

Liepājas Valsts Tehnikums

**Music Player**

Kvalifikācijas eksāmena praktiskās daļas dokumentācija

Profesionālā kvalifikācija Programmēšanas tehniķis

Grupas nosaukums 4PT

Projekta izstrādātājs Annija Zorgenberga

/vārds, uzvārds, paraksts/

Eksāmena datums 2024. gada 21. jūnijs

Liepāja 2024

**Saturs**

[Ievads 3](#_Toc162189995)

[1. Uzdevuma formulējums 4](#_Toc162189996)

[2. Programmatūras prasību specifikācija 5](#_Toc162189997)

[2.1 Produkta perspektīva 5](#_Toc162189998)

[2.2 Sistēmas funkcionālās prasības 5](#_Toc162189999)

[2.3 Sistēmas nefunkcionālās prasības 16](#_Toc162190000)

[2.4 Gala lietotāja raksturiezīmes 16](#_Toc162190001)

[3. Izstrādes līdzekļu, rīku apraksts un izvēles pamatojums 17](#_Toc162190002)

[3.1 Izvēlēto risinājuma līdzekļu un valodu apraksts 17](#_Toc162190003)

[3.2 Iespējamo (alternatīvo) risinājuma līdzekļu un valodu apraksts 22](#_Toc162190004)

[4. Sistēmas modelēšana un projektēšana 26](#_Toc162190005)

[4.1 Sistēmas struktūras modelis 26](#_Toc162190006)

[4.2 ER diagramma 26](#_Toc162190007)

[4.3 Funkcionālais un dinamiskais sistēmas modelis 27](#_Toc162190008)

[4.4 Aktivitāšu diagramma 27](#_Toc162190009)

[4.5 Lietojumgadījumu diagramma 28](#_Toc162190010)

[4.6 Sistēmas moduļu apraksts un algoritmu shēmas 28](#_Toc162190011)

[5. Lietotāju ceļvedis 30](#_Toc162190012)

[6. Testēšanas dokumentācija 31](#_Toc162190015)

[6.1 Izvēlētās testēšanas metodes, rīku apraksts un pamatojums 34](#_Toc162190016)

[6.2 Testpiemēru kopa 34](#_Toc162190017)

[6.3 Testēšanas žurnāls 34](#_Toc162190018)

[7. Secinājumi 35](#_Toc162190019)

[8. Lietoto terminu saīsinājumu skaidrojumi 36](#_Toc162190020)

[9. Literatūras un informācijas avotu saraksts 37](#_Toc162190021)

# Ievads

Šis dokuments apraksta WEB tīmekļa vietni “Music Player”. Dokumentā ir iekļautas deviņas sadaļas. Sadaļas ir sekojošas - Uzdevuma formulējums, tajā tiek aprakstīts mazliet vairāk par veicamajiem darbiem un procesiem projekta izveidē, Programmas prasību specifikācija, kurā plašāk aprakstītas prasības projektam, Izstrādes līdzekļu, rīku apraksts, Sistēmas modelēšana un projektēšana, Lietotāju ceļvedis, Testēšanas dokumentācija, Secinājumi, Lietoto terminu saīsinājumu skaidrojumi un Literatūras un informācijas avotu saraksts. Vairākām sadaļām ir arī vairākas apakšnodaļas, kurās vēl dziļāk tiek aprakstīti konkrēti aspekti.

Music Player WEB tīmekļa vietne ir noderīga ne tikai mūzikas atskaņošanai, bet arī MP3 audio failu atskaņošanai. Viss tiek apvienots vienā vietā, ir iespēja gan meklēt populāras dziesmas, gan ievietot savus audio failus vai nolādētās dziesmas savos atskaņošanas sarakstos. Tā piedāvā iespēju pielāgot savu mūzikas klausīšanos pieredzi savām velmēm nodrošinot rekomendācijas no esošā mūzikas izklāsta balstoties uz dziesmām, kas ir visbiežāk atskaņotas vai pievienotas favorītu sadaļā balstoties ne tikai uz izpildītāju, bet arī uz žanru.

Primārie projekta mērķi ir nodrošināt lietotājam iespēju apvienot visas savas mūzikas un audio failu atskaņošanas vajadzības vienā vietā bez ierobežojumiem, piemēram, lietotājs var ne tikai atskaņot pieejamo WEB tīmekļa vietnes mūzikas izklāstu, bet arī pievienot savus audio failus un apvienot tos vienā atskaņošanas sarakstā. Bieži vien citās mūzikas atskaņošanas vietnēs ir konkrēts mūzikas izklāsts atkarībā no kurienes tas ir pievienots un lai klausītos savus audio failus to ir jādara iebūvētās aplikācijas savās ierīcēs, taču šeit lietotājs var ievietot savus failus, kas glabāsies konkrēti šī lietotāja profilā un netiks nekur publicēti. Tas, kas tiks augšuplādēts uz šo tīmekļa vietni būs pieejams tikai un vienīgi šim lietotājam.

WEB tīmekļa vietne iekļauj datu bāzi, kas ļauj lietotājiem augšuplādēt, saglabāt un atlasīt savus audio failus. Tas nozīmē, ka lietotāji var augšuplādēt un saglabāt savus audio failus un tie paliks lietotāja profilā līdz brīdim, kad lietotājs to izdzēsīs. Lietotājam nav iespēja izdzēst tos audio failus, kurus lietotājs pats neaugšuplādēja, taču vienmēr ir iespēja izlaist kādu dziesmu vai noņemt to no saviem atskaņošanas sarakstiem, ja to vairs negrib dzirdēt.

Galvenā mērķauditorija ir ikviens, kurš vēlas atskaņot visu savu mūziku un audio failus vienā vietā, it sevišķi ja ikdienā mūzika ir svarīga un ir svarīgi atskaņot konkrēto mūziku konkrētai dzīves situācijai balstoties uz žanru, mūzikas izpildītāju vai vienkārši uz to, kas ir augšuplādēts.

# Uzdevuma formulējums

Galvenais uzdevums ir izveidot WEB tīmekļa vietni “Music Player”, piedāvā lietotājiem iespēju izpausties un personalizēt savu mūzikas klausīšanas pieredzi nododot pilnu kontroli lietotājam par to, ko tas atskaņo un dodot iespēju vienmēr pievienot vēl audio failus, ja WEB tīmekļa vietnē nav iekļauts tas, ko lietotājs meklē.

Tīmekļa vietnē jābūt iekļautai klienta pusei, kurā ietilps lietotāja saskarsne, audio failu attēlošana, atskaņošana utt., un servera pusei, kurā ietilps datubāze, kas saglabās un apstrādās lietotāju augšupielādētos audio failus, preferences un visus pārējos datus, kas nepieciešami WEB tīmekļa vietnes darbībai.

Lietotāja saskarsnē ir jābūt attiecīgajām sadaļām, kas vajadzīgas, lai attēlotu mūziku un audio failus, lietotāja atskaņošanas sarakstus, mīļākās dziesmas, kā arī rekomendācijas, klausīšanās vēsturi un tīri tikai augšupielādētās dziesmas, kur var tos arī izdzēst. Ir jābūt arī ielogošanās sistēmai un lietotāja profilam, kurā lietotājs var veikt izmaiņas, ja tas ir nepieciešams. Ekrāna apakšā ir jābūt slīdnim, kas norāda dziesmas garumu, iespējai pārvietoties starp konkrēto dziesmu un starp citām dziesmām (uz priekšu un atpakaļ), kā arī tās pievienot atskaņošanas sarakstam un mīļāko dziesmu sarakstam. Vajag izveidot lietotāju saskarni, kas ir viegli izprotama un sniedz lietotājiem iespēju ērti un efektīvi kontrolēt savu klausīšanās pieredzi.

Papildus galvenajam mērķim, kurā ietilpst vieta, kurā ir nodrošināts visus audio failus klausīties vienā vietā, WEB tīmekļa vietnes centrā ir arī nodrošināt lietotājiem pilnīgu klausīšanās personalizāciju, kurā ietilpst dziesmu rekomendācijas balstoties uz iepriekš vai visbiežāk klausītajām dziesmu izpildītājiem un žanriem, kā arī iespēja pielāgot visdažādākos atskaņošanas sarakstus atbilstoši viņu individuālajām vajadzībām. Šis mērķis tiecas nodrošināt lietotājiem ne tikai standarta vietu, kur atskaņot audio, bet arī papildināt savus atskaņošanas sarakstus, atklāt jaunas dziesmas. Piemēram, lietotājs var noklausīties vienu dziesmu vairākas reizes vai pievienot to savām mīļākajām dziesmām un tad rekomendāciju lapā, balstoties no šīs dziesmas, lietotājam parādīs līdzīgas dziesmas balstoties uz vairākiem faktoriem.

Pagaidām tīmekļa vietnes dziesmu pamatā, kas nav audio faili, ko pats lietotājs augšupielādējis tiek izmantots Spotify API, kas nodrošina plašu dziesmu klāstu, lai lietotājs varētu no plaša dziesmu izklāsta meklēt sev piemērotās.

Svarīgi ir arī nodrošināt drošību datubāzē, lai lietotājs var būt drošs, ka paša augšuplādētie audio faili netiks izplatīti vai publicēti un būs pieejami tikai konkrētajam lietotājam šī lietotāja profilā.

