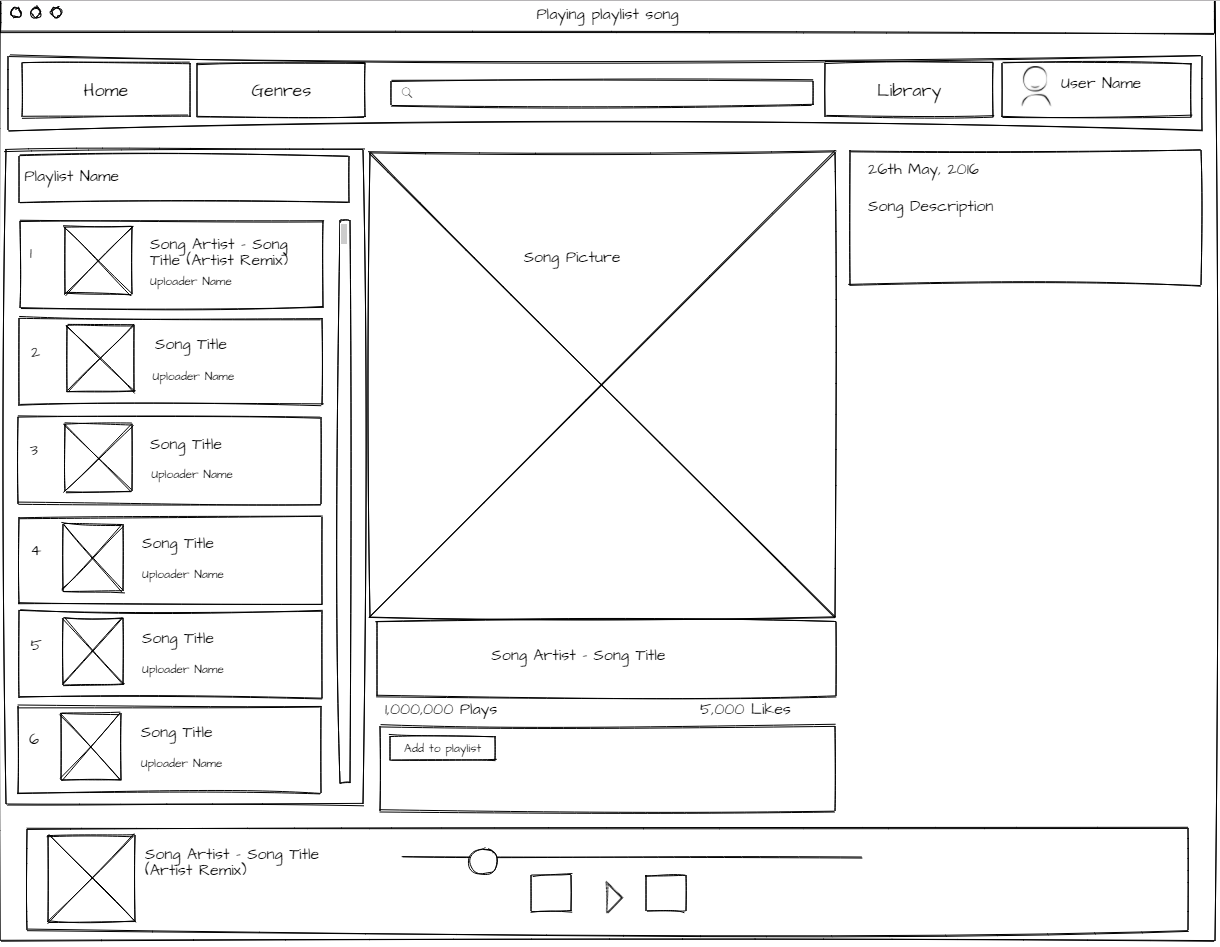


1 pav. Sistemos architektūra

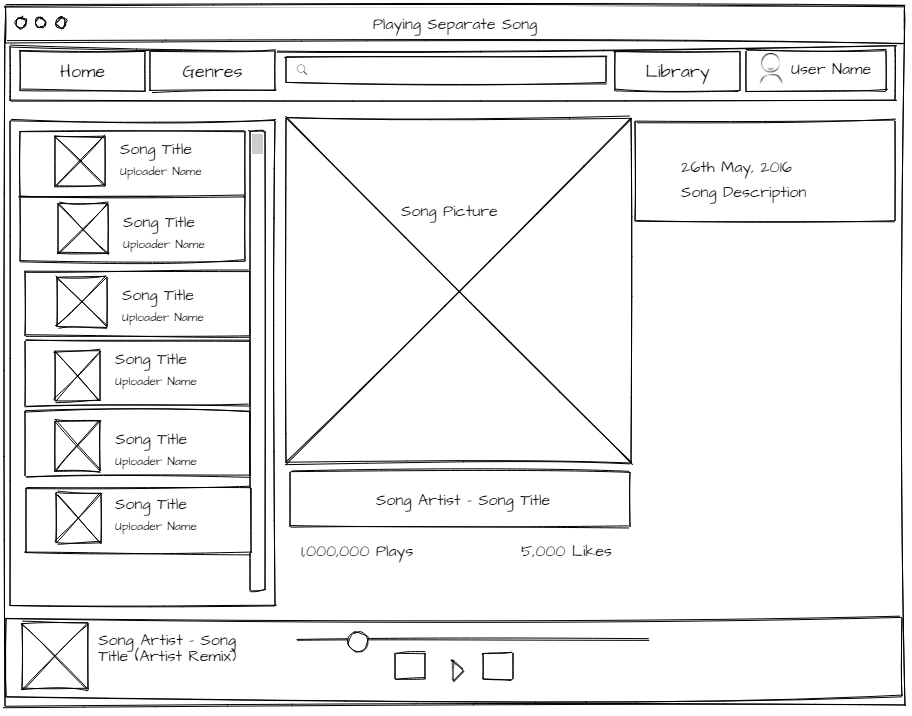
# Vartotojo sąsajos projektas

Vartotojo sąsajos projektavimui buvo naudojamas įrankis MockFlow. Buvo sukurti vartotojo sąsajos prototipai su šiuo įrankiu, vėliau pagal sukurtus prototipus buvo kuriama vartotojo sąsaja.

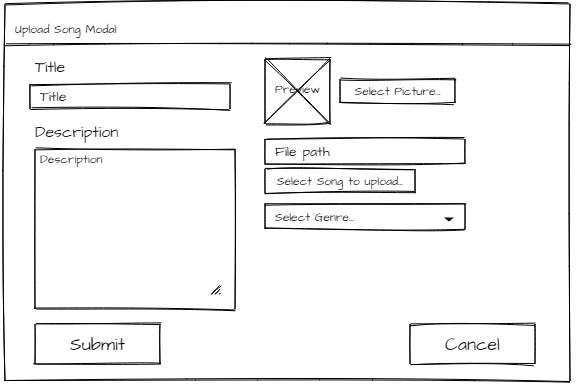
Žemiau pateikti sukurti vartotojo sąsajos prototipai:



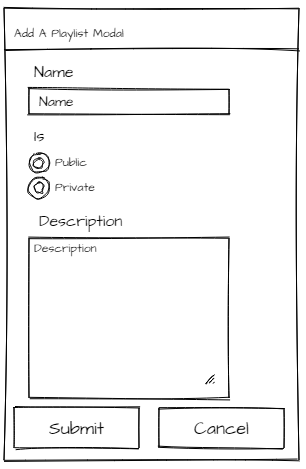
2 pav. Tam tikros grojaraščio dainos grojimo lango prototipas



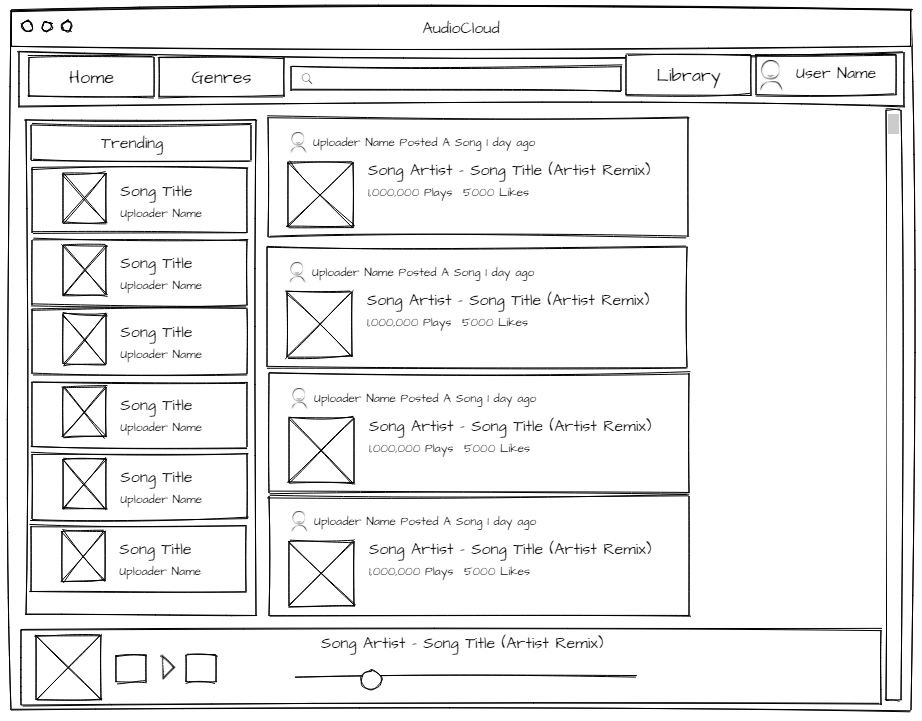
3 pav. Atskiros dainos grojimo lango prototipas



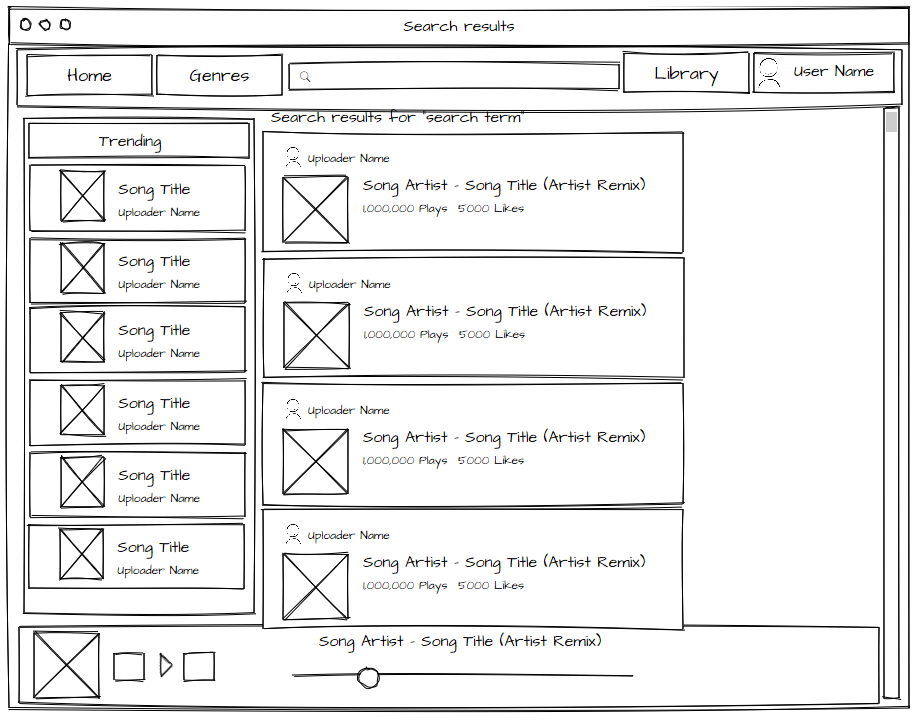
4 pav. Dainos sukūrimo modalinio lango prototipas



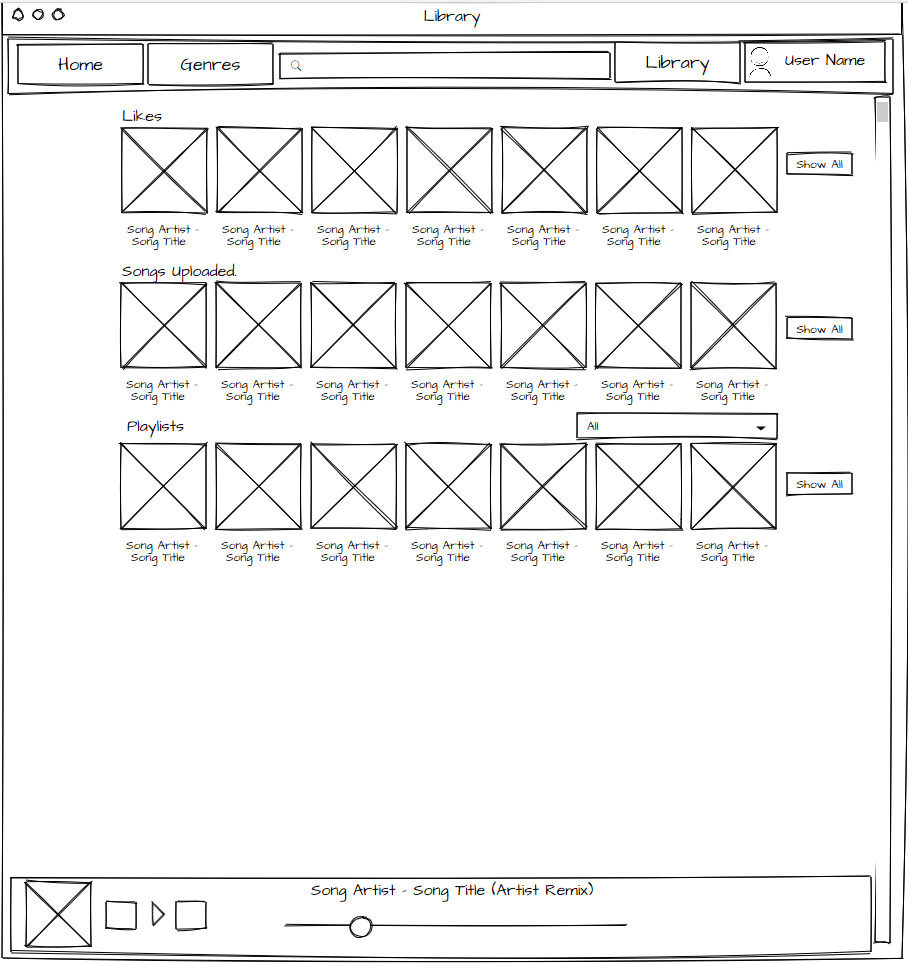
5 pav. Grojaraščio sukūrimo modalinio lango prototipas