# Programmatūras prasību specifikācija

Šajā nodaļā tiks aprakstītas programmatūras prasību specifikācijas WEB tīmekļa vietnei “Music Player”. Šajā nodaļā tiek iekļautas četras apakšnodaļas, kuras satur produkta perspektīvu, funkcionālās un nefunkcionālās prasības un gala lietotāja raksturiezīmes. Produkta perspektīvā būs dziļāk aprakstīts par pašu projektu un tā iespējām atkarībā no attiecīgās puses. Funkcionālās prasības sadaļa aprakstīs sistēmas pamata funkcijas un darbība, ar kuru palīdzību WEB tīmekļa vietne darbosies. Sistēmas nefunkcionālās prasības sevī iekļaus prasības, kas neattiecas uz programmatūras darbību un funkcijām. Pēdējā apakšnodaļa, gala lietotāja raksturiezīmes, dziļāk aprakstīs produkta mērķauditoriju un WEB tīmekļa vietnes lietotājus.

## Produkta perspektīva

Produkts “Music Player” ir WEB tīmekļa vietne, kas izstrādāta, lai nodrošinātu lietotāju ar ērtu un personalizētu pieredzi klausoties mūziku un savus audio failus vienā vietā. Tīmekļa vietne ir lietotājiem draudzīga, piedāvājot skaidru izklāstu par pieejamām sadaļām un iespējām, ko tā sniedz. Izmantojot šo produktu tiek piedāvāts ne tikai vieglāks veids, kā klausīties visu vienā vietā, bet arī daudz personalizētāks, jo tas dod iespēju lietotājam pārvaldīt savus atskaņošanas sarakstus, savus audio failus, kā arī izvēlēties no plaša rekomendāciju izklāsta balstoties uz to, ko lietotājs ikdienā klausās. Salīdzinot ar parastajām mazāka apmēra mūzikas atskaņošanas vietnēm, kur ir iespēja izvēlēties tikai no konkrēti pieejamām dziesmām un bieži vien nav iespējams atrast dziesmu, kuru meklē, šajā tīmekļa vietnē lietotājs pats var ievietot savus audio failus un iekļaut tos savos atskaņošanas sarakstos nevis visu laiku iet uz savu lokālo failu glabātuvi.

Salīdzinot šo produktu ar citiem tam līdzīgiem produktiem, ir konkurences produkti, kas sniedz pakalpojumus ļoti lielā apmērā ar šīm līdzīgām funkcijām, taču tie ir parasti maksas produkti un lietotājam nav iespēja izmantot visas šīs funkcijas nemaksājot. Zemāka apmēra mūzikas atskaņošanas aplikācijas sniedz līdzīgas iespējas, tomēr parasti nav iespēja ievietot arī savus audio failus un piekļūt tiem no jebkuras ierīces un vietas pasaulē izmantojot savu lietotāja profilu.

## Sistēmas funkcionālās prasības

**PR.01. Autentifikācija**

Mērķis:

Ļaut lietotājam autentificēties WEB vietnē.

Ievaddati:

Nosacījumi, lai autentificētos:

**1. tabula**

|  |  |  |
| --- | --- | --- |
| Nosaukums | Obligāts | Piezīmes |
| Lietotājvārds | Jā | Max. 80 rakstzīmes, unikāls |
| Parole | Jā | Max. 60 rakstzīmes, droša šifrēšana, droša parole |

1. Aizpildīts lauks “Lietotājvārds”
2. Aizpildīts lauks “Parole”
3. Nospiesta poga “Ielogoties”

Apstrāde:

Programma pārbauda, vai lietotājvārds un parole sakrīt ar datiem datubāzē.

Izvaddati:

1. Lietotājs tiek autentificēts tīmekļa vietnes sistēmā ar savu profilu;
2. Tiek izvadīts kļūdas ziņojums par neatbilstošu vai tukšu lietotājvārdu;
3. Tiek izvadīts kļūdas ziņojums par neatbilstošu vai tukšu paroli;

**PR.02. Pārslēgšanās starp sadaļām**

Mērķis:

Ļaut lietotājam pārriet no vienas sadaļas un citu

Ievaddati:

Nosacījumi, lai pārrietu no vienas sadaļas un citu:

1. Veiksmīga autorizācija
2. Noklikšķināts uz konkrēto sadaļu

Apstrāde:

Programma pārbauda uz kuru sadaļu tika noklikšķināts.

Izvaddati:

1. Tiek atvērta konkrētā sadaļa uz ko lietotājs uzklikšķināja;
2. Kļūdas paziņojums par atļaujām.

**PR.03. Mūzikas atskaņošana**

Mērķis:

Atskaņot dziesmu uz kuras tika uzklikšķināts

Ievaddati:

Nosacījumi, lai atskaņotu audio failu:

1. Lietotājs noklikšķinājis uz konkrētu dziesmu;

Apstrāde:

Programma pārbauda uz kuru dziesmu tika noklikšķināts un apstrādā audio faila meta datus.

Izvaddati:

1. Tiek atskaņots konkrētais audio fails;
2. Kļūdas paziņojums par dziesmas nepieejamību.

**PR.04. Pievienošana atskaņošanas sarakstam**

Mērķis:

Ļaut lietotājam pievienot konkrēto dziesmu kādam atskaņošanas sarakstam.

Ievaddati:

Nosacījumi, lai pievienotu dziesmu atskaņošanas sarakstam:

1. Uz dziesmas nospiesta pievienot atskaņošanas sarakstam poga;
2. Izvēlēts atskaņošanas saraksts, kuram dziesmu pievienot;

Apstrāde:

Programma pārbauda, kura dziesma tika izvēlēta un kurš atskaņošanas saraksts tika izvēlēts un attiecīgi to pievieno atskaņošanas sarakstam.

Izvaddati:

1. Paziņojums par veiksmīgu pievienošanu un dziesma parādās atskaņošanas sarakstā;
2. Apstiprinošs paziņojums par to, ka dziesma jau ir atskaņošanas sarakstā un vai lietotājs vēlreiz to grib pievienot;
3. Kļūdas paziņojums, ka neizdevās pievienot dziesmu atskaņošanas sarakstam.

**PR.05. Izveidot atskaņošanas sarakstu**

Mērķis:

Ļaut lietotājam izveidot jaunu atskaņošanas sarakstu

Ievaddati:

Nosacījumi, lai izveidotu jaunu atskaņošanas sarakstu:

1. Nospiesta poga izveidot jaunus atskaņošanas sarakstu;
2. Aizpildīts lauks “Nosaukums”;
3. Pēc izvēles aizpildīts lauks “Apraksts”;
4. Pēc izvēles ievietota atskaņošanas saraksta priekšbilde.

Apstrāde:

Programma pievieno jaunu atskaņošanas sarakstu ar ievadīto nosaukumu, aprakstu un priekšbildi.

Izvaddati:

1. Paziņojums par veiksmīgu atskaņošanas saraksta izveidi;
2. Kļūdas paziņojums par tukšu vai nekorektu atskaņošanas saraksta nosaukumu;
3. Kļūdas paziņojums par neveiksmīgu izveidi.

**PR.06. Rediģēt atskaņošanas sarakstu**

Mērķis:

Rediģēt kādu no esošajiem atskaņošanas sarakstiem.

Ievaddati:

Nosacījumi, lai rediģētu kādu no esošajiem atskaņošanas sarakstiem:

1. Nospiesta poga rediģēt sarakstu uz konkrēto atskaņošanas sarakstu;
2. Aizpildīts lauks “Nosaukums”;
3. Pēc izvēles aizpildīts lauks “Apraksts”;
4. Pēc izvēles izvēlēta atskaņošanas saraksta priekšbilde.

Apstrāde:

Programma pārbauda kurš atskaņošanas saraksts tiek rediģēts un attiecīgi tiek izmainītas mainītās vērtības.

Izvaddati:

1. Paziņojums par veiksmīgu atskaņošanas saraksta rediģēšanu;
2. Kļūdas paziņojums par tukšu vai nekorekti aizpildītu lauku “Nosaukums”;
3. Kļūdas paziņojums par neveiksmīgu rediģēšanos.

**PR.07. Noņemšana no atskaņošanas saraksta**

Mērķis:

Ļaut lietotājam noņemt dziesmu no kāda atskaņošanas saraksta.

Ievaddati:

Nosacījumi, lai noņemt dziesmu no kāda atskaņošanas saraksta:

1. Konkrētajā atskaņošanas sarakstā noklikšķināts uz noņemt dziesmu no atskaņošanas saraksta pogas;

Apstrāde:

Programma pārbauda uz kuru atskaņošanas sarakstu un kuru dziesmu tika noklikšķināts.

Izvaddati:

1. Paziņojums par veiksmīgu dziesmas noņemšanu no atskaņošanas saraksta;
2. Kļūdas paziņojums par neveiksmīgu dziesmas noņemšanu no atskaņošanas saraksta.

**PR.08. Reģistrācija**

Mērķis:

Reģistrēt jaunu lietotāja kontu sistēmā.

Ievaddati:

**2. tabula**

|  |  |  |
| --- | --- | --- |
| Nosaukums | Obligāts | Piezīmes |
| Lietotājvārds | Jā | Max 80. rakstzīmes, unikāls |
| Parole | Jā | Max. 60 rakstzīmes, droša šifrēšana, droša parole |
| Atkārtot paroli | Jā | Max. 60 rakstzīmes, droša šifrēšana, droša parole |

Nosacījumi, lai reģistrētu lietotāja kontu sistēmā:

1. Nospiesta poga “Reģistrēties”;
2. Aizpildīts lauks “Lietotājvārds”;
3. Aizpildīts lauks “Parole”;
4. Aizpildīts lauks “Atkārtot paroli”.

Apstrāde:

Programma pārbauda vai ir aizpildīti visi trīs lauki un vai lauks “Parole” un “Atkārtot paroli” sakrīt, kā arī, ka lietotājvārds jau neeksistē.

Izvaddati:

1. Nospiesta poga “Reģistrēties”;
2. Aizpildīts lauks “Lietotājvārds”;
3. Aizpildīts lauks “Parole”;
4. Aizpildīts lauks “Atkārtot paroli”.

**PR.09 Paroles maiņa**

Mērķis:

Ļaut lietotājam nomainīt savu paroli.

Ievaddati:

**3. tabula**

|  |  |  |
| --- | --- | --- |
| Nosaukums | Obligāts | Piezīmes |
| Esošā parole | Jā | Max. 60 rakstzīmes, droša šifrēšana, drošas paroles nosacījumi |
| Jaunā parole | Jā | Max. 60 rakstzīmes, droša šifrēšana, drošas paroles nosacījumi |

Nosacījumi, lai iestatītu gaismas izslēgšanas grafiku:

1. Nospiesta poga “Mainīt paroli”;
2. Aizpildīts lauks “Esošā parole”;
3. Aizpildīts lauks “Jaunā parole”.

Apstrāde:

Programma pārbauda, vai esošā parole ir aizpildīta pareizi un vai jaunā parole atbilst paroles kritērijiem.

Izvaddati:

1. Paziņojums par veiksmīgu paroles maiņu;
2. Paziņojums par tukšu vai nekorektu esošo paroli;
3. Paziņojums par tukšu vai nekorektu jauno paroli.

**PR.10. Filtrēšana pēc nosaukuma**

Mērķis:

Ļaut lietotājam filtrēt dziesmas pēc nosaukuma.

Ievaddati:

Nosacījumi, lai filtrētu dziesmas pēc nosaukuma:

1. Ievadīts dziesmas nosaukums;
2. Nospiesta poga “Meklēt”.

Apstrāde:

Programma nolasa dziesmas nosaukumu un meklē dziesmas, kuru nosaukumos tas ir iekļauts gan Spotify API, gan augšuplādētajos failos.

Izvaddati:

1. Atlasās audio faili ar ievadīto nosaukumu;
2. Paziņojums par to, ka tādi audio faili nav;
3. Kļūdas paziņojums par neveiksmīgu meklēšanu.

**PR.11. Atskaņošanas saraksta dzēšana**

Mērķis:

Ļaut lietotājam dzēst savus atskaņošanas sarakstus.

Ievaddati:

Nosacījumi, lai dzēstu atskaņošanas sarakstu:

1. Izvēlēts dzēšamais atskaņošanas saraksts;
2. Nospiesta poga “Dzēst”;
3. Nospiesta poga “Jā” apstiprinājuma logā.

Apstrāde:

Programma pārbauda, kurš saraksts tika izvēlēts un to izdzēš no datubāzes.

Izvaddati:

1. Paziņojums par veiksmīgu dzēšanu;
2. Kļūdas paziņojums par neveiksmīgu dzēšanu.

**PR.12. Filtrēšana pēc izpildītāja**

Mērķis:

Ļaut lietotājam meklēt/filtrēt dziesmas pēc izpildītāja.

Ievaddati:

Nosacījumi, lai filtrētu dziesmas pēc izpildītāja:

1. Ievadīts izpildītāja nosaukums;
2. Nospiesta poga “Meklēt”.

Apstrāde:

Programma nolasa izpildītāju un attiecīgi filtrē gan caur Spotify API, gan pašaugšuplādētajiem audio failiem.

Izvaddati:

1. Atlasās audio faili ar ievadīto nosaukumu izpildītājā;
2. Paziņojums par to, ka tādi audio faili nav;
3. Kļūdas paziņojums par neveiksmīgu meklēšanu.

**PR.13. Izlogošanās**

Mērķis:

Izlogoties no esošā lietotāja profila.

Ievaddati:

Nosacījumi, lai izrakstītos no esošā lietotāja profila:

1. Lietotājs ir autorizējies;
2. Nospiesta poga “Izlogoties”.

Apstrāde:

Programma izraksta lietotāju un pāriet uz autorizācijas sākumlapu.

Izvaddati:

1. Parādās sākumlapa ar pieprasījumu autorizēties.

**PR.14. Konta dzēšana**

Mērķis:

Izdzēst lietotāja kontu no sistēmas

Ievaddati:

Nosacījumi, lai lietotājs varētu izdzēst savu kontu:

1. Nospiesta poga “Izdzēst kontu”;
2. Aizpildīts lauks “Parole”;
3. Nospiesta poga “Jā” uz apstiprinājuma pieprasījuma.

Apstrāde:

Programma pārbauda vai ievadītā parole ir korekta un vai lietotājs apstiprinājis konta dzēšanu.

Izvaddati:

1. Paziņojums par veiksmīgu konta dzēšanu un ielādē sākumlapu;
2. Paziņojums par tukšu vai nekorektu paroli.

**PR.15. Konta statusa maiņa**

Mērķis:

Iestatīt lietotāja kontu uz aktīvu vai neaktīvu,

Ievaddati:

Nosacījumi, lai lietotājs nomainītu konta statusu:

1. Nospiesta poga “Statusa maiņa”;
2. Izvēlēts aktivizēt vai deaktivizēt.

Apstrāde:

Programma pārbauda kurš statuss kontam jāuzliek.

Izvaddati:

1. Konta status tiek iestatīts uz “Aktīvs”;
2. Konta statuss tiek iestatīts uz “Neaktīvs”

**PR.16. Lietotājvārda maiņa**

Mērķis:

Nomainīt lietotāja konta lietotājvārdu.

Ievaddati:

Nosacījumi, lai nomainītu lietotāja konta lietotājvārdu:

1. Nospiesta poga “Mainīt lietotājvārdu”
2. Aizpildīts lauks “Jaunais lietotājvārds”

Apstrāde:

Programma pārbauda vai lietotājvārds nav tukšs un ir aizpildīts korekti. Izvaddati:

1. Paziņojums par veiksmīgu lietotājvārda maiņu;
2. Paziņojums par tukšu vai nekorektu lietotājvārdu;
3. Paziņojums par to, ka tāds lietotājvārds jau eksistē.

**PR.17. Sava audio faila augšupielāde**

Mērķis:

Ļaut lietotājam augšupielādēt savus audio failus.

Ievaddati:

Nosacījumi, lai augšupielādētu savus audio failus:

1. Augšupielādes sadaļā nospiesta poga “Pievienot failu”
2. Tiek izvēlēts MP3 fails;
3. Nospiesta poga “Augšupielādēt”.

Apstrāde:

Programma pārbauda vai fails, kas tika augšupielādēts ir drošs un pareizais faila tips, tad izvelk saglabāšanai vajadzīgos datus un pievieno tos datubāzei.

Izvaddati:

1. Paziņojums par veiksmīgu pievienošanos un audio fails parādās lietotājam;
2. Paziņojums par nepareizu vai nedrošu failu;
3. Kļūdas paziņojums par neveiksmīgu faila pievienošanu.

**PR.18. Sava audio faila dzēšana**

Mērķis:

Ļaut lietotājam dzēst savus audio failus.

Ievaddati:

Nosacījumi, lai dzēstu savus audio failus:

1. Savu failu sadaļā izvēlēts konkrēts fails;
2. Nospiesta poga “Dzēst”;
3. Nospiesta poga “Jā” uz apstiprinājuma ziņojuma.

Apstrāde:

Programma pārbauda kurš fails tika izvēlēts un attiecīgi to izdzēš no datubāzes (tajā skaitā arī no atskaņošanas sarakstiem).

Izvaddati:

1. Paziņojums par veiksmīgu dzēšanu;
2. Paziņojums par neveiksmīgu dzēšanu.

**PR.19 Klausīšanās vēsture**

Mērķis:

Attēlot lietotājam savu klausīšanās vēsturi.

Ievaddati:

Nosacījumi, lai attēlotu klausīšanās vēsturi:

1. Nospiesta poga “Vēsture”;
2. Lietotājs ir klausījies vismaz vienu audio failu;

Apstrāde:

Programma atlasa visas dziesmas, ko šobrīd ielogojies lietotājs ir klausījies un attēlo to nosaukumus un izpildītājus vēstures sadaļā.

Izvaddati:

1. Saraksts ar klausīšanās vēsturi;

**PR.20 Pievienošana favorītiem**

Mērķis:

Ļaut lietotājam pievienot kādu dziesmu mīļāko dziesmu sadaļai.

Ievaddati:

Nosacījumi, lai pievienotu dziesmu mīļāko dziesmu sadaļai:

1. Izvēlēta konkrēta dziesma;
2. Nospiesta zvaigznītes poga ar tooltipu “Favorīti”;

Apstrāde:

Programma pārbauda, kura dziesma tā ir un pievieno to mīļāko dziesmu sarakstam.

Izvaddati:

1. Paziņojums par veiksmīgu pievienošanu mīļāko dziesmu sarakstam un attēlojums favorītu sadaļā;
2. Kļūdas paziņojums par neveiksmīgu pievienošanos.

**PR.21. Filtrēšana pēc žanra**

Mērķis:

Ļaut lietotājam meklēt/filtrēt dziesmas pēc izpildītāja.

Ievaddati:

Nosacījumi, lai filtrētu dziesmas pēc izpildītāja:

1. Ievadīts izpildītāja nosaukums;
2. Nospiesta poga “Meklēt”.

Apstrāde:

Programma nolasa izpildītāju un attiecīgi filtrē gan caur Spotify API, gan paša augšupielādētajiem audio failiem.

Izvaddati:

1. Atlasās audio faili ar ievadīto nosaukumu izpildītājā;
2. Paziņojums par to, ka tādi audio faili nav;
3. Kļūdas paziņojums par neveiksmīgu meklēšanu.

**PR.22. Pauzes poga**

Mērķis:

Ļaut lietotājam nopauzēt šī brīža dziesmu ar pauzes pogu.