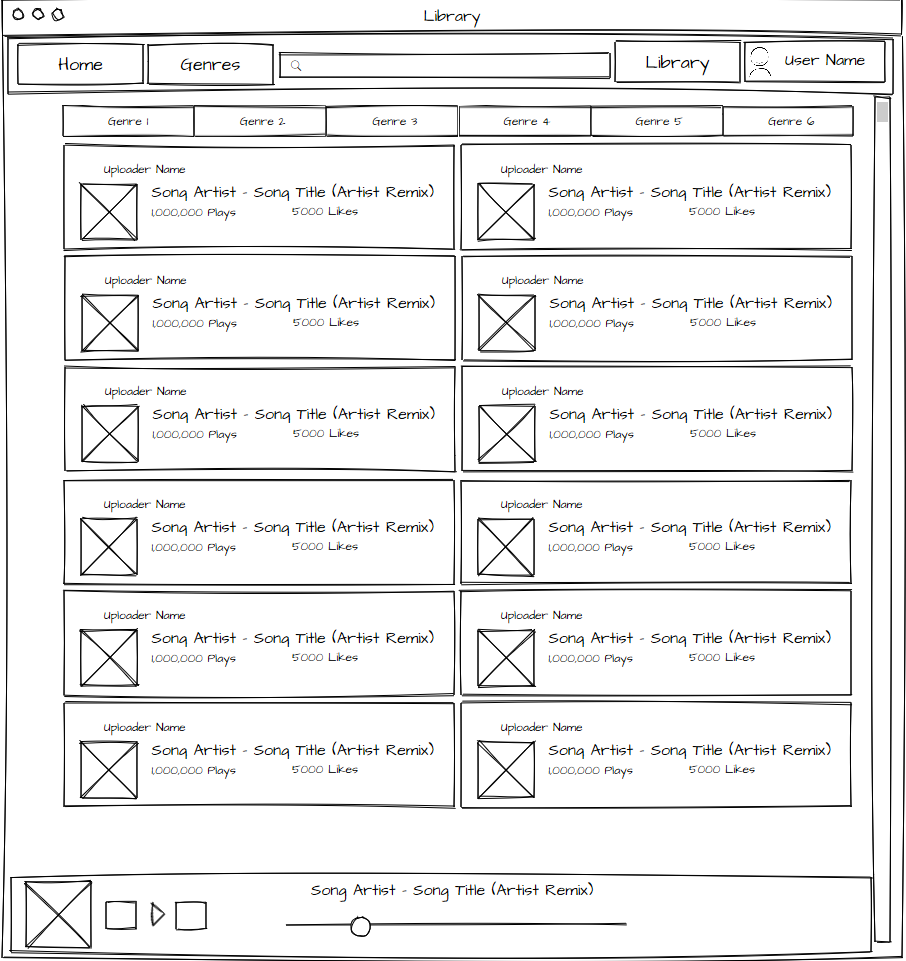
6 pav. Naujienų puslapio prototipas



7 pav. Paieškos rezultatų lango prototipas

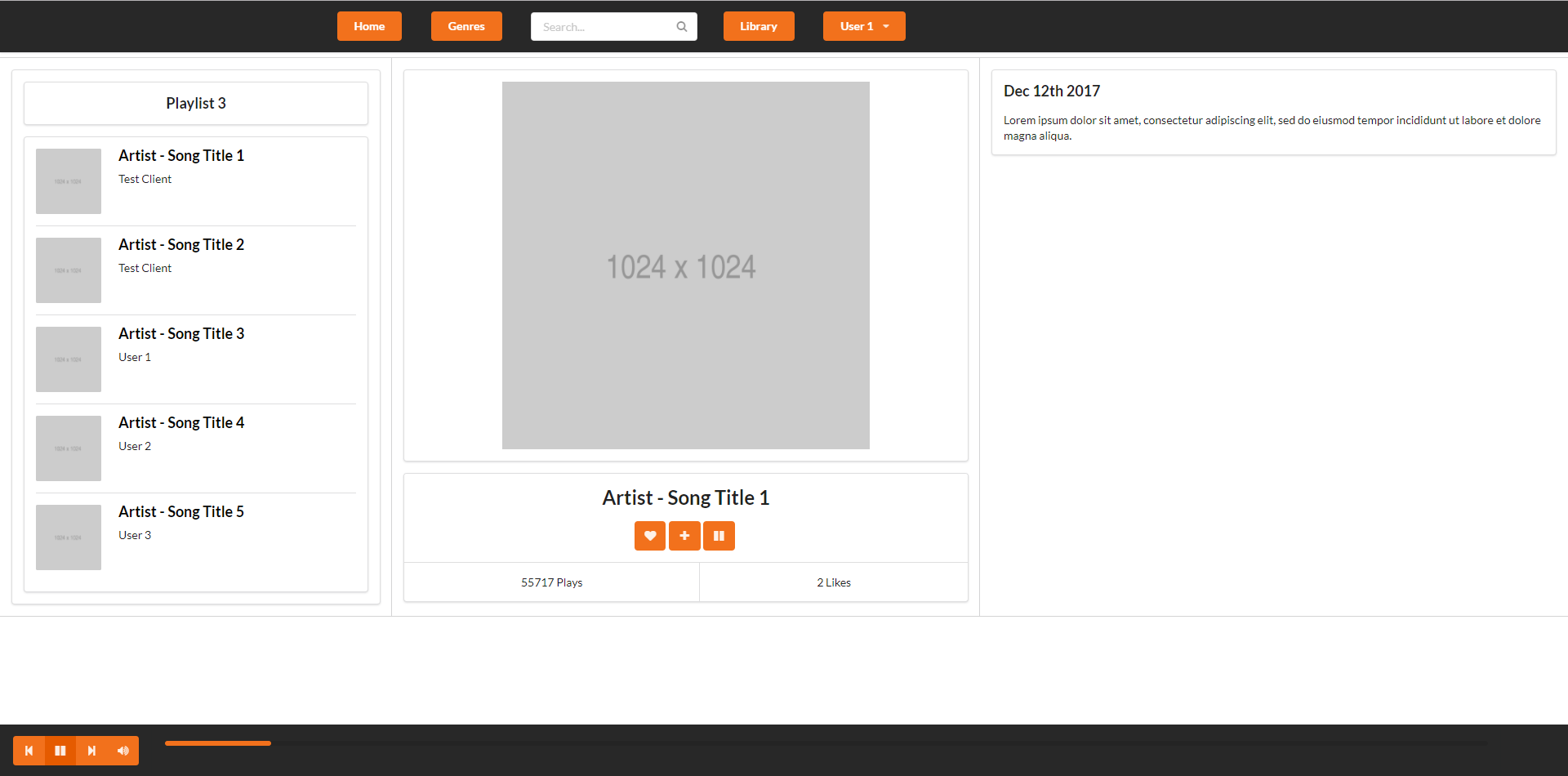


8 pav. Vartotojo bibliotekos puslapio prototipas

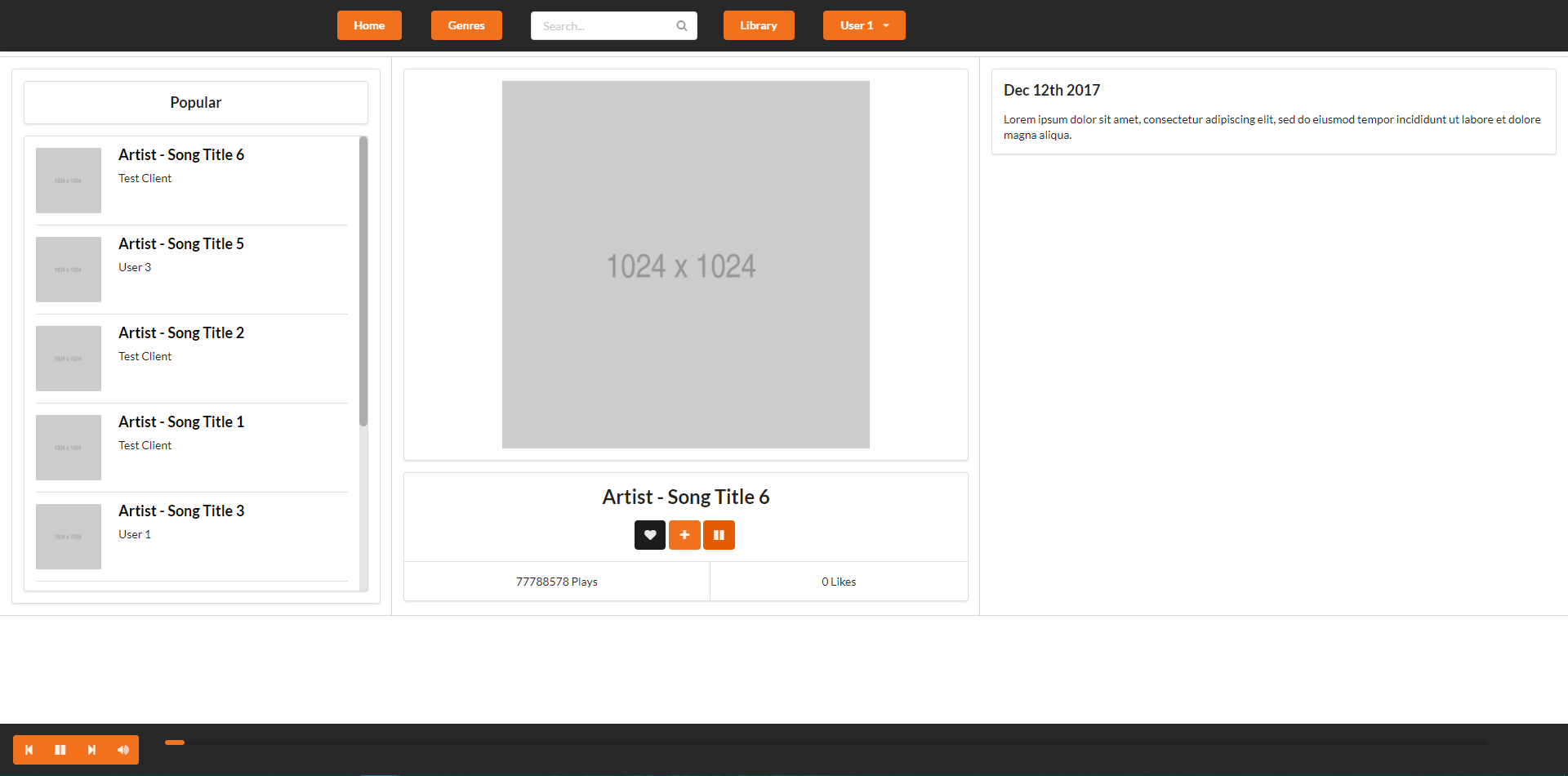


9 pav. Žanrų puslapio prototipas

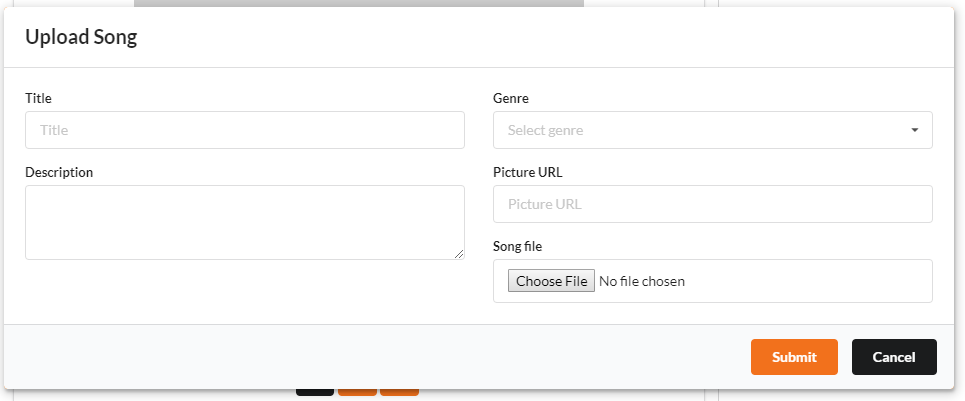
Pagal prototipus buvo įgyvendinama puslapio išvaizda. Grafinės sąsajos kūrimui buvo naudojamas Semantic UI React karkasas. Žemiau pateikti įgyvendintos grafinės sąsajos paveikslėliai.



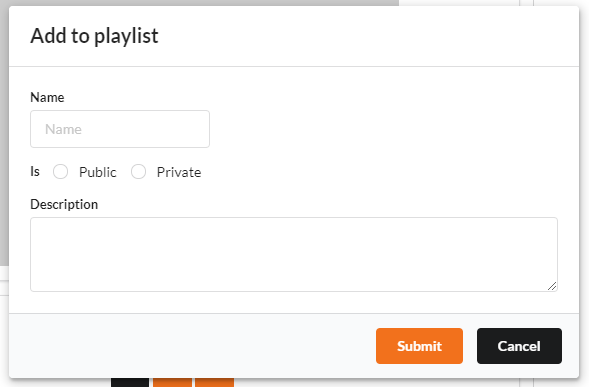
10 pav. Tam tikros grojaraščio dainos grojimo puslapis



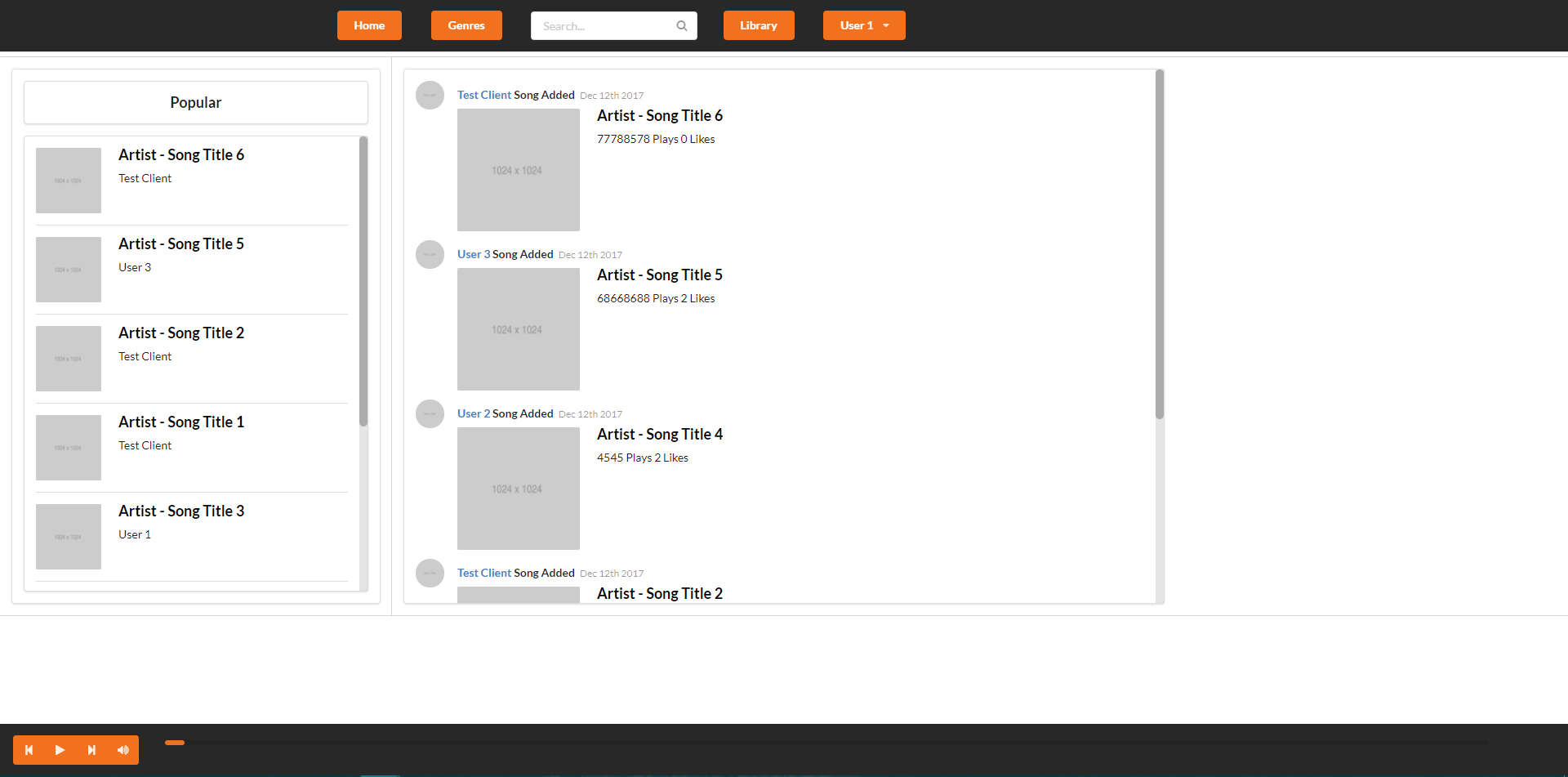
11 pav. Atskiros dainos grojimo puslapis