Ievaddati:

Nosacījumi, lai nopauzētu šī brīža dziesmu ar pauzes pogu:

1. Tiek spēlēta kāda dziesma;
2. Nospiesta poga “Pauze”.

Apstrāde:

Programma nolasa cik tālu dziesmā lietotājs tika un kuru dziesmu jānopauzē.

Izvaddati:

1. Dziesma tiek nopauzēta šī brīža progresā.

**PR.23. Nākamā dziesma**

Mērķis:

Ļaut lietotājam aiziet uz nākamo dziesmu atskaņošanas sarakstā izmantojot “Nākamā” pogu.

Ievaddati:

Nosacījumi, lai aizietu uz nākamo dziesmu atskaņošanas sarakstā:

1. Tiek spēlēta kāda dziesma;
2. Nospiesta poga “Nākamā”.

Apstrāde:

Programma nolasa kurā atskaņošanas sarakstā šobrīd skan dziesma un sāk spēlēt nākamo.

Izvaddati:

1. Sāk atskaņot nākamo dziesmu.

**PR.24. Iepriekšējā dziesma**

Mērķis:

Ļaut lietotājam aiziet uz iepriekšējo dziesmu atskaņošanas sarakstā izmantojot “Atpakaļ” pogu.

Ievaddati:

Nosacījumi, lai aizietu atpakaļ uz iepriekšējo dziesmu:

1. Tiek spēlēta kāda dziesma;
2. Nospiesta poga “Atpakaļ”.

Apstrāde:

Programma pārbauda kurā atskaņošanas sarakstā šobrīd skan dziesma un sāk spēlēt iepriekšējo.

Izvaddati:

1. Sāk atskaņot iepriekšējo dziesmu.

## Sistēmas nefunkcionālās prasības

1. Tīmekļa vietnei jābūt skaidrā un saprotamā Angļu valodā;
2. Tīmekļa vietnei saskarnei jābūt intuitīvai un viegli saprotamai;
3. Tīmekļa vietnei sadaļu izkārtojumam ir jābūt saprotamam un ērtam;
4. Tīmekļa vietnei krāsām ir jābūt ar violetiem un tumšiem toņiem, gaišu tekstu;
5. Tīmekļa vietnei jāiekļauj React.js bibliotēku, MySQL datubāzi un PHP;.
6. Tīmekļa vietnes atbildes laikam ir jābūt zem 5000ms;
7. Tīmekļa vietnes izstrādes termiņš ir 2024. gada 20. Jūnijs;
8. Atbilstība Latvijas Valsts standarta “Programmatūras prasību specifikācijas ceļvedis” (LVS 68:1996) dokumentam;

## Gala lietotāja raksturiezīmes

Mūzikas atskaņotāja tīmekļa vietnes galvenie lietotāji ir visi, kas vēlas klausīties mūziku un augšupielādēt savus audio failus vienuviet. Lietotāji var būt ar dažādām tehnoloģiju prasmēm, bet mūsu vietnes saskarne būs ērti izmantojama un intuitīva, prasot minimālas tehniskās zināšanas. Tie, kas ir vairāk ieinteresēti tehnoloģijās, var būt vairāk gatavi izmēģināt mūsu vietni, kā arī indivīdi, kuri meklē ērtības, vienkāršojot savu mūzikas pieredzi. Turklāt mūsu vietne var būt īpaši noderīga tiem, kas vēlas glabāt un pārvaldīt savus audio failus vienuviet, bez papildu lietojumprogrammām.

Attiecībā uz lietotāju lomām vietnē, ir administrators, kas nodrošina palīdzību lietotājam sakarā ar augšupielādēto failu problēmām un lietotāju profilu pārvaldīšanu, ja tas ir nepieciešams.

# Izstrādes līdzekļu, rīku apraksts un izvēles pamatojums

Šajā nodaļā tiks aprakstīti “Music Player” WEB vietnes izstrādes līdzekļi un rīki, kā arī pamatojums, kādēļ tieši šos rīkus vēlējos izmantot. Tīmekļu vietņu izstrādē ir pieejami daudz un dažādi līdzekļi un rīki, lai atvieglotu un optimizētu izstrādes procesu, tādēļ apakšnodaļā tiks arī aprakstīti vairāki alternatīvie risinājuma līdzekļi un valodas.

## Izvēlēto risinājuma līdzekļu un valodu apraksts

Izstrādes līdzekļi un valodas, kuras tika izmantotas “Music Player” tīmekļa vietnē.

**3.1.1 Visual Studio code**

Apraksts - Visual Studio Code ir Microsoft izstrādāts pirmkoda redaktors.

Plusi:

* Atbalsts atkļūdošanai,
* Sintakses izcelšana,
* Inteliģenta koda pabeigšanai, fragmentiem, koda pārveidošanai u.c.
* Var mainīt preferences
* Var nolādēt paplašinājumus, kas papildina funkcionalitāti.

Tieši tādēļ es izvēlējos šo koda redaktoru, jo tas piedāvā ļoti daudz funkcijas un paplašinājumus, ko citi koda redaktori nē. Visual studio code ir ļoti populārs programmēšanas pasaulē, tādēļ arī interneta resursos var atrast daudz ko, ja rodas kāda neparasta kļūda utt. Kopumā tā ir pievilcīga un ērta koda izstrādes vide arī pēc dizaina izskata.

Mīnusi:

* Nepieciešama lielāka konfigurācija
* Resursu ietilpīgs
* Nav tik iesācējiem draudzīgs

**JavaScript**

Apraksts - JavaScript ir skriptēšanas jeb programmēšanas valoda, kas ļauj tīmekļa lapās ieviest sarežģītas funkcijas. Tā ir interpretēta valoda ar objektorientētām iespējām.

Plusi:

* Klienta puses nodrošināšanas iespējas
* Viegla valoda, kurai nav nepieciešama smaga programmatūras instalēšana
* Diezgan ātra
* Populāra

Ar to ir viegli apieties un visbiežāk tiek izmanto kā daļu no tīmekļa lapām, kuru ieviešana ļauj klienta puses skriptam mijiedarboties ar lietotāju un izveidot dinamiskas lapas. Es šo izvēlējos, jo gribēju implementēt šādas funkcijas savā tīmekļa vietnē un šis ir vispopulārākais un labākais variants apejoties ar klienta pusi tīmekļa vietnēs, jo ar to var paveikt ļoti plašas un sarežģītas funkcijas un aprēķinus.

Mīnusi:

* Drošības riski, ja paļaujas tikai uz klienta puses drošību
* Pārlūka neatbilstības
* Trūkst vairāku mantojuma funkciju
* Atkļūdošanas rīku trūkums

**React.js**

Apraksts – React.js ir atvērtā pirmkoda JavaScript bibliotēka, ko Facebook izstrādājis un kuras mērķis ir vienkāršot sarežģīto interaktīvo lietotāja saskarņu veidošanas procesu.

Plusi:

* Uz komponentiem balstīta arhitektūra
* Augsta veiktspēja
* Redux, kas nodrošina attēloto datu pārvaldi
* Viegli iemācīties

Es izvēlējos tieši React.js, jo prakses vietā diezgan plaši iepazinu React.js, kas man atviegloja tā integrāciju šajā vietnē. Manuprāt, ta ir ļoti efektīva bibliotēka un saprotamāk izdala kodu, datu ielādi un attēlošanu.

Mīnusi:

* Augsts attīstības temps
* Elastīgums un konvenciju trūkums
* Nav pilnvērtīgs ietvars (tā ir bibliotēka nevis ietvars jeb angliski framework)
* Slikta dokumentācija dēļ tā, cik ātri attīstās (ne vienmēr tiek laicīgi atjaunota)
* SEO jeb Search Engine Optimisation apstrādes riski sakarā ar meklētājprogrammu spēju uzskaitīt dinamiski ielādētu saturu

**CSS**

Apraksts - CSS (Cascading Style Sheets) ir vienkārši izstrādāta valoda, kas paredzēta, lai vienkāršotu procesu, kā padarīt tīmekļa lapu dizainu pievilcīgāku. CSS ļauj piemērot stilus HTML dokumentiem. Tas apraksta, kā tīmekļa lapai vajadzētu izskatīties. Es šo izvēlējos, jo tas ir kā standarts stilam veidojot tīmekļa vietni.

Plusi:

* Satura un stila atdalīšana
* Pastāvīgums
* Ātrāks ielādes laiks
* Responsive Design
* Apkopes vienkāršība

Mīnusi:

* Pārlūka saderības riski
* Mācību līkne
* Drošības trūkums
* Ierobežota izkārtojuma kontrole

**SQLite**

Apraksts - SQLite ir programmatūras bibliotēka, kas nodrošina relāciju datu bāzes pārvaldības sistēmu. SQLite nodrošina vieglu iestatīšanas, datu bāzes administrēšanas un nepieciešamo resursu ziņā.

Plusi:

* Viegls
* Labāka veiktspēja, tas ir gandrīz par 35% ātrāks nekā failu sistēma
* Nav nepieciešama instalēšana
* Pārnēsājams

Izvēlējos šo datubāzes sistēmu, jo serverim šād apmēra projektam pagaidām nebūtu vajadzības.

Mīnusi:

* Datu bāzes lielums vairumā gadījumu ir ierobežots līdz 2 GB
* Mazāk resursu tiem, kas vēlas to apgūt

**PHP**

Apraksts - PHP (Hypertext Processor) ir vispārēja lietojuma skriptu valoda, kas ir brīvi pieejama un plaši izmantota tīmekļa vietņu izstrādei. Valoda galvenokārt tiek izmantota servera puses skriptēšanai.