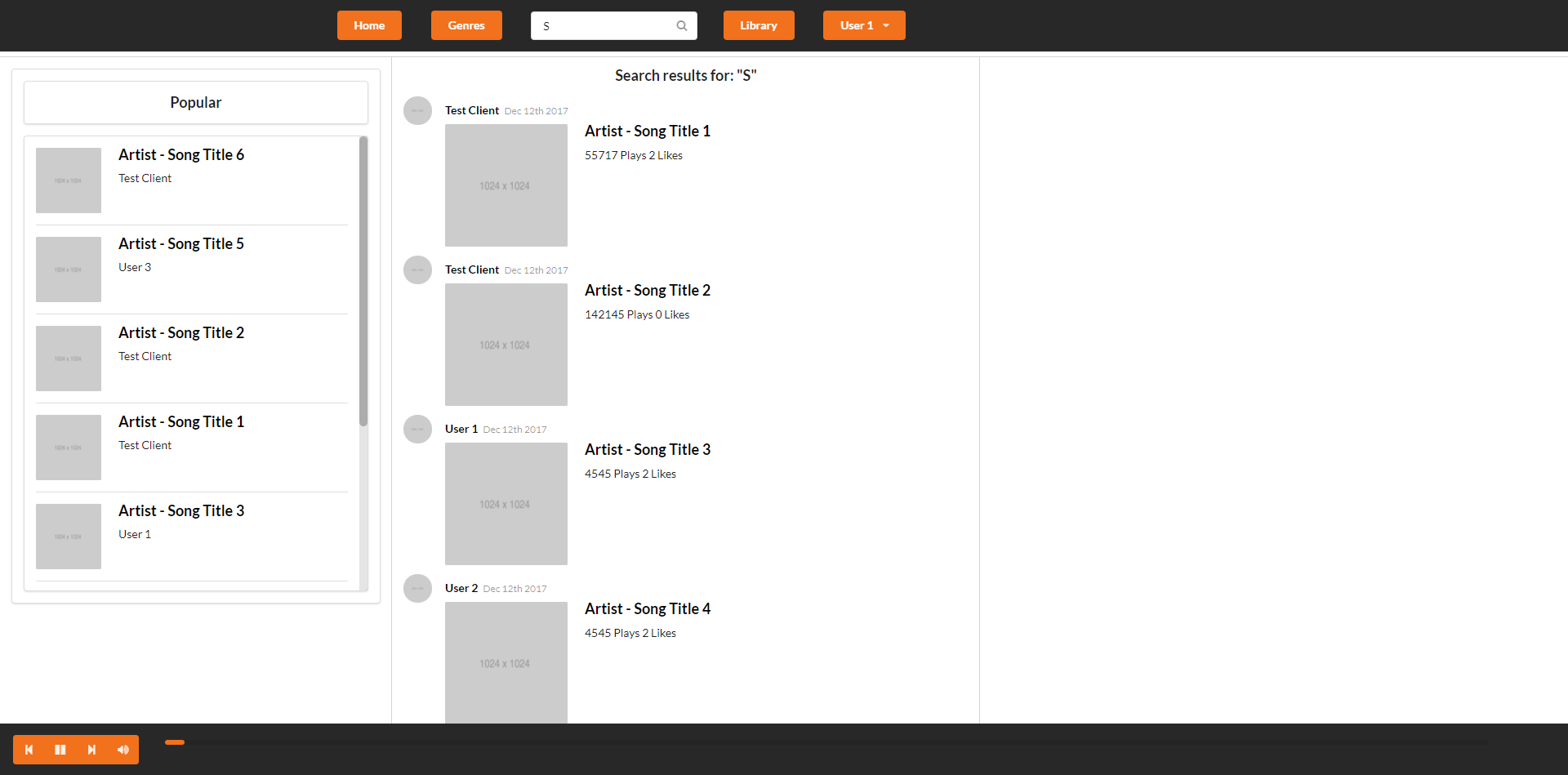
12 pav. Dainos sukūrimo modalinis langas



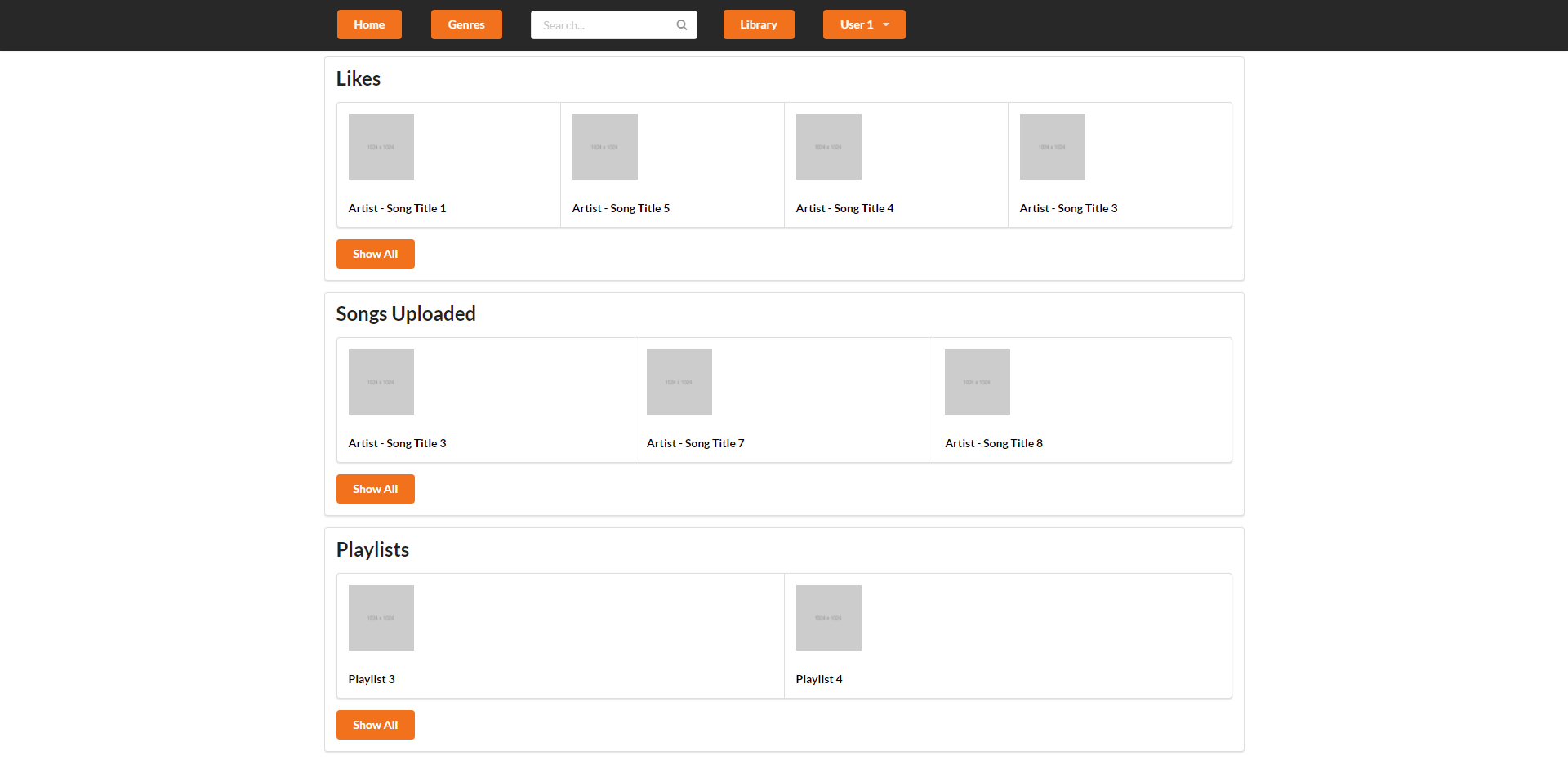
13 pav. Grojaraščio sukūrimo modalinis langas



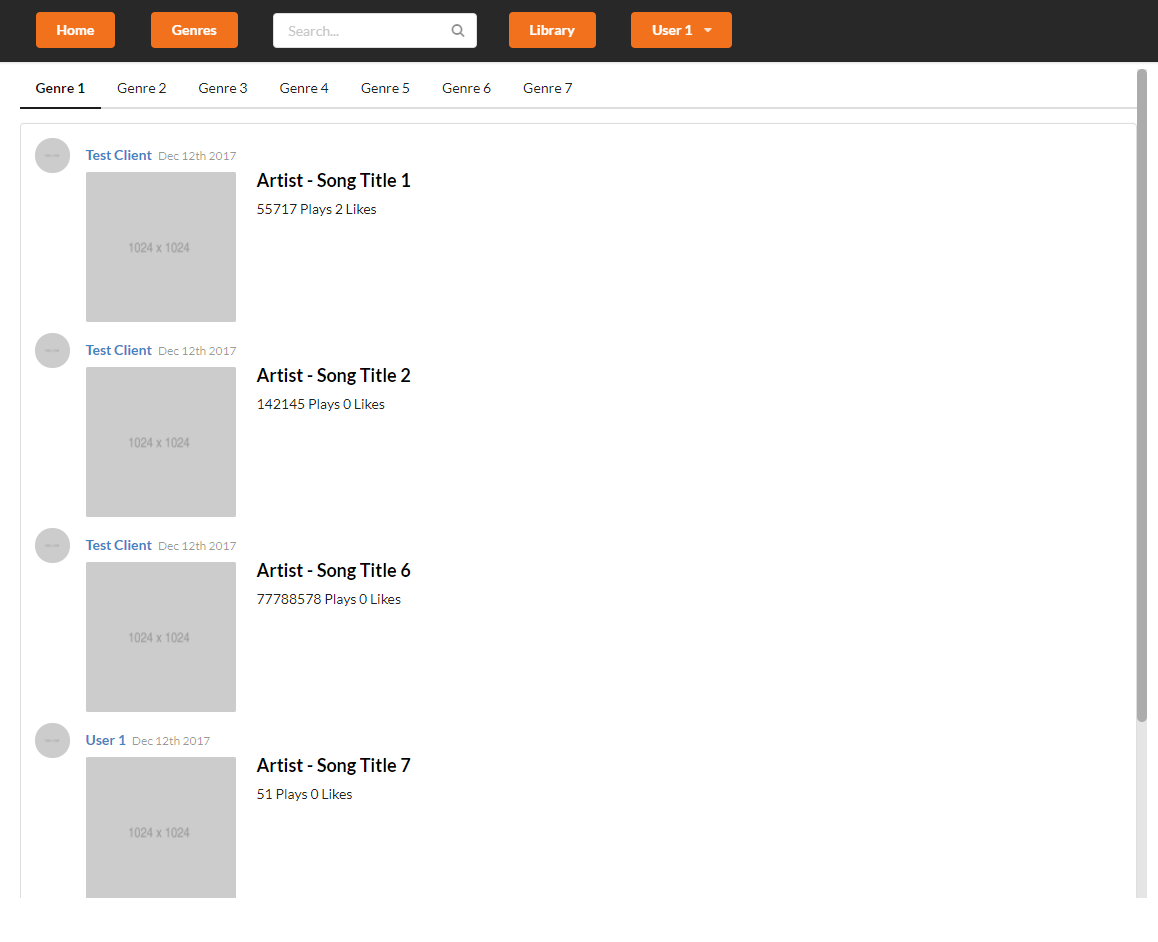
14 pav. Naujienų langas



15 pav. Paieškos rezultatų puslapis



16 pav. Vartotojo bibliotekos puslapis



17 pav. Žanrų puslapis

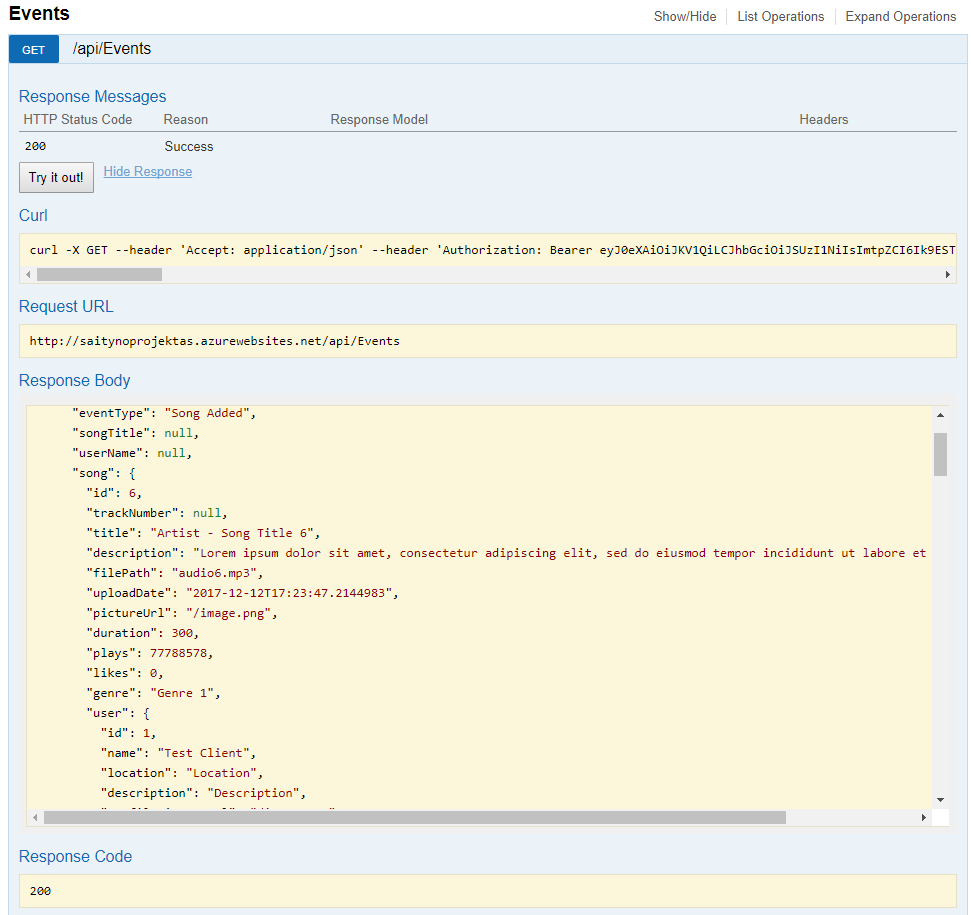
# API Specifikacija ir panaudojimo pavyzdžiai

Žemiau pateikta kiekvieno API metodo specifikacija ir panaudojimo pavyzdys.