Plusi:

* Bezmaksas lietošana
* Kopienas atbalsts
* Lietošanas ērtums
* Ātrums

Es savām servera puses vajadzībām izvēlējos PHP, jo diezgan plaši to iepazinu Tīmekļa vietņu programmēšanas kursā un kaut vai man praksē bija pieredze ar C#, PHP man ir ērtāk izmantojams šāda apjoma tīmekļa vietnei. PHP arī ir pieejama plaša dokumentācija un viegla sintakse no tā, kas ir mācīts.

Mīnusi:

* Ne tik daudzpusīga
* Ierobežoti atkļūdošanas rīki
* Nevar mainīt galveno uzvedību
* Ir vieglāk lietojamas valodas

**Spotify API**

Apraksts - Spotify WEB API (Application Programming Interface) ļauj izveidot lietojumprogrammas, kas var mijiedarboties ar Spotify pakalpojumu, piemēram, izgūt satura metadatus, saņemt ieteikumus, izveidot un pārvaldīt atskaņošanas sarakstus vai kontrolēt atskaņošanu.

Plusi:

* Piekļuve plašajai mūzikas bibliotēkai
* Integrācijas iespējas
* Personalizēšana
* Kopiena un resursi

Spotify developers sadaļā no spotify izmantoju tā WEB API, lai būtu pieejams plašs mūzikas izklāsts tīmekļa vietnē, izvēlējos tieši šo, jo Spotify ir uzticams ar dokumentāciju un ļoti plašu pieejamo dziesmu klāstu.

Mīnusi:

* Likmes ierobežojumi
* Atkarība no Spotify
* Riski saistībā ar datu privātumu
* Ierobežota funkcionalitāte

**Git**

Apraksts - Git ir bezmaksas un atvērtā koda izplatīta versiju kontroles sistēma, kas paredzēta, lai ātri un efektīvi apstrādātu visu, sākot no maziem līdz ļoti lieliem projektiem.

Plusi:

* Atzaru veidošana un apvienošana
* Ātrums un veiktspēja
* Datu integritāte
* Sadarbība

Tieši tādēļ es to izvēlējos izmantot versiju kontrolei, jo tas ir plaši dokumentēts, apmācīts skolā un tam ir vairākas labas funkcionālās vērtības.

Mīnusi:

* Mācīšanās grūtības
* Sarežģītība
* Uzglabāšanas prasības, jo glabā lokāli
* Apvienošanas konflikti

**GitHub**

Apraksts - GitHub ir plaši izmantots tīmeklī izvietots koda izstrādātāju platforma, kas ļauj izstrādātājiem izveidot, uzglabāt, pārvaldīt un kopīgot savu kodu.

Plusi:

* Projektu un kodu glabāšana
* Atvērtība un koplietošana
* Versiju vadība
* Kopīga darba un koda pārskats
* Integrācijas un papildinājumi

Šeit es uzglabāju savu kodu tīmeklī un izvēlējos to dēļ pieredzes skolā un popularitātes, kura sniedz dziļāku dokumentāciju problēmu gadījumā.

Mīnusi:

* Privātumu un datu drošību jautājumi
* Maksas plāni un ierobežojumi
* Atkarība no platformas
* Konkurējošo platformu alternatīvas

**Draw.io**

Apraksts - Draw.io ir bezmaksas tiešsaistes diagrammu veidošanas rīks, kas ļauj izveidot blokshēmas, diagrammas, domu kartes, organizācijas diagrammas un daudz ko citu.

Plusi:

* Pilnīga funkcionalitāte bezmaksas
* Vienkārši izmantojams
* Dažādi diagrammu veidi
* Izvērsta sadarbības iespēja
* Integrācija ar citiem pakalpojumiem

Izvēlējos Draw.io nevis alternatīvas, jo man nebija īpaša nepieciešamība kompleksām diagrammām, tādēļ man tas bija visērtākais un vieglākais variants viedojot diagrammas, jo arī varēju tās pieglabāt Google Drive un vēlāk rediģēt pēc vajadzības.

Mīnusi:

* Nepieciešama interneta pieeja
* Ierobežots krāsu un dizaina varianti
* Ierobežota sadarbības funkcionalitāte
* Ierobežota mobilā atbalsta

Microsoft Word - Microsoft Word ir Microsoft izstrādāta plaši izmantota komerciāla tekstapstrādes programma, kuru es izmantoju šīs dokumentācijas rakstīšanas procesā.

## Iespējamo (alternatīvo) risinājuma līdzekļu un valodu apraksts

Kā jau iepriekš tika minēts, lai izstrādātu tīmekļa vietnes ir pieejami ļoti daudz resursi un rīki, uzskaitot alternatīvas manis minētajiem rīkiem būtu:

**Atom**

Apraksts - GitHub izveidotais teksta redaktors Atom ir ieguvis popularitāti, pateicoties tā plašajam pakešu krātuvei, kuru var viegli atjaunināt, un lietotājam draudzīgajam interfeisam. Tā ir lieliska iespēja programmētājiem, kuri meklē elastīgu vidi sava koda izveidei.

Plusi:

* Ļoti pielāgojams
* Tīrs un intuitīvs interfeiss
* Lieliska Git integrācija
* Spēcīga izstrādātāju kopiena

Mīnusi:

* Lēnāks nekā citi redaktori (it sevišķi strādājot ar lielākiem projektiem)
* Resursu ietilpīgs
* Ierobežots valodu atbalsts
* Mazāk stabils

Neizvēlējos šo koda redaktoru, jo negribēju mācīties apieties ar jaunu koda redaktoru un iespējams izmantot vairākus dēļ ierobežotā valodu atbalsta, kā arī tērēt vairāk resursu, bet strādāt lēnāk.

**Vue js**

Apraksts - Vue js ir JavaScript ietvars lietotāja interfeisu veidošanai. Tas ir balstīts uz standarta HTML, CSS un JavaScript, un nodrošina deklaratīvu un uz komponentiem balstītu programmēšanas modeli, kas palīdz efektīvi izstrādāt lietotāja saskarnes neatkarīgi no tā, vai tās ir vienkāršas vai sarežģītas.

Plusi:

* Viegli uztverams un iemācāms
* Lietotāja draudzīga sintakse
* Modularitāte
* Izstrādes ātrums
* Aktīva kopiena un resursi

Mīnusi:

* Mazāka ekosistēma salīdzinājumā ar citiem rīkiem
* Korporatīvo atbalstu trūkums
* Nestabilitāte
* Resursu pieprasījums

Vue js ir laba alternatīva React.js, taču es to neizvēlējos, jo man ir vairāk pieredze ar React.js un es nevēlējos pagarināt projekta izstrādes ilgumu mācoties jaunus principus.

**MySQL**

Apraksts - MySQL ir pasaulē populārākā atvērtā koda datubāze. Relāciju datu bāze sakārto datus vienā vai vairākās datu tabulās, kurās dati var būt saistīti viens ar otru.

Plusi:

* Brīvi pieejama un var tikt izmantota bez maksas
* Plaša pielietojamība
* Augsta veiktspēja
* Laba drošība

Mīnusi:

* Ierobežots atbalsts un dokumentācija, par spīti popularitātei
* Nepilnīgas transakcijas atbalsts
* Jaunāka SQL standarta neievērošana

Šo rīku es noteikti apsvēru tīmekļa vietnes izstrādes laikā, taču šai datubāzei ir nepieciešams serveris un lai man nebūtu vajadzība pēc tāda es šoreiz no tā atteicos, taču ja palaistu “Music Player” tīmekļa vietni plašāk pasaulē un rastos daudz dati un iespēja serverim noteikti izmantotu šo alternatīvu, jo tas ir ļoti atpazīstams un iespējams daudz ko atrast interneta resursos par tā lietojamību.

**Subversion (SVN)**

Apraksts - Subversion ir atvērtā pirmkoda versiju kontroles sistēma, kas centrālajā serverī uztur pirmkodu; ikviens, kas vēlas mainīt kodu, piekļūst šiem failiem no klientiem. Šis klienta servera modelis ir vecāks stils, salīdzinot ar izplatīto git modeli, kurā izmaiņas var saglabāt lokāli un pēc tam izplatīt centrālajā vēsturē (un citās filiālēs), kad tās tiek virzītas uz augšupējo krātuvi.

Plusi:

* Daudzfunkcionāla versiju vadība
* Centralizēts repozitorijs
* Stabilitāte un uzticamība
* Labāka atbalsta struktūra ar binārajām datnēm

Mīnusi:

* Atkārtota noklusējuma darbība
* Ierobežota atzaru un apvienošanas elastība
* Mazāka veiktspēja ar lieliem projektiem
* Nespēja veikt darbu bezsaistē

Ar šo versijas kontroli man bija neliela pieredze prakses vietā un tā var būt alternatīva, taču to var uzskatīt par mazāk elastīgu un mazāk modernu risinājumu salīdzinot ar Git. Git man likās modernāks un vieglāks, kā arī par to ir daudz vairāk aprakstīts tīmeklī, jo tas ir daudz plašāk izmantots.

**C#**

Apraksts - C# ir universāla augsta līmeņa programmēšanas valoda, kas atbalsta vairākas paradigmas. Ar to ir iespējams paveikt vairākas lietas vairākās platformās, kas iekļauj arī mobilo aplikāciju izstrādi.

Plusi:

* Platformas neatkarība, var tikt izpildīta uz dažādām operētājsistēmām
* Lielais kopienas atbalsts
* Objektorientēta valoda, kas nodrošina labu organizāciju

Mīnusi:

* Labākā integrācija un atbalsts parasti tiek nodrošināts tikai uz Microsoft
* Lai izprastu visus C# iespējas un labas prakses, var būt nepieciešams laiks
* Ierobežota lietojumprogrammu joma
* Veiktspējas ierobežojumi

Pieredze ar C# man arī bija prakses vietā, taču kā jau minēts iepriekšējā nodaļā, PHP vienkārši šķita ērtāks un vieglāks šāda apjoma tīmekļa vietnei.

**Deezer API**

Apraksts - Deezer API ir spēcīgs rīks izstrādātājiem, kuri vēlas savām lietojumprogrammām pievienot ar mūziku saistītas funkcijas. Tas nodrošina piekļuvi plašam datu klāstam, tostarp detalizētai informācijai par ierakstiem, izpildītājiem, albumiem un lietotāju profiliem.

Plusi:

* Piekļuve milzīgam mūzikas datu krājumam
* Plašas funkcijas un iespējas

Mīnusi:

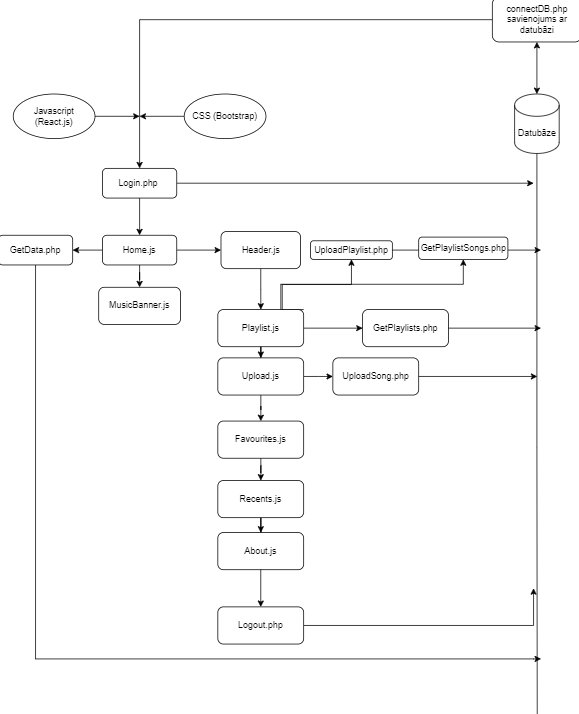
* Ierobežotas bezmaksas iespējas
* Konkurences jautājumi
* Pakalpojuma atkarība no Deezer platformas
* Mūzikas licences jautājumi
* Nepastāvīgas izmaiņas un atjauninājumi

Es šo gandrīz izvēlējos nevis Spotify API, taču Spotify bija plašāka un vieglāk saprotamāka dokumentācija.

# Sistēmas modelēšana un projektēšana

Šajā sadaļā tiks attēloti sistēmas modeļi un projektēšana “Music Player” tīmekļa vietnei. Sistēmas struktūrai ir vairākas daļas. Zemāk var iepazīties ar sistēmas struktūras modeli. (Skat. 1. attēlu **Sistēmas struktūras modelis**)

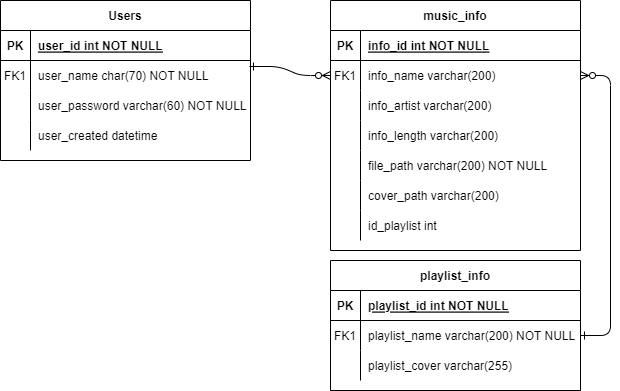
## Sistēmas struktūras modelis



**1. attēls. Sistēmas struktūras modelis**

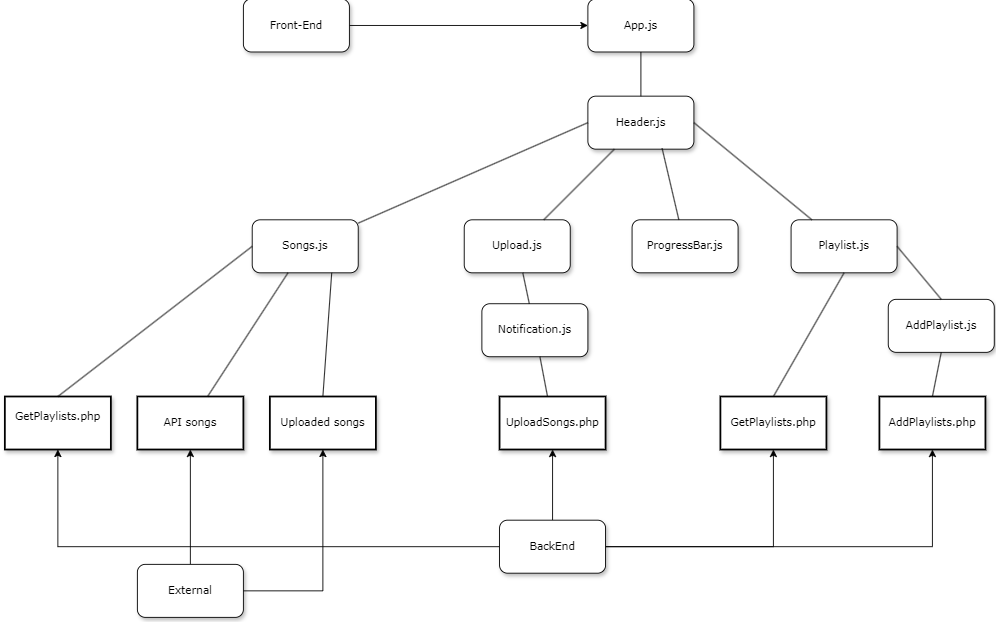
## ER diagramma

ER diagrammas ir datu bāzu modelēšanas metode, kura tiek izmantota, lai izveidotu datu bāzes shēmu vai datu modeli un izprastu veidojamās datu bāzes uzbūvi un pamatprincipus.

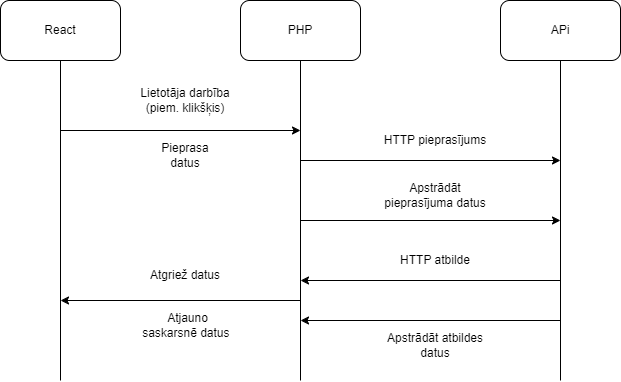


**2. attēls. ER Diagramma**

## Funkcionālais un dinamiskais sistēmas modelis



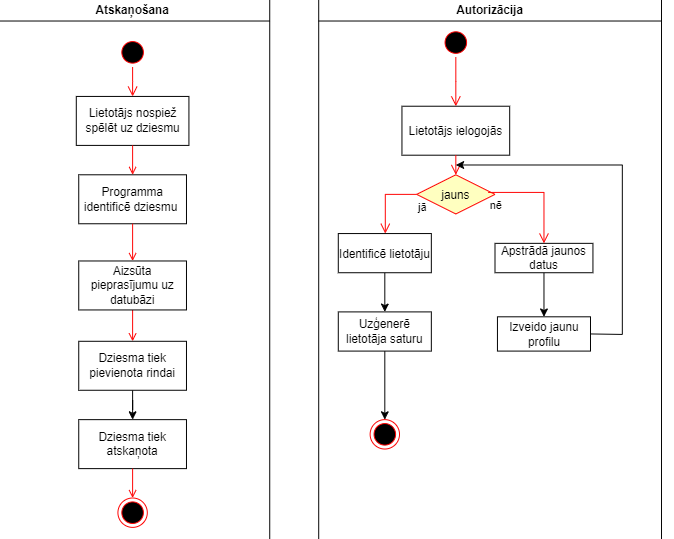
**3. attēls. Komponentu modelis**



**4. attēls. Datu plūsma**

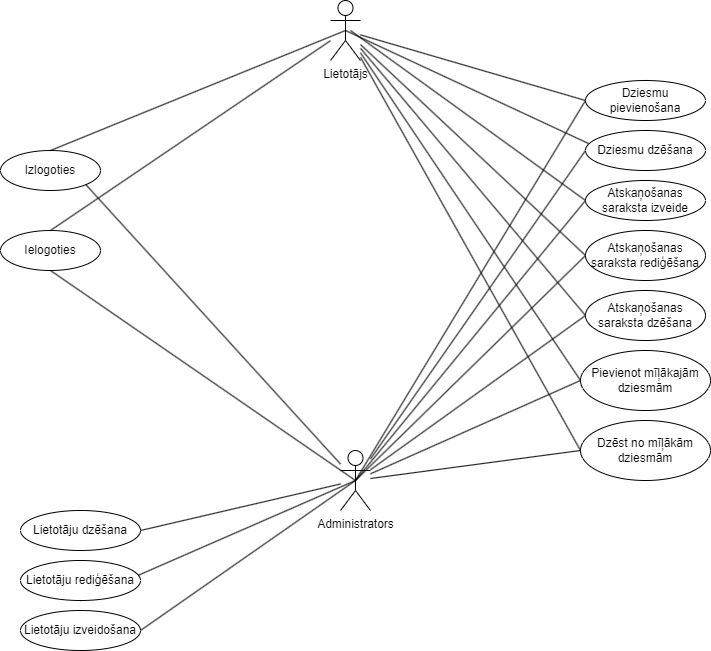
## Aktivitāšu diagramma

Šajā sadaļā ir aktivitāšu diagrammas vairākām aktivitātēm, kas notiek mājaslapas darbībā.