# Events

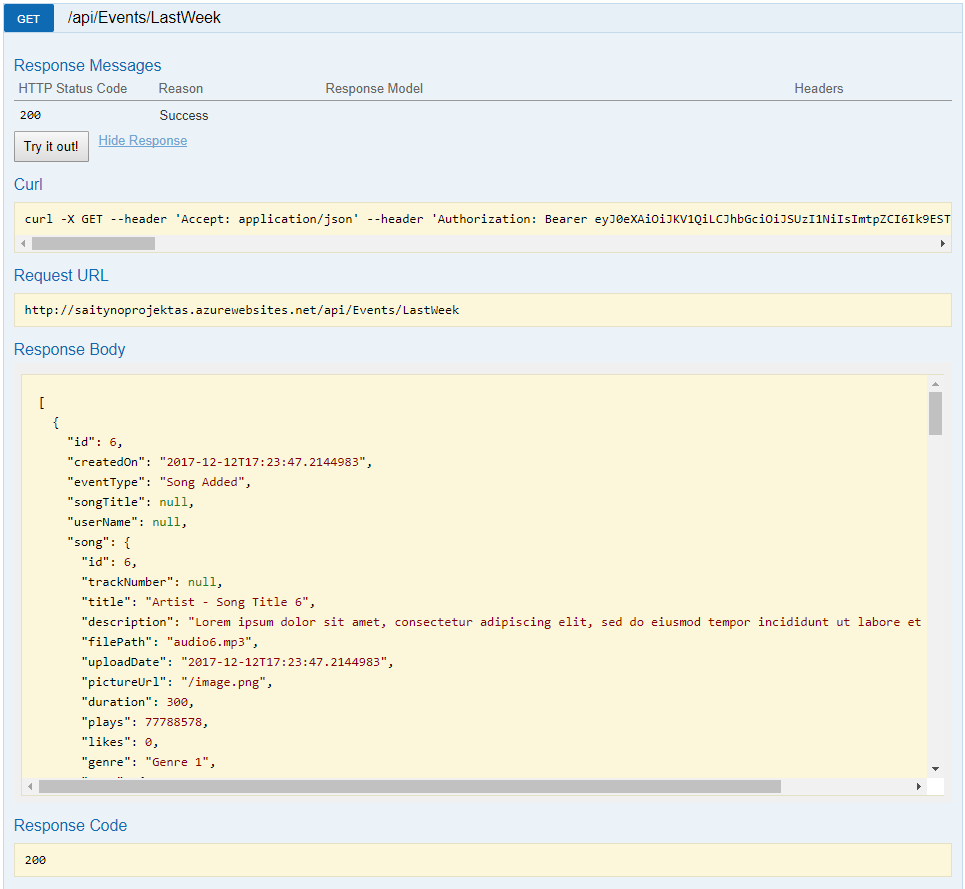
Sistemoje yra resursas „Įvykiai“. Šis resursas turi du panaudojimo metodus.

Pirmasis yra GET metodas, kuris gauna visų sistemoje įvykių sąrašą. Sistemoje kol kas yra tik vienas įvykio tipas: sukurta daina.



18 pav. Visų įvykių gavimo pavyzdys

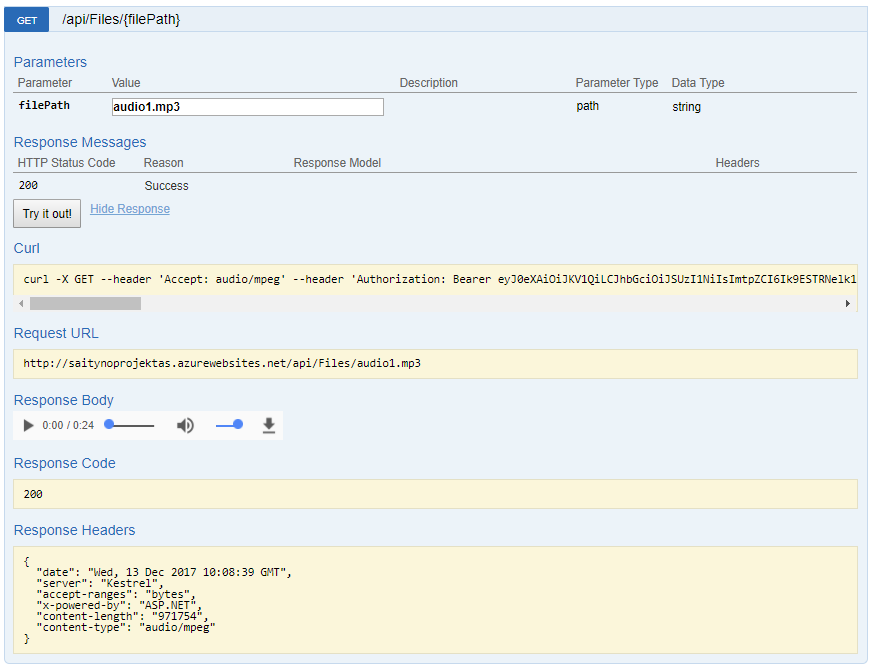
Kitas metodas taip pat GET ir grąžina paskutinės savaitės įvykius. Šis metodas naudojamas naujienų puslapyje. Abu metodai nereikalauja jokių parametrų, tereikia kad antraštėje būtų paduodamas priėjimo tokenas.



19 pav. Paskutinės savaitės įvykių gavimo pavyzdys

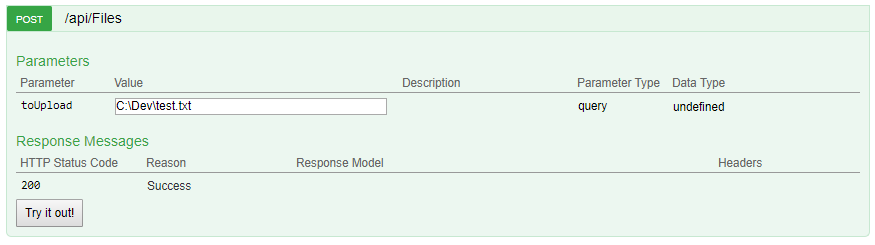
# Files

Failų resursas turi du prieigos metodus. Pirmasis skirtas dainos muzikos failų parsisiuntimui. Nurodomas vienas parametras – dainos sistemoje vardas. Muzikos failai saugomi Dropbox talpykloje, muzikos failo vardas sužinomas iš dainos informacijos.

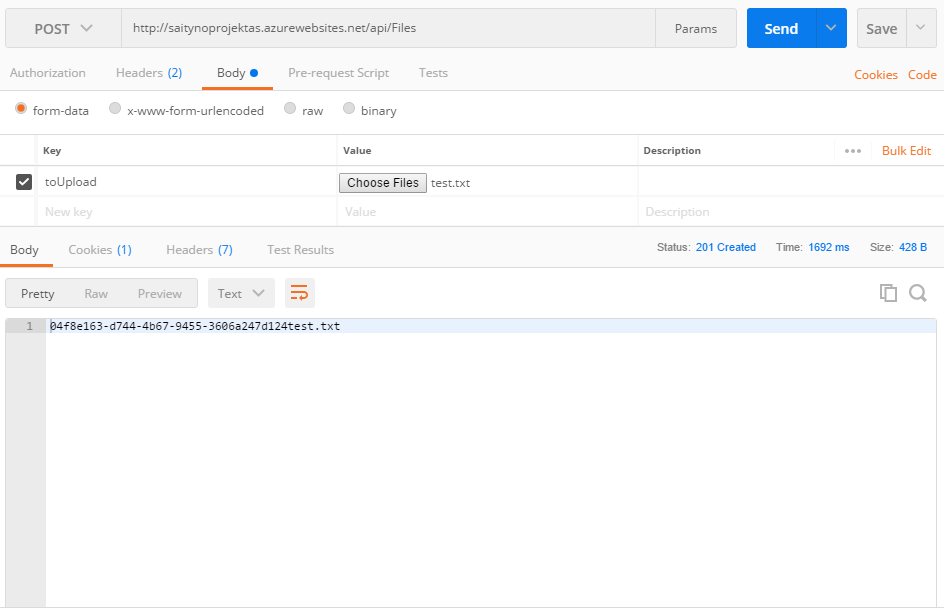


20 pav. Muzikos ar kito failo parsisiuntimo pavyzdys

Antrasis metodas yra POST ir skirtas muzikos ar kito failo įkėlimui. Įrankis Swagger UI nepalaiko failų įkėlimo testuojant metodus, todėl teko tam panaudoti Postman, žr. 22 pav. Metodas grąžina teksto pavidalu sistemoje sukurto failo vardą. Vartotojo sąsaja sukurta taip, kad pirma vykdomas šis metodas – sukuriamas failas talpykloje. O vėliau sukurto failo vardas ir kita dainos informacija siunčiama kitu API metodu. Dainos sukūrimo metode yra tikrinama, ar nurodytas failas egzistuoja talpykloje, ir ar jis nėra jau panaudotas kitos dainos.



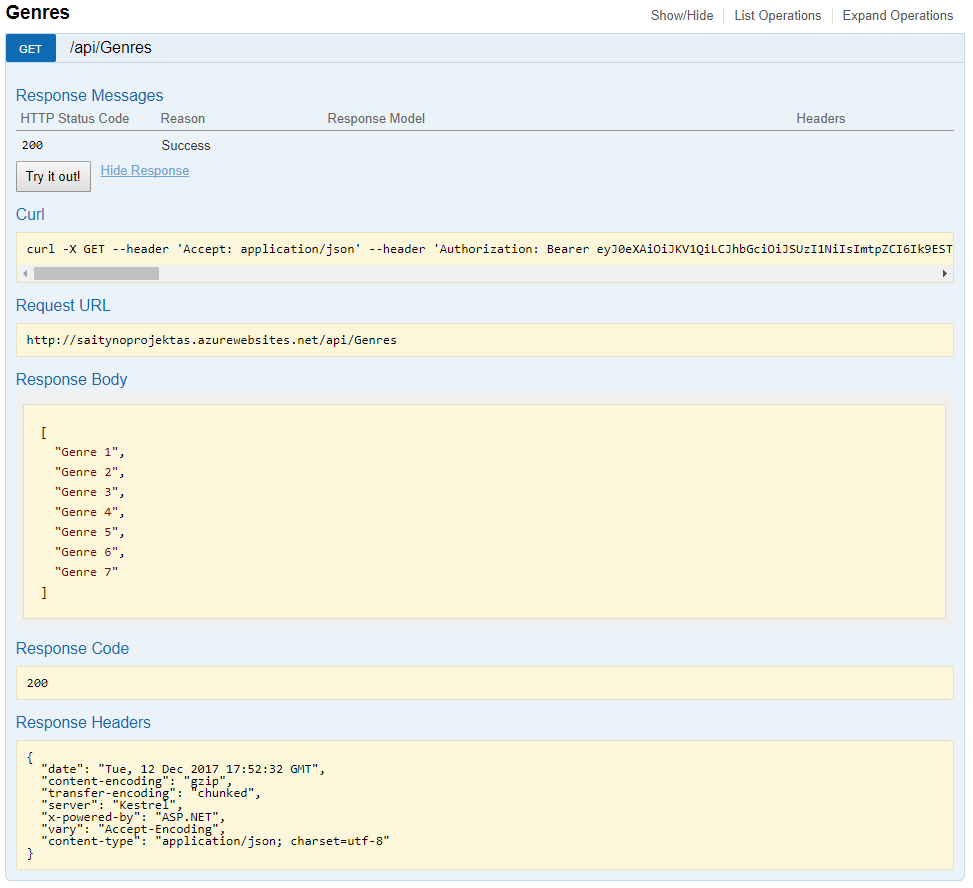
21 pav. Muzikos ar kito failo įkėlimo specifikacija



22 pav. Failo įkėlimo pavyzdys

# Genres

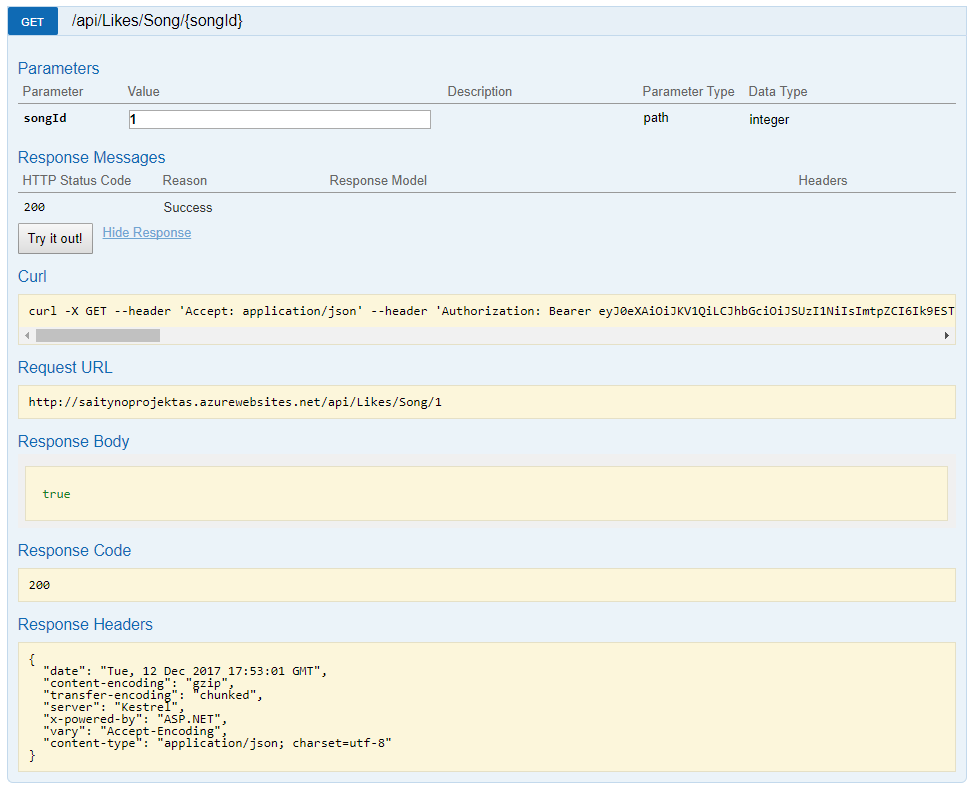
Žanrų resursas turi tik vieną paprastą GET metodą, kuris grąžina visų sistemoje esančių žanrų sąrašą. Šių žanrų sąrašas negali būti keičiamas vartotojų, jį įmanoma papildyti tik kreipiantis tiesiogiai į duomenų bazę.



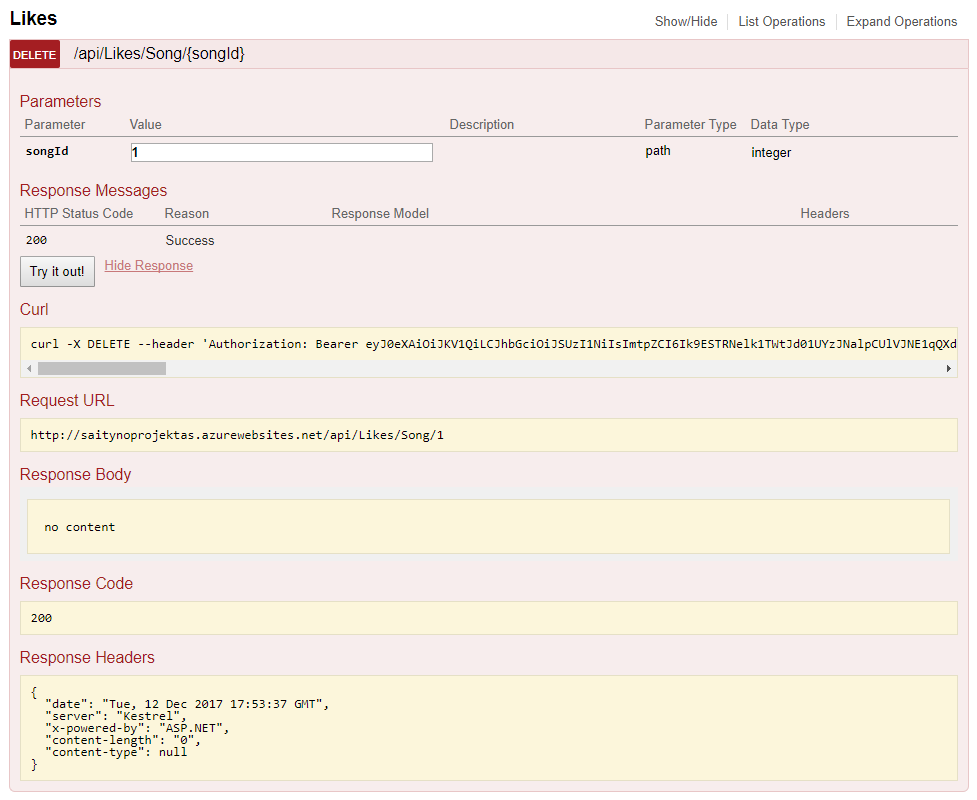
23 pav. Visų žanrų sistemoje gavimo pavyzdys

# Likes

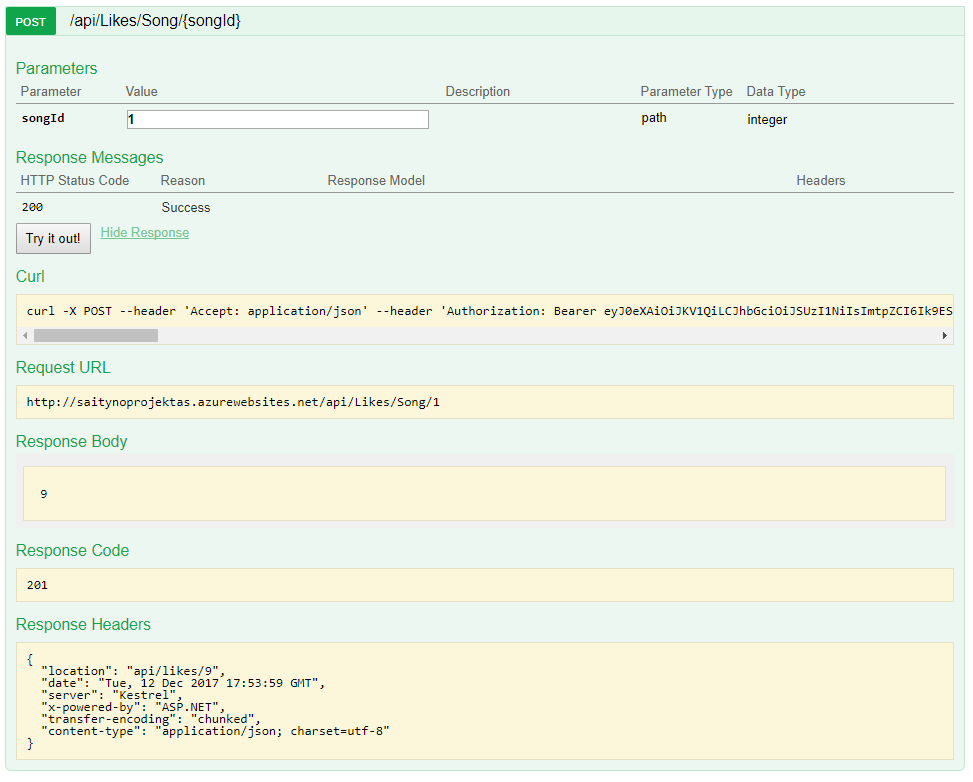
Pamėgimų resursas turi tris metodus: GET, POST ir DELETE. Šie metodai skirti, kad būtų gauta, ar prisijungęs vartotojas yra pamėgęs nurodytą dainą, kad prisijungęs vartotojas pamėgtų nurodytą dainą arba kad nurodytos dainos nebemėgtų. Parametruose paduodamas dainos id, antraštėje nurodomas prieigos tokenas.



24 pav. Gavimo, ar vartotojas mėgsta tam tikrą dainą, pavyzdys



25 pav. Vartotojo tam tikros dainos nepamėgimo pavyzdys

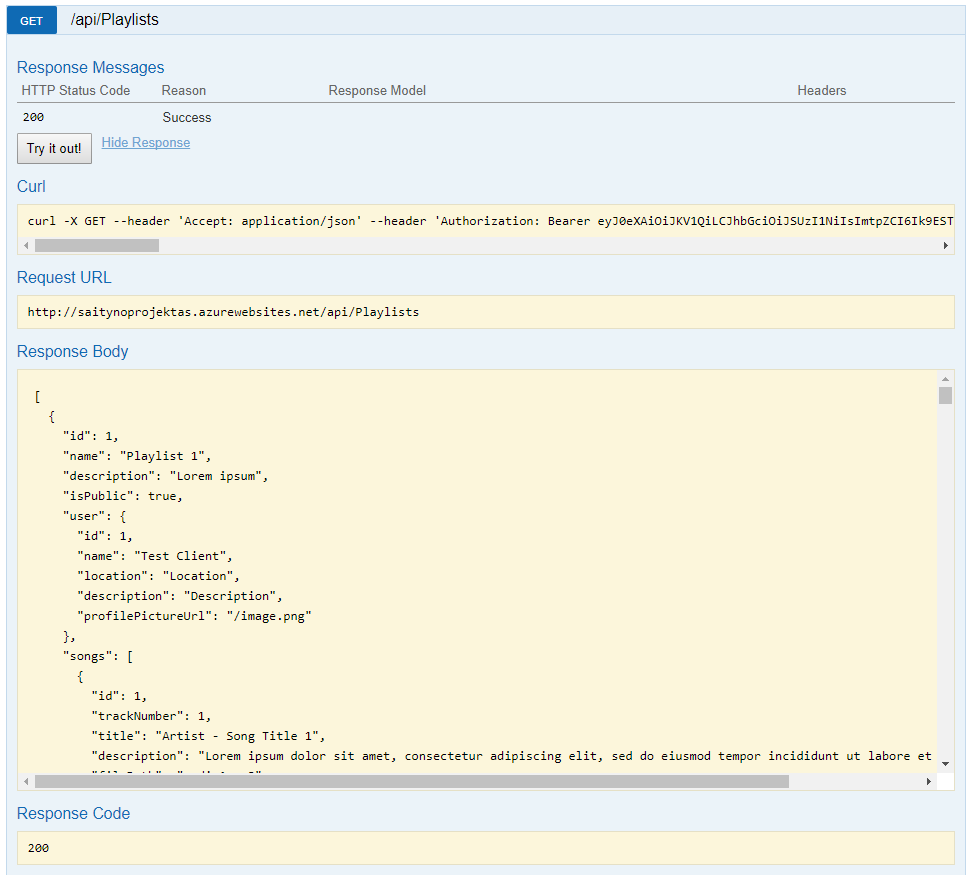


26 pav. Vartotojo tam tikros dainos pamėgimo pavyzdys

# Playlists

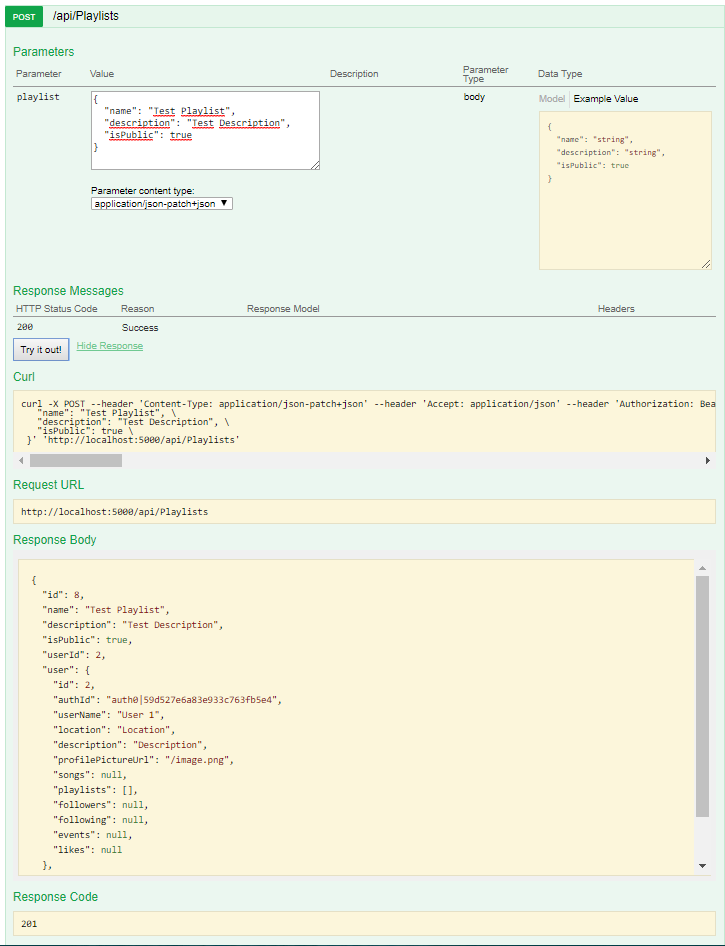
Grojaraščio resursas turi 8 metodus. Žemiau prie kiekvieno paveikslėlio parašyta apie kiekvieną atskirai.

27 pav. pavaizduotas metodas skirtas visų sistemoje esančių grojaraščių sąrašui gauti. Paduodamas tik antraštėje vartotojo prieigos tokenas. Svarbu tai, kad šiuo metodu nepavyks gauti privačiai sukurtų grojaraščių, nebent tuos privačius grojaraščius, kuriuos sukūrė metodą kviečiantis vartotojas.



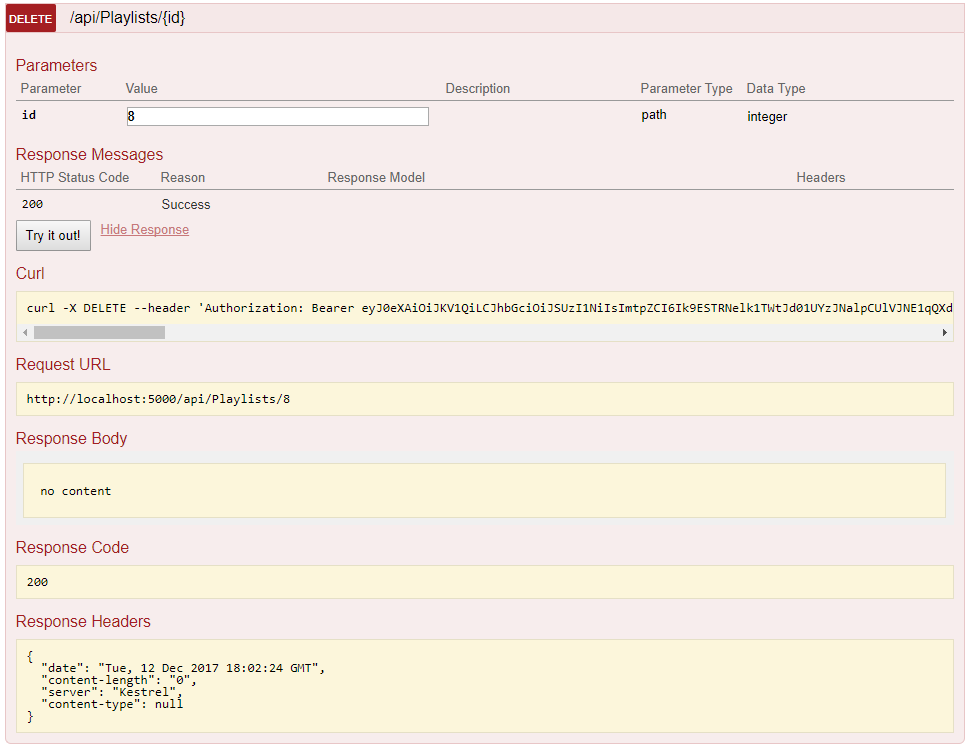
27 pav. Visų sistemoje grojaraščių gavimo pavyzdys

28 pav. pavaizduotas POST metodas, kuris sukuria grojaraštį sistemoje. Nurodomi trys parametrai: grojaraščio pavadinimas, aprašymas bei pasiekiamumo lygis – privatus arba viešas.