**5. attēls. Aktivitāšu diagrammas**

## Lietojumgadījumu diagramma



**6. attēls. Lietojumgadījumu diagramma**

## Sistēmas moduļu apraksts un algoritmu shēmas

Front-End daļa ir svarīga sistēmas komponente, jo tā nodrošina lietotāja saskarnes izstrādi un veidošanu, kas ir galvenais kontaktpunkts starp lietotāju un aplikāciju. Šajā daļā tiek izmantotas dažādas tehnoloģijas un rīki, piemēram, React.js un CSS Bootstrap, lai nodrošinātu modernu, elastīgu un vizuāli pievilcīgu lietotāja pieredzi. Šie rīki ļauj veidot dinamiskas un interaktīvas saskarnes, kas padara lietošanas procesu intuitīvu un viegli saprotamu.

Front-End ir sadalīts vairākās komponentēs, kas atbild par dažādām funkcionalitātēm un sadaļām aplikācijā. Lielās komponentes var saturēt galveno saskarni vai funkcijas, kā arī to ietvaros atrodamas mazākas komponentes, kas atbild par detalizētākām sadaļām vai funkcionalitātēm. Tātad, galvenā mūzikas atskaņotāja komponente var ietvert sadaļas, piemēram, atskaņotāja kontroli un informāciju par pašreizējo dziesmu, bet tās sastāvā var būt arī mazākas sadaļas, piemēram, skaņas kontroles un saraksta pārvaldības funkcijas.

Back-End daļa ir būtiska sistēmas daļa, kas nodrošina datu apstrādi, drošību un datu glabāšanu. Šajā daļā tiek izmantoti PHP un datubāze, lai veiktu dažādus uzdevumus, piemēram, datu ieguvi, ievadi, rediģēšanu un dzēšanu. Back-End faili satur loģiku un funkcijas, kas tiek izmantotas, lai sazinātos ar datubāzi un veiktu nepieciešamās darbības. Šāda sadalījuma un organizācijas sistēma nodrošina sistēmas stabilitāti, efektivitāti un uzticamību, nodrošinot, ka dati tiek pareizi apstrādāti un saglabāti.

PHP faili satur kodu, kas tiek izpildīts servera pusē, un tiek izmantots, lai veiktu dažādas darbības ar datubāzi, piemēram, ierakstu atlasīšanu, pievienošanu, rediģēšanu vai dzēšanu. Šie faili izmanto MySQLi vai PDO bibliotēkas, lai izveidotu savienojumu ar datubāzi un veiktu darbības, izmantojot SQL valodu.

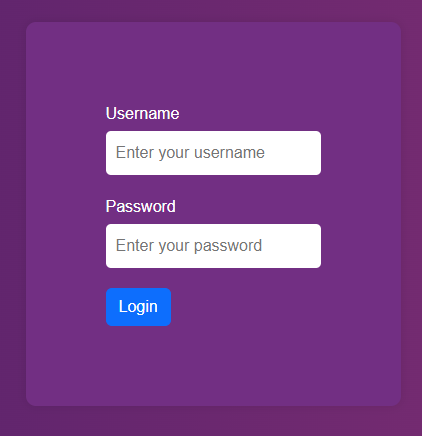
Savienojums ar datubāzi tiek uzstādīts, izmantojot savienojuma parametrus, kas ietver servera nosaukumu, lietotājvārdu, paroli un datubāzes nosaukumu. Kad savienojums ir veiksmīgi uzstādīts, PHP faili var izpildīt dažādas datu apstrādes darbības, izmantojot SQL vaicājumus, kā SELECT, INSERT, UPDATE un DELETE.

Kopumā, gan Front-End, gan Back-End daļas ir būtiskas sistēmas veiksmīgai darbībai, un to sadarbība ļauj nodrošināt lietotājiem labu un pilnīgu lietošanas pieredzi, kas atbilst mūsdienu prasībām un standartiem.

# Lietotāju ceļvedis

Zemāk tiek aprakstīti instrukcijas soļi “Music Player” tīmekļa vietnes veiksmīgai un ērtai izmantošanai.

## Autentifikācija



**7. attēls. Autorizācijas lapa**

**5.1.1. Ielogošanās**

Ielogošanās lapā lietotājam jāaizpilda lauks Username un Password, ja parole un lietotājvārds sakrīt, tad lietotājs tiek autorizēts sistēmā.

**5.1.2. Izlogošanās**

Lai lietotājs izlogotos ārā no sistēmas jānospiež poga “Logout”, kas atrodas navigācijas joslā, pēc veiksmīgas izlogošanās lietotājs tiek atgriezts uz ielogošanās lapu.

**5.1.3. Reģistrācija**

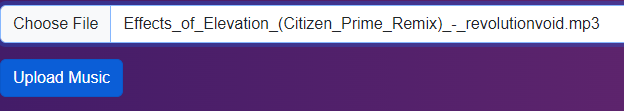
Ielogošanās lapā lietotājam jāuzspiež uz pogu “Register”, kur atveras forma ar 3 laukiem, kas ir Username, Password, Confirm password. Password un Confirm password laukiem ir jāsakrīt un Username ir jābūt unikālam, ja viss sakrīt, tad lietotājs var iet atpakaļ uz ielogošanos un ielogoties ar jauno lietotāju.

## Dziesmas augšuplāde



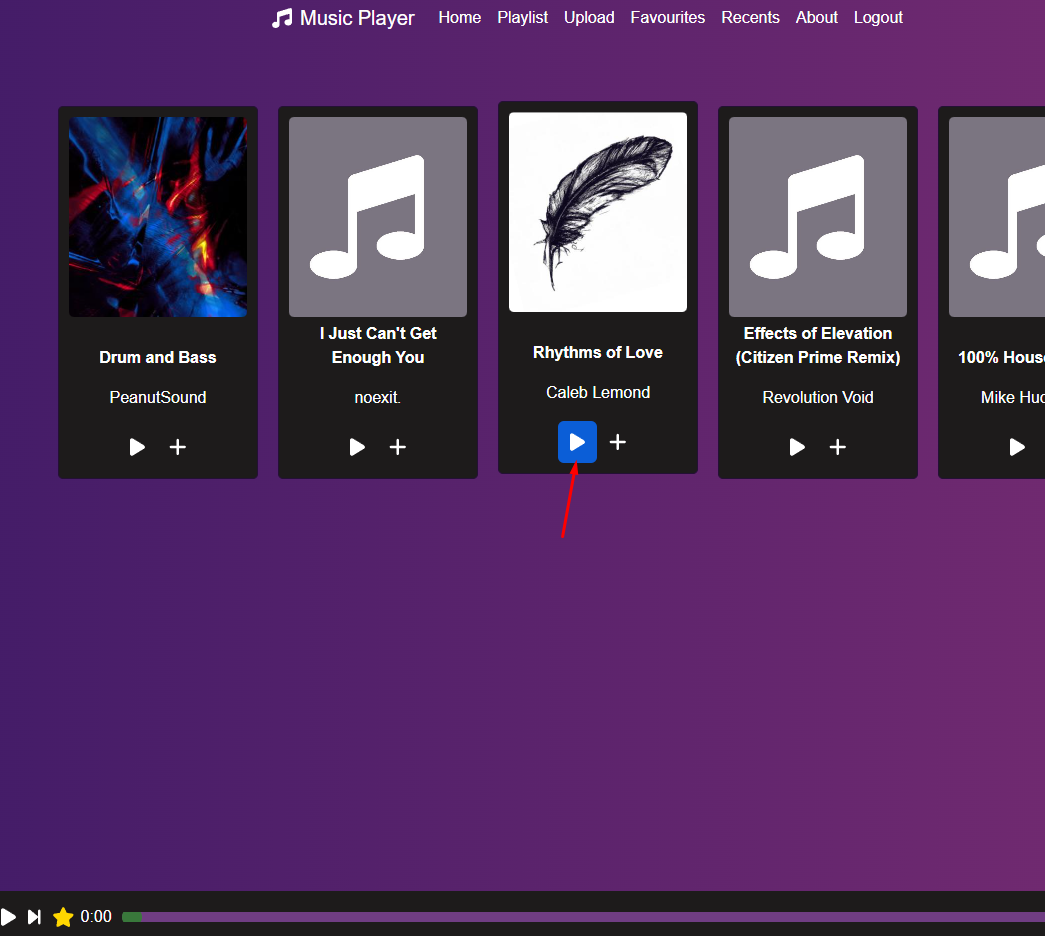
**8. attēls. Navigācijā augšuplāde**

Navigācijas joslā lietotājam jāuzspiež uz sadaļu “Upload”, kur lietotājam ir iespēja izvēlēties MP3 failu no lokālās krātuves, nospiežot pogu “Upload Music” lietotāja fails tiek augšuplādēts, ja viss ir izdevies, tad parādās paziņojums par veiksmīgu augšuplādi.