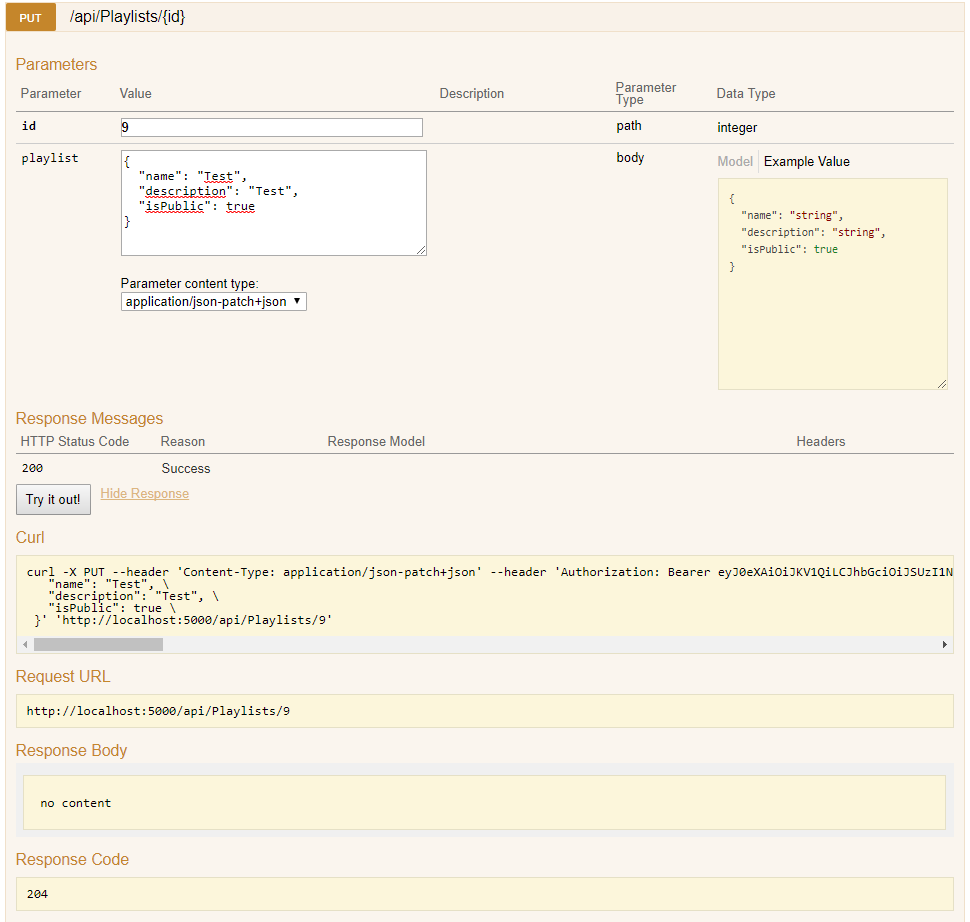
28 pav. Grojaraščio sukūrimo pavyzdys

29 pav. Pavaizduotas metodas, skirtas grojaraščio pašalinimui. Paduodamas tik grojaraščio id.



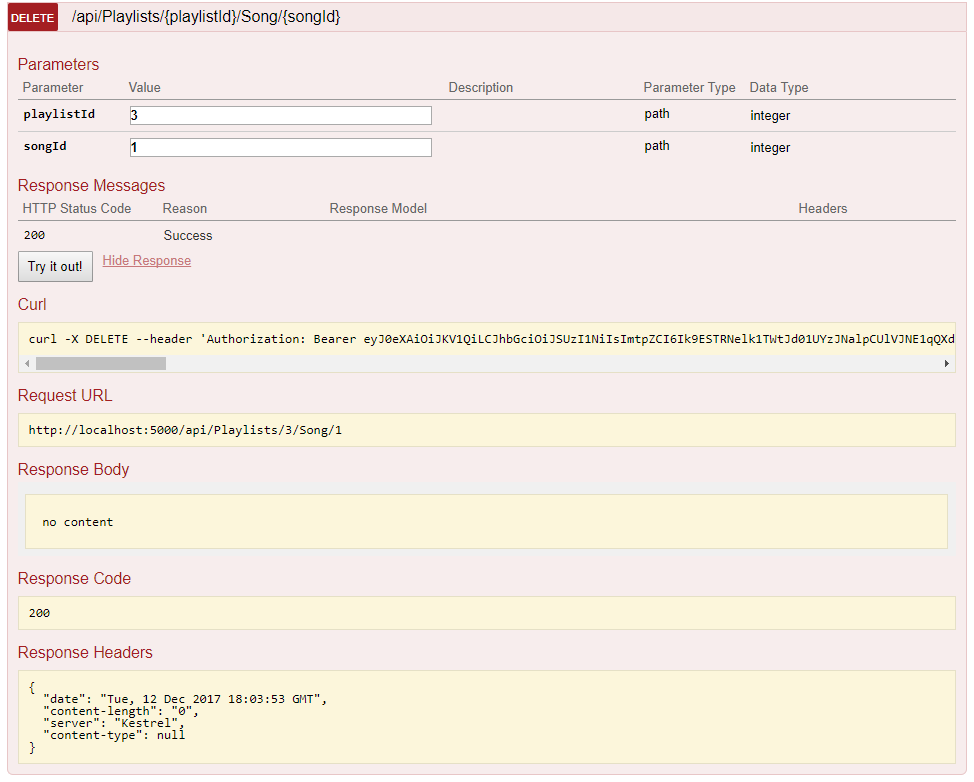
29 pav. Grojaraščio pašalinimo pavyzdys

30 pav. pavaizduotas PUT metodas, skirtas paredaguoti grojaraščio informacijai. Paduodamas grojaraščio id bei norimi parametrai, kurie bus keičiami. Neprivaloma paduoti visų parametrų, tik tuos, kuriuos norima pakeisti.

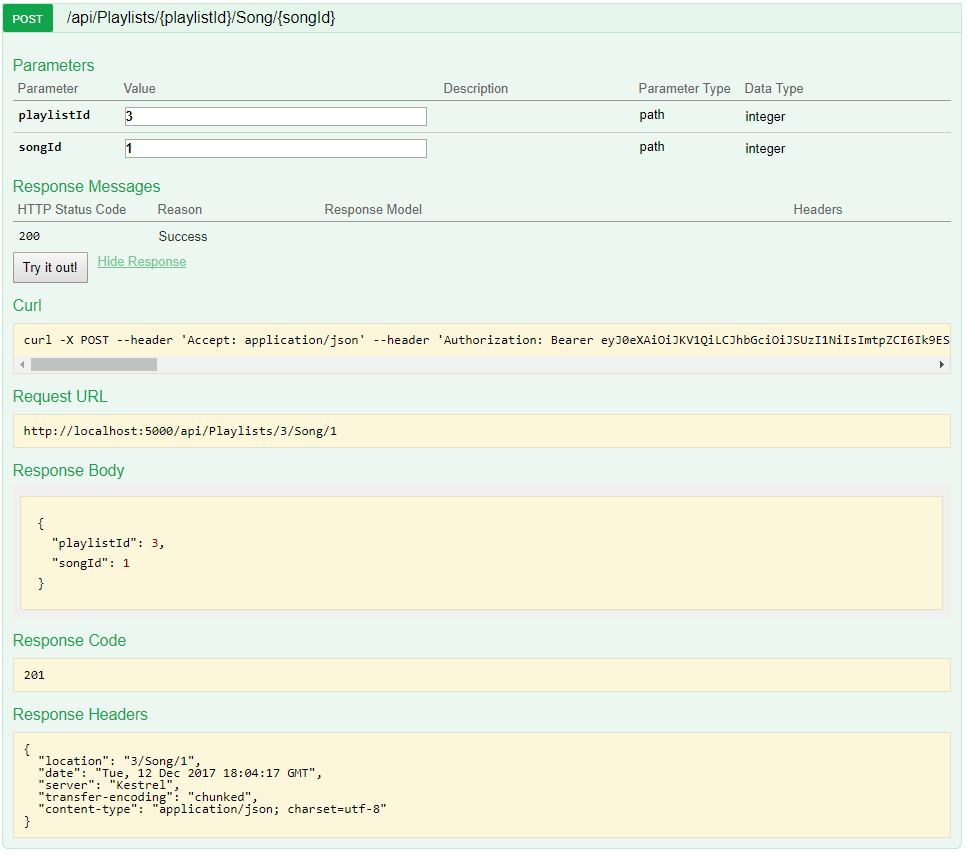


30 pav. Grojaraščio redagavimo pavyzdys

31 pav. ir 32 pav. pavaizduoti dainos iš grojaraščio pašalinimo ir įdėjimo į grojaraštį metodai. Abu priima du parametrus: dainos id ir grojaraščio id.

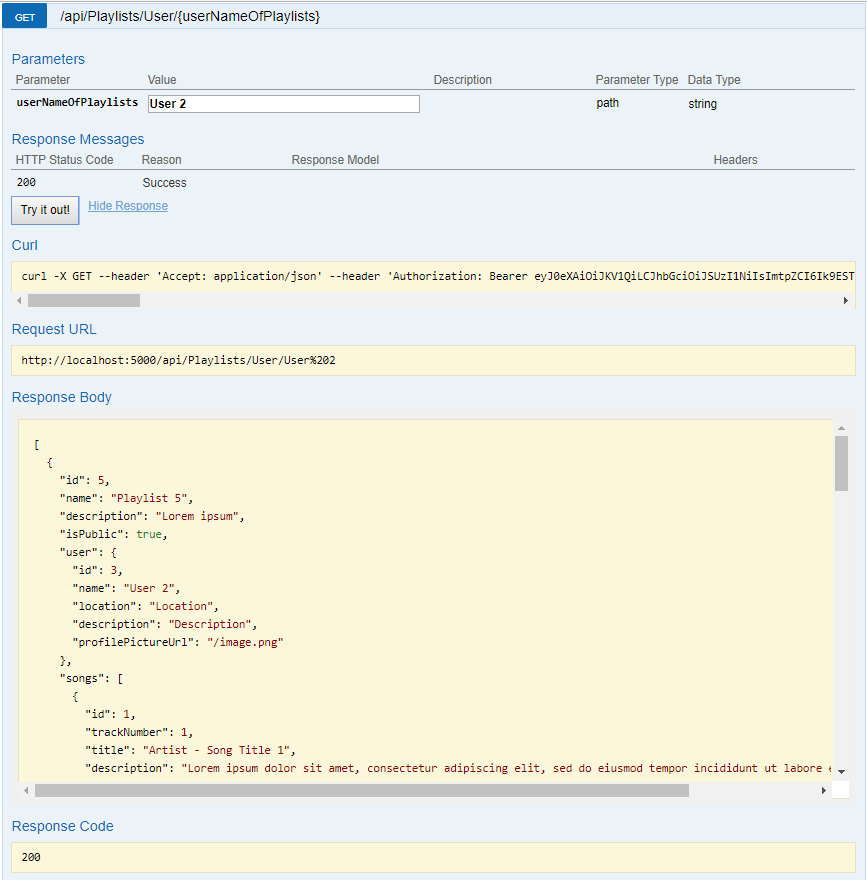


31 pav. Dainos iš grojaraščio pašalinimo pavyzdys



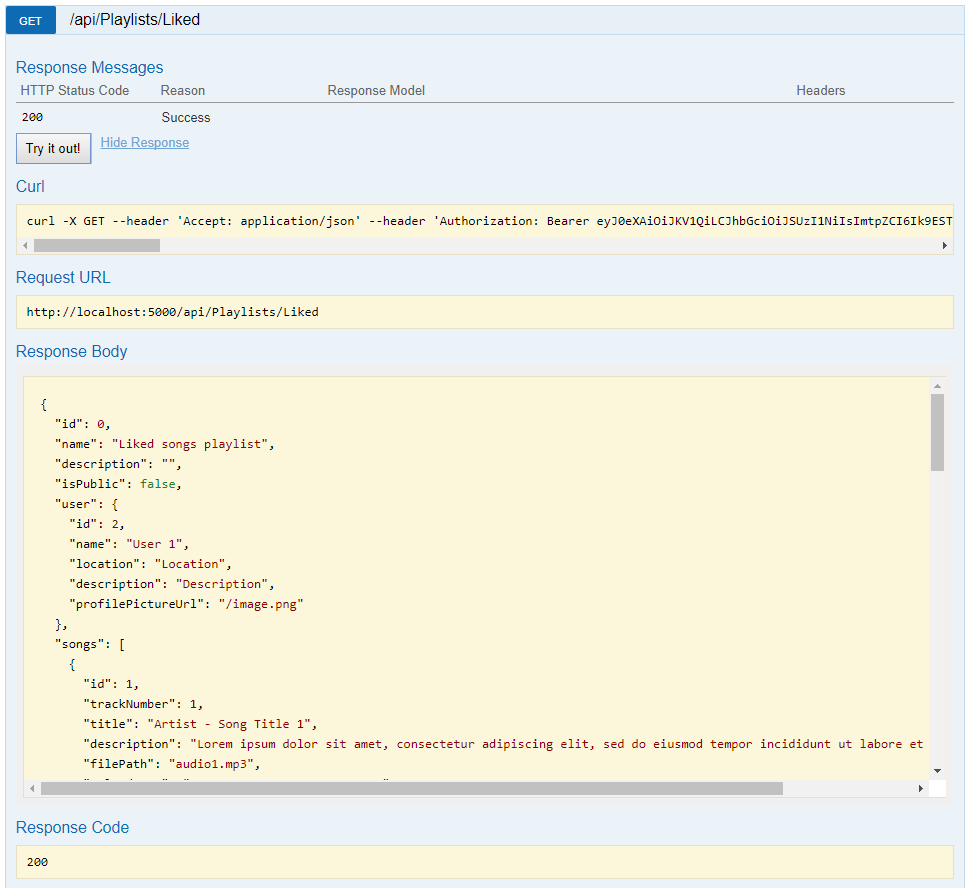
32 pav. Dainos įdėjimo į grojaraštį pavyzdys

33 pav. pavaizduotas GET metodas, kuris gauna nurodyto vartotojo grojaraščius. Jeigu metodą kviečiančio vartotojo vardas sutampa su nurodyto vartotojo vardu, tai grąžinami ir šio vartotojo privatūs grojaraščiai. Metodas priima vieną parametrą – vartotojo vardą.



33 pav. Tam tikro vartotojo grojaraščių gavimo pavyzdys

34 pav. pavaizduotas GET metodas, kuris grąžina prisijungusio vartotojo visų pamėgtų dainų grojaraštį. Metodas nepriima jokių parametrų, vartotoją identifikuoja su prieigos tokenu.

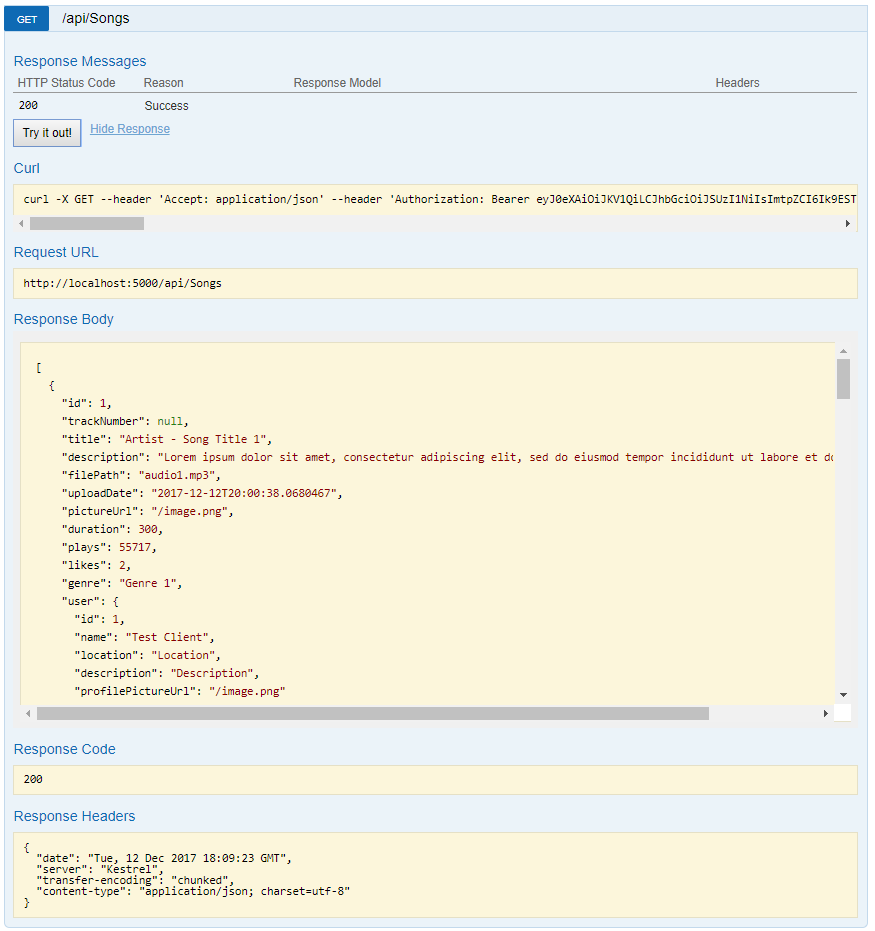


34 pav. Prisijungusio vartotojo pamėgtų dainų grojaraščio gavimo pavyzdys

# Songs

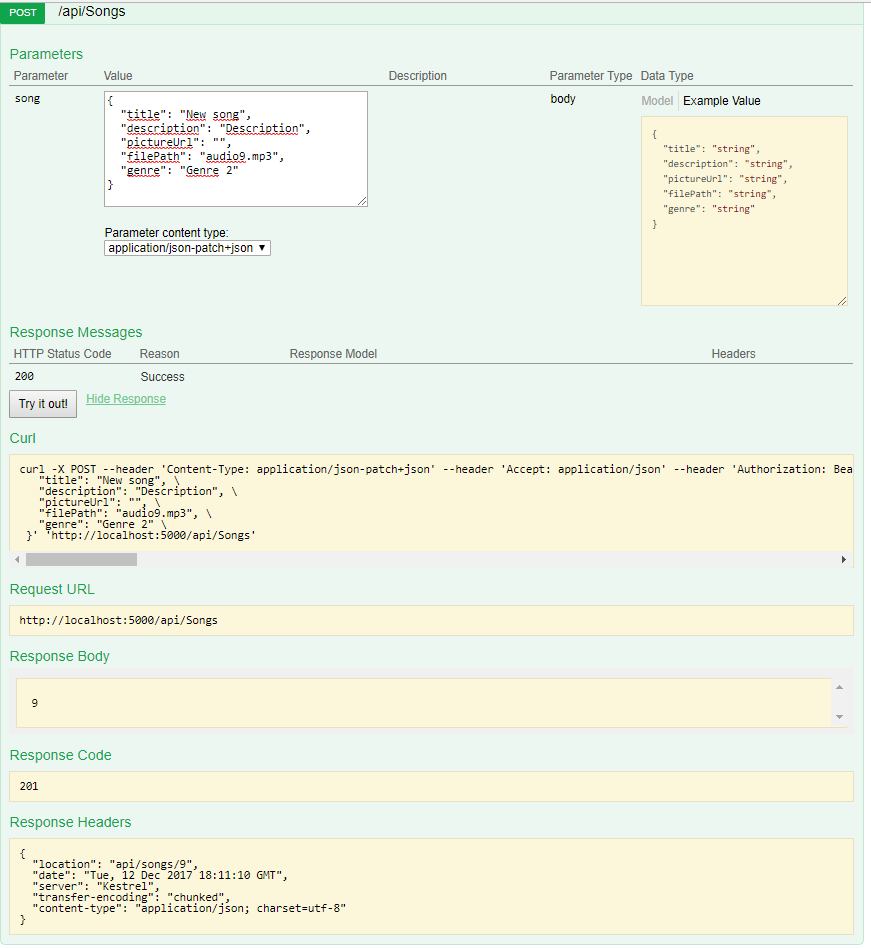
Dainų resursas turi 4 metodus. Žemiau prie kiekvieno paveikslėlio apie kiekvieną parašyta išsamiau.

35 pav. pavaizduotas GET metodas, kuris grąžina visų sistemoje esančių dainų sąrašą. Visos dainos sistemoje yra viešai pasiekiamos.



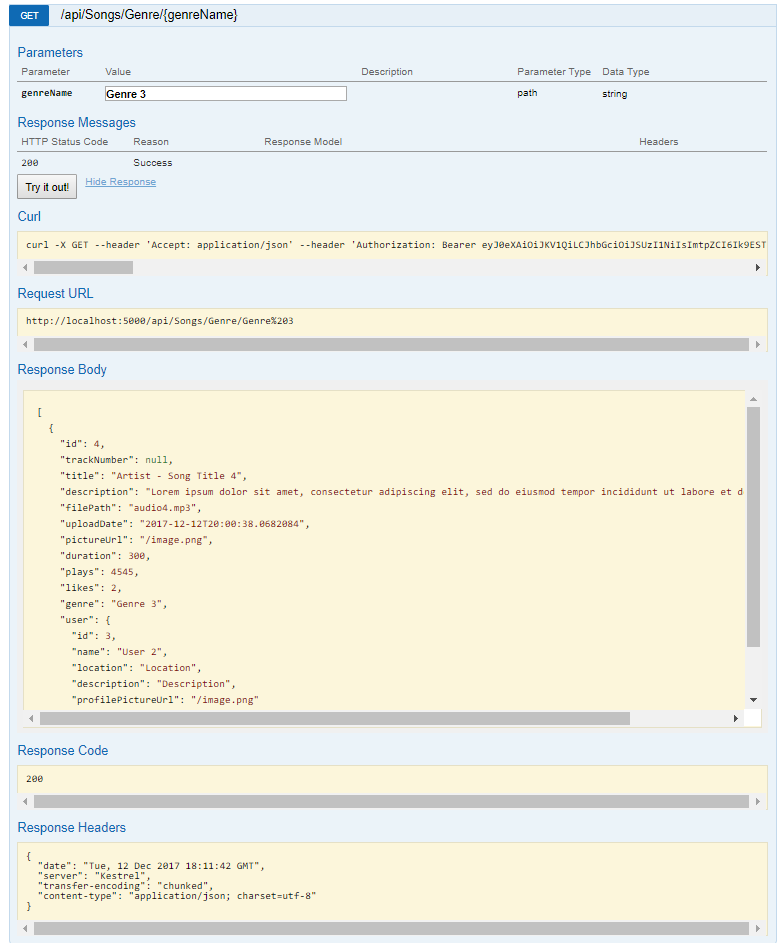
35 pav. Visų sistemoje dainų gavimo pavyzdys

36 pav. pavaizduotas dainos sukūrimo POST metodas. Paduodami šie parametrai: dainos antraštė, aprašymas, paveikslėlio nuoroda, muzikos failo vardas ir žanras. Žanrą būtina nurodyti vieną iš egzistuojančių sistemoje, o muzikos failo vardas turi egzistuoti talpykloje bei būti nepanaudotas.

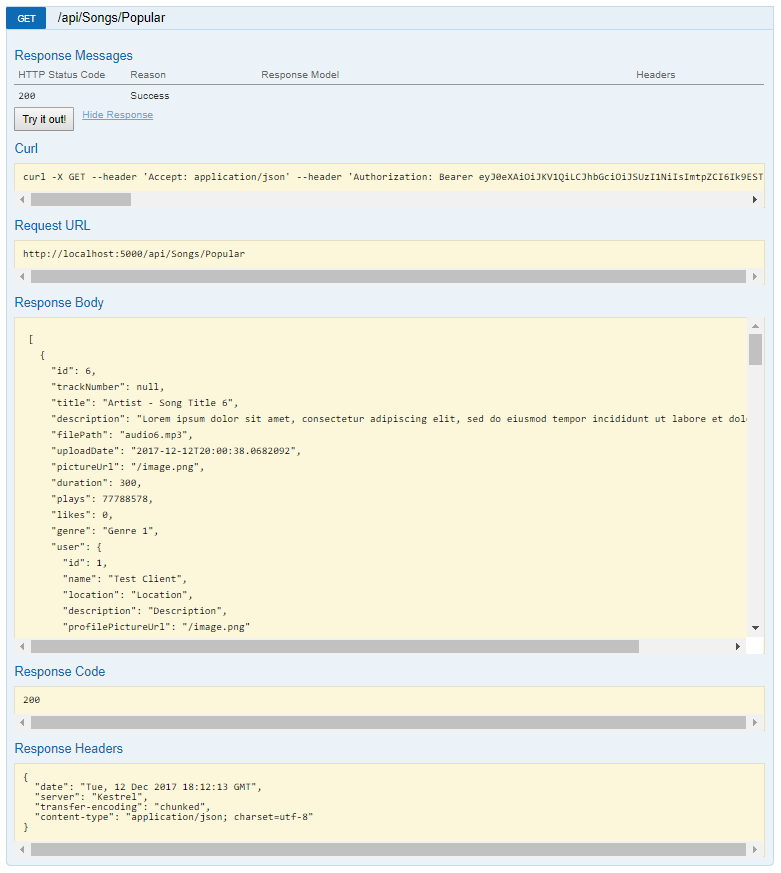


36 pav. Dainos sukūrimo pavyzdys

37 pav. ir 38 pav. pavaizduoti dar du dainų sąrašo gavimo GET metodai. Pirmasis grąžina tam tikro žanro dainų sąrašą, o antrasis grąžina sistemoje populiarių dainų sąrašą. Populiarios dainos yra tos, kurios turi daugiausiai peržiūrų. Jų yra grąžinama 10.



37 pav. Tam tikro žanro visų dainų gavimo pavyzdys

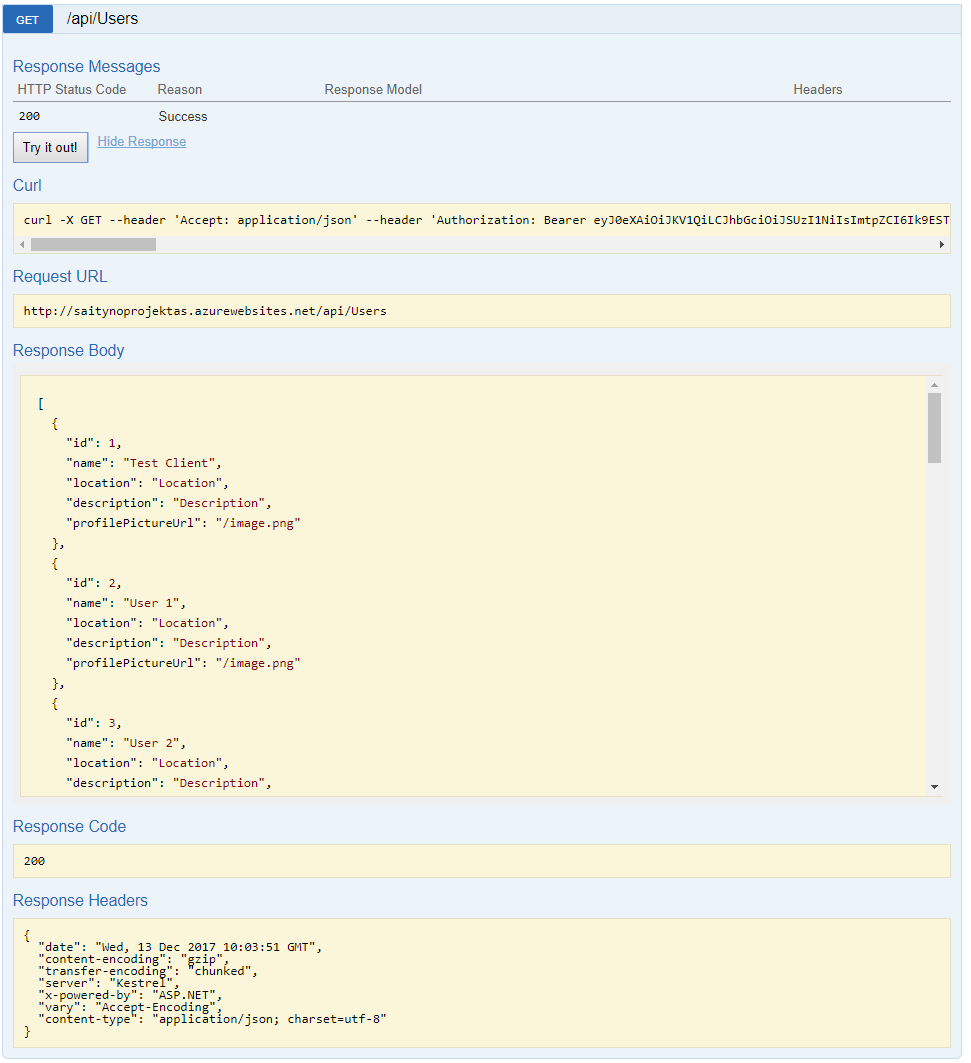


38 pav. Populiarių dainų gavimo pavyzdys

# Users

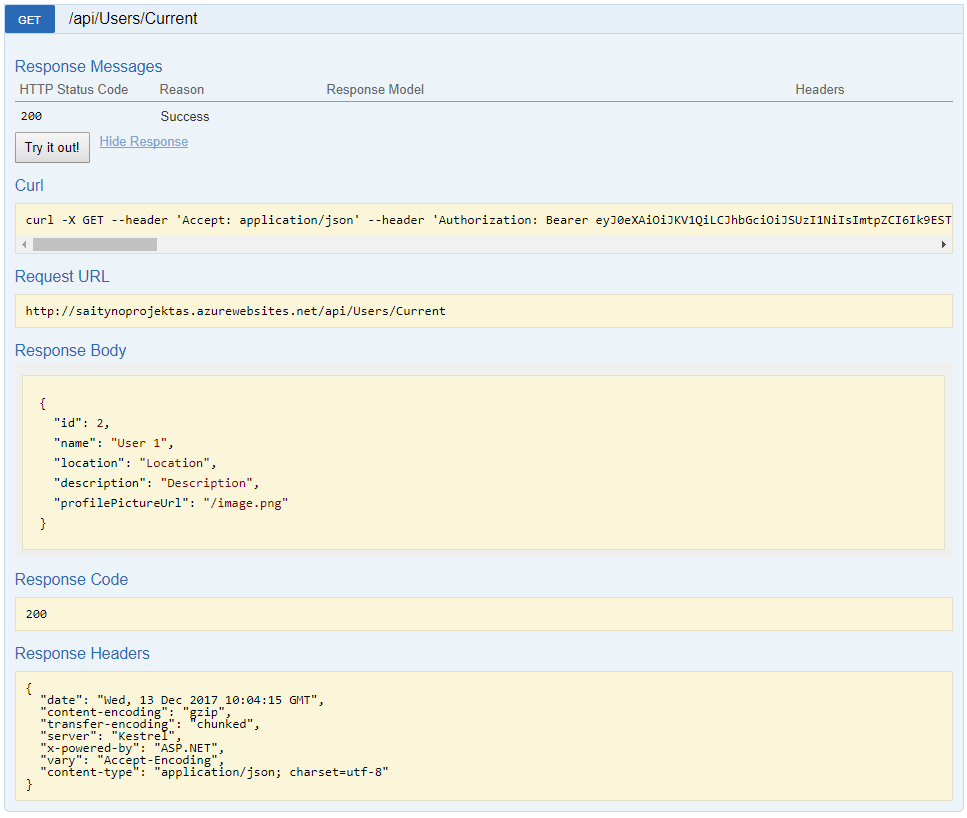
Vartotojo resursas turi 6 metodus. Žemiau prie kiekvieno paveikslėlio apie kiekvieną aprašyta išsamiau.