**9. attēls. Augšuplāde**

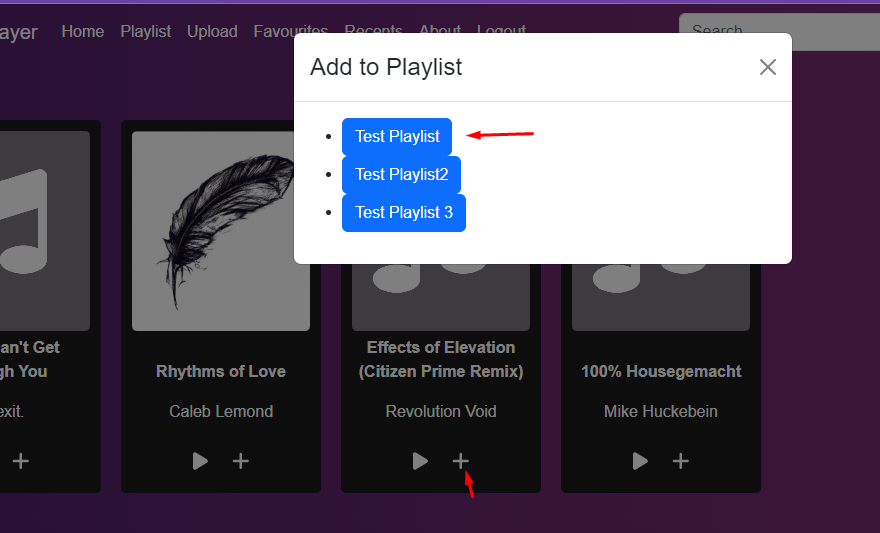
## Dziesmas spēlēšana



**10. attēls. Spēlēšanas poga**

Lai spēlētu dziesmu lietotājs gan Playlist sadaļa kāda atskaņošanas sarakstā, gan Home sadaļā, kur it pieejamas dziesmas un jāuzspiež uz “Play” pogu, dziesma sāks atskaņoties un tās progress būs redzams ekrāna apakšā progresa joslā.

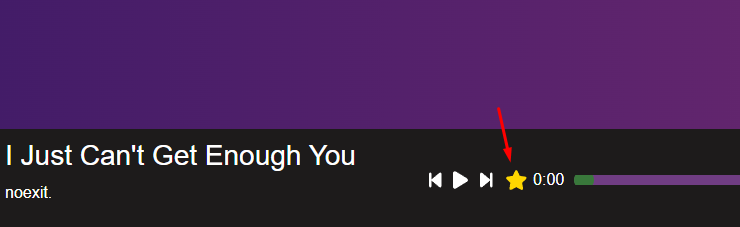
## Dziesmas pievienošana atskaņošanas sarakstam



**11. attēls. Pievienošana atskaņošanas sarakstam**

Lai lietotājs pievienotu dziesmu atskaņošanas sarakstam, uz dziesmas uzspiež uz plusa pogas, kura parādīs visus pieejamos atskaņošanas sarakstus, kad dziesma pievienota atskaņošanas sarakstam parādīsies paziņojums par veiksmīgu pievienošanos.

## Dziesmas pievienošana favorītiem (mīļākajām)



**12. attēls. Favorīta pievienošana**

Kamēr tiek spēlēta dziesma, progresa joslā ir redzama zvaigznīte, kuru uzspiežot lietotājam ir iespēja konkrēto dziesmu noņemt vai pievienot favorītiem (mīļākajām dziesmām). Tās iespējams apskatīt “Favourites” sadaļā navigācijas joslā.

## Dziesmas meklēšana



**13. attēls. Meklēšana**

**5.6.1. Meklēšana pēc nosaukuma**

Lietotājam ir iespēja meklēt dziesmu pēc nosaukuma, rakstot tās nosaukumu un nospiežot meklēšanas pogu parādīsies vairākas dziesmas, kuras satur lietotāja ievadīto nosaukumā.

**5.6.2. Meklēšana pēc izpildītāja**

Lietotājam ir iespēja meklēt dziesmu pēc izpildītāja, ievadot izpildītāju un nospiežot meklēšanas pogu parādīsies vairākas dziesmas, kuras satur lietotāja ievadīto izpildītājā.

**5.6.3. Meklēšana pēc izpildītāja**

Lietotājam ir iespēja meklēt dziesmas pēc žanra, ievadot žanru un nospiežot meklēšanas pogu parādīsies vairākas dziesmas, kuras satur lietotāja ievadīto žanru.

## Lietotāja profila apskatīšana

Lietotājam ir iespēja apskatīt savu profilu nospiežot uz profila ikonas ekrāna labajā augšējā stūrī, kur lietotājs var gan apskatīt, gan rediģēt savu informāciju nospiežot “Edit” pogu blakus attiecīgajai informācijai. Lietotājs profila apskatā pašā apakšā var arī pieprasīt dzēst vai deaktivizēt savu profilu.

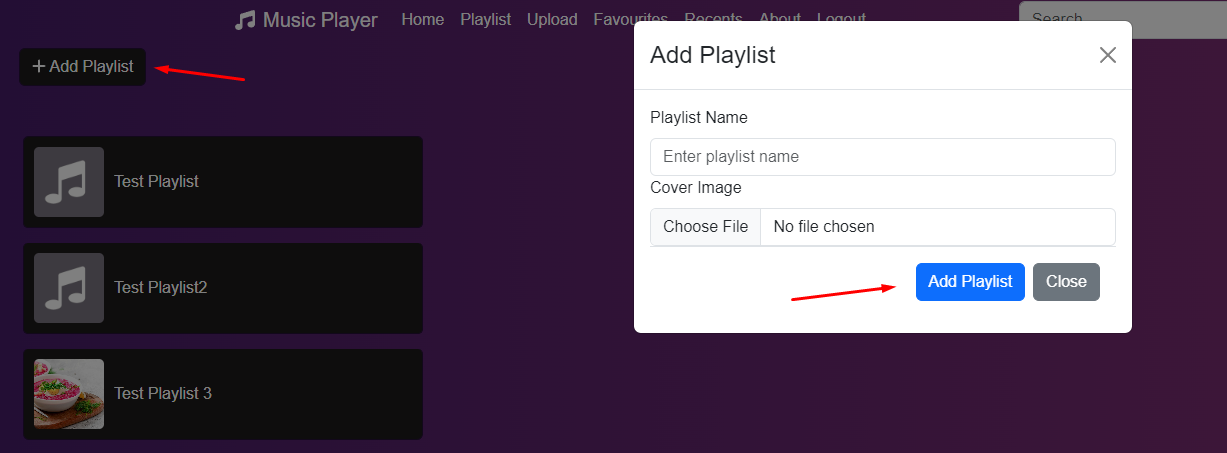
## Lietotāja dziesmu vēsture

Lietotājam navigācijas joslā nospiežot uz sadaļu “Recents” ir iespēja apskatīt šajā sesijā iepriekš klausītās dziesmas (klausīšanas vēsturi), kā arī tās vēlreiz atskaņot nospiežot uz “Play” pogu.

## Tīmekļa vietnes info

Lietotājam ir iespēja apskatīt informāciju par tīmekļa vietni “Music Player” nospiežot uz “About” sadaļu navigācijas joslā, kas ir informatīva sadaļa ar kontaktinformāciju pēc vajadzības.

## Atskaņošanas saraksta pievienošana

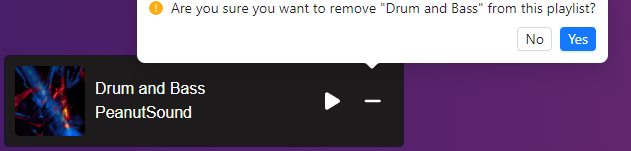


**14. attēls. Atskaņošanas saraksts**

Lai lietotājs pievienotu jaunu atskaņošanas sarakstu, no sākuma lietotājam ir jāuzspiež uz sadaļu “Playlist” navigācijas joslā un pēc tās lietotājs uzspiežot uz pogu “Add Playlist” kreisajā augšējā stūrī atver formu, kur nepieciešams aizpildīt informāciju par veidojamo atskaņošanas sarakstu. Ja visa informācija ir pareiza parādās paziņojums par veiksmīgi pievienotu jauno atskaņošanas sarakstu un to iespējams apskatīt “Playlist” sadaļā, kā arī izvēlēties kā vienu no atskaņošanas sarakstiem, kur pievienot dziesmas.

## Atskaņošanas saraksta apskate

Lai apskatītu atskaņošanas sarakstu un dziesmas, kas tajā ietilpst lietotājam jāuzspiež uz sadaļu “Playlist” navigācijas joslā un tad jāuzspiež uz vēlamo atskaņošanas sarakstu, kad tas ir izdarīts tiek parādītas visas dziesmas, kas atrodas atskaņošanas sarakstā. Dziesmas ir gan iespējams spēlēt, gan noņemt no atskaņošanas saraksta.



**15. attēls. Atskaņošanas saraksta dziesma**

# Testēšanas dokumentācija

Šajā nodaļā tiek aprakstīta un attēlota mājaslapas “Music player” testēšanas dokumentācija.

## Izvēlētās testēšanas metodes, rīku apraksts un pamatojums

Mūzikas atskaņošanas platformas izstrādes laikā bija būtiski nodrošināt augstu kvalitāti un funkcionalitāti visos tās aspektos. Izvēloties metodes, tika ņemtas vērā sistēmas specifikas, izstrādes cikla laiks un resursu pieejamība. Tika arī veikti zaru pārklājumu aprēķini, kā arī komandu pārklājuma aprēķini. Pielietotie rīki bija Microsoft Excel, kas ir izklājlapu lietotne, kā arī developer tools interneta pārlūkā, jo ņemot vērā, ka šis projekts nav pārāk liels apjomā un viss kods tika izstrādāts Visual Studio Code, nebija vajadzības iesaistīt plašākus vai sarežģītākus testēšanas rīkus.

Melnās kastes testēšana (Black Box Testing): Tika izvēlēta tāpēc, ka tā ļauj efektīvi novērtēt sistēmas ārējo uzvedību neatkarīgi no tās iekšējās implementācijas. Šī pieeja ir ideāli piemērota funkcionalitātes un lietotāja saskarnes pārbaudīšanai, jo ļauj koncentrēties uz galvenajām lietotāju vajadzībām un prasībām.

Balta kastes testēšana (White Box