39 pav. pavaizduotas GET metodas, kuris grąžina visų vartotojų sistemoje sąrašą. Metodas priima tik antraštėje nurodytą prieigos tokeną.



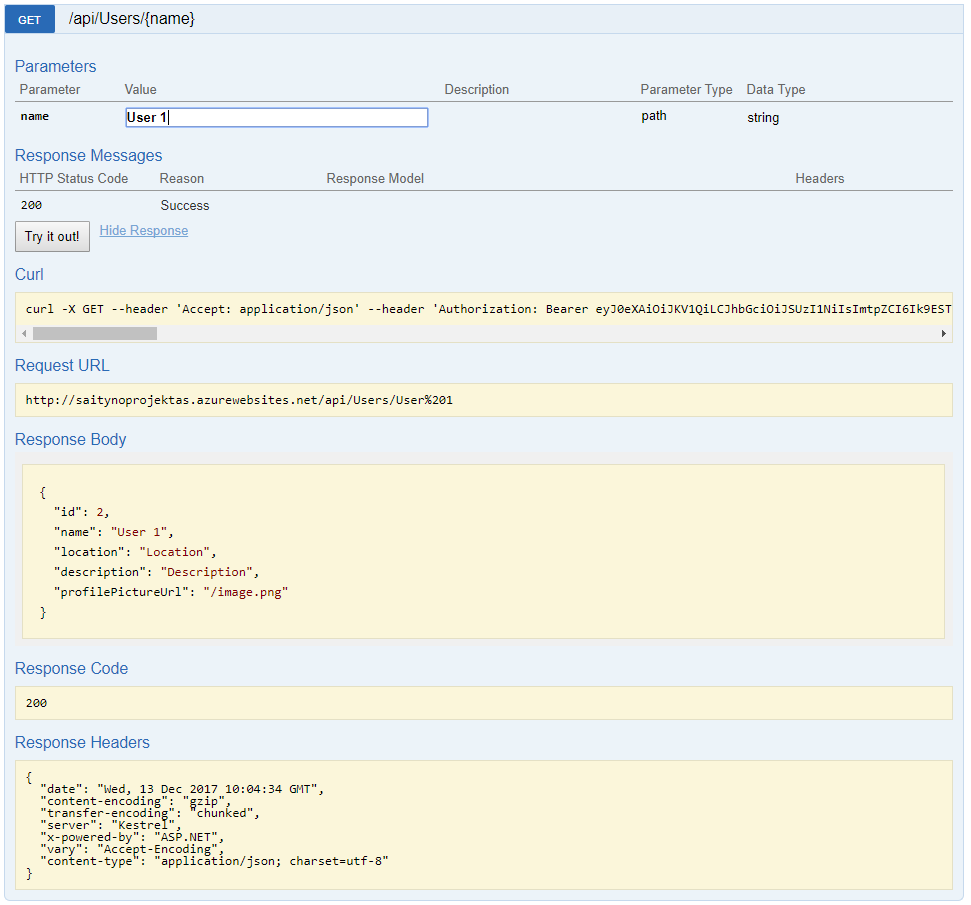
39 pav. Visų sistemoje esančių vartotojų gavimo pavyzdys

40 pav. pavaizduotas GET metodas, kuris grąžina šiuo metu prisijungusio vartotojo duomenis. Prisijungęs vartotojas yra sužinomas iš paduodamo prieigos tokeno.



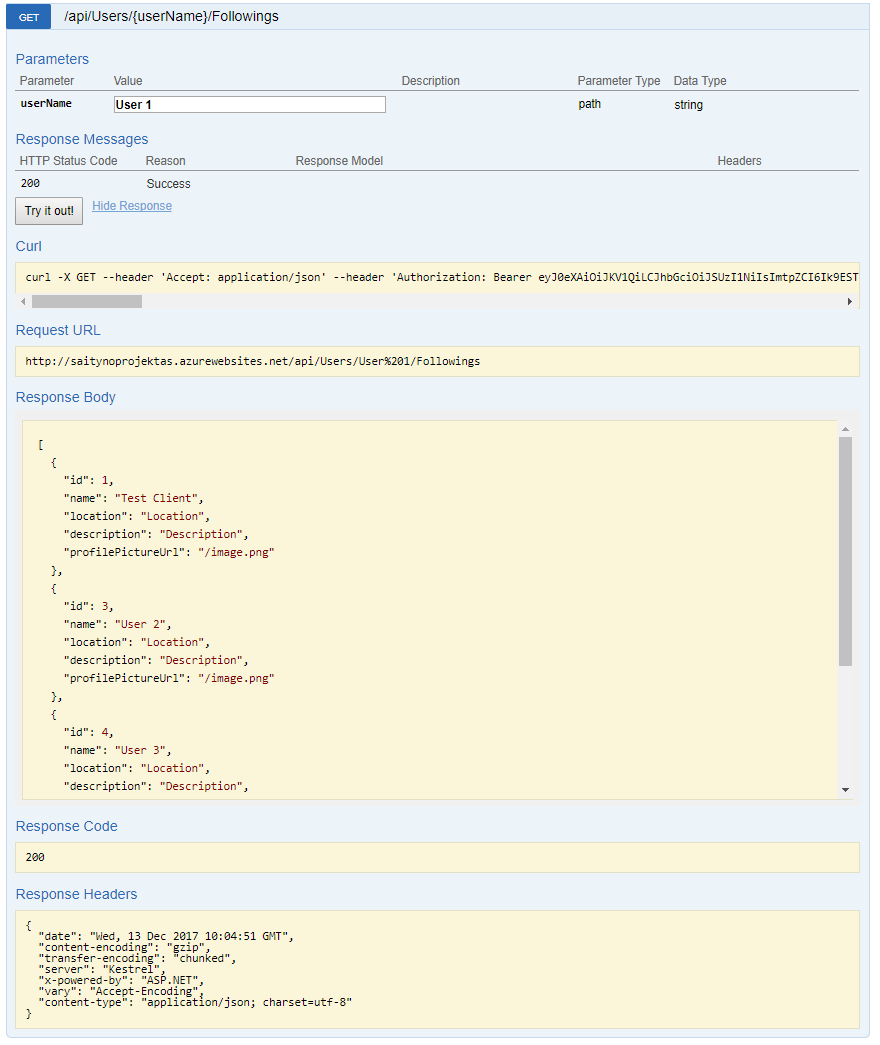
40 pav. Šiuo metu prisijungusio vartotojo duomenų gavimo pavyzdys

41 pav. pavaizduotas metodas grąžina nurodyto vardo vartotojo duomenis. Priima vieną parametrą – vartotojo vardą.



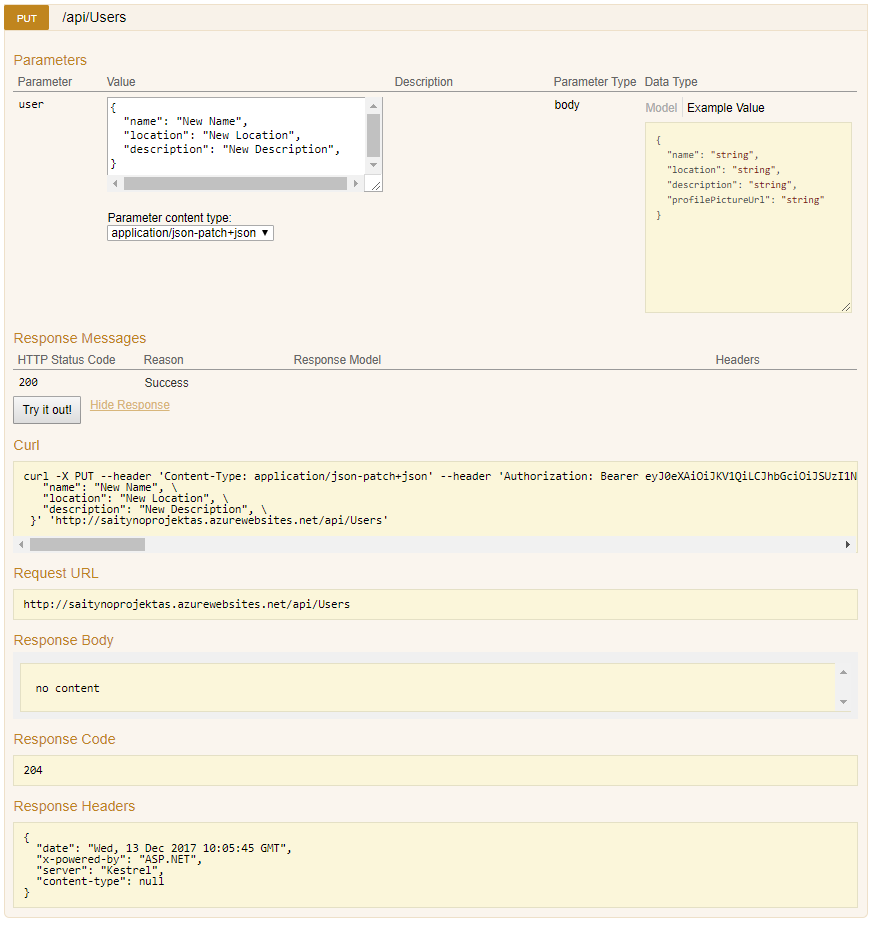
41 pav. Nurodyto vartotojo duomenų gavimo pavyzdys

42 pav. pavaizduotas GET metodas grąžina nurodyto vartotojo visų sekamų vartotojų sąrašą. Šių sekamų vartotojų įvykiai yra matomi naujienų puslapyje. Metodas priima tik vieną parametrą – vartotojo vardą.



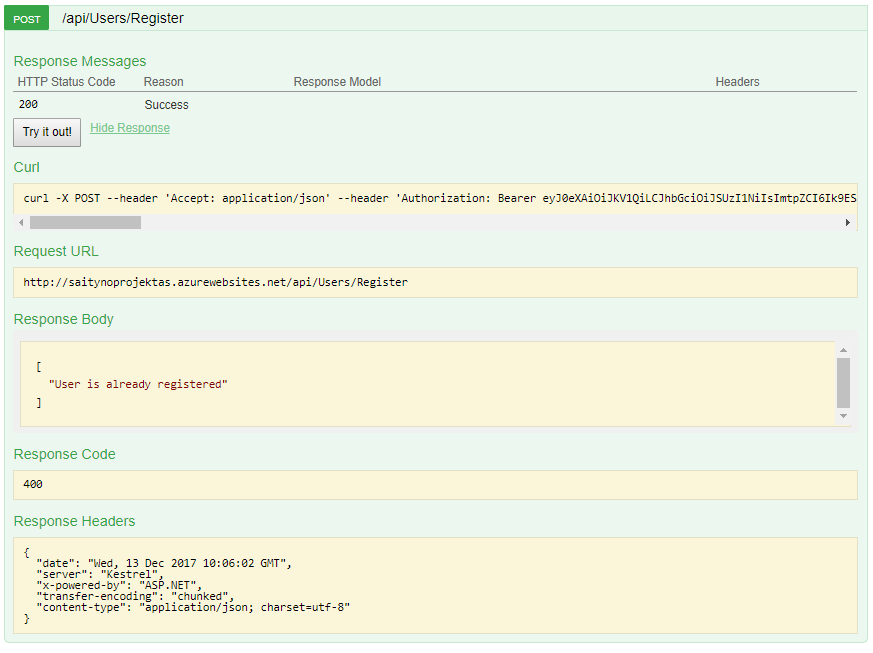
42 pav. Tam tikro vartotojo sekamų vartotojų sąrašo gavimo pavyzdys

43 pav. pavaizduotas PUT metodas, kuris paredaguoja prisijungusio vartotojo duomenis. Prisijungęs vartotojas yra nustatomas iš pateikiamo prieigos tokeno. Metodas priima tris parametrus: vartotojo vardą, vietą ir aprašymą. Kviečiant metodą neprivaloma nurodytų visų šių trijų parametrų. Bus pakeičiami tik tie duomenys, kurie bus pateikiami.



43 pav. Prisijungusio vartotojo duomenų redagavimo pavyzdys

44 pav. pavaizduotas vartotojo registracijos POST metodas. Paveikslėlyje pavaizduotas atvejis, kai bandoma užregistruoti vartotoją, kuris jau yra užregistruotas. Šis metodas yra kviečiamas iš karto, kai Auth0 autorizacijos serveryje yra užregistruojamas naujas vartotojas. Sukurtas vartotojas gauna numatytuosius vartotojo duomenis, kuriuos vėliau galima pakeisti.



44 pav. Registracijos pavyzdys

# Išvados

Projektinis darbas atliktas sėkmingai, tačiau dalis funkcionalumo nebuvo įgyvendinta dėl laiko stokos. Darant projektinį darbą buvo susipažinta su vieno puslapio aplikacijos kūrimu, kuriama kliento bei serverio dalis.

Pirmą kartą teko naudoti Auth0 autentifikacijos serverį. Įrankis pasirodė paprastas ir patogus naudoti, kad būtų įgyvendinta autentifikacija bei autorizacija.

Buvo pastebėta, kad didžiąją laiko dalį užtruko sukurti kliento aplikaciją, nors ir turima patirties naudojant kliento pusės bibliotekas React bei Redux. MockFlow įrankis labai padėjo kuriant abstrakčią kliento dalies išvaizdą, tačiau didelę laiko dalį užėmė grafinės sąsajos kūrimas su Semantic UI React biblioteka. Kūrimo metu buvo pastebėta, kad Redux bibliotekos naudojimas pastebimai sulėtina aplikacijos kūrimą. Ši biblioteka reikalauja rašyti daug papildomo kodo. Dėl šios priežasties sulėtėja naujo funkcionalumo kūrimas. Didelis Redux pastebėtas pliusas – aiški ir gerai supranta kliento aplikacijos struktūra.

Kuriant serverio dalį nepastebėta didelių sunkumų, kadangi C# programavimo kalbą bei ASP.net Core karkasą teko naudoti prieš darant šį projektą. Daugiausiai laiko užtruko sukurti tinkamus API metodus skirtus failo įkėlimui bei parsisiuntimui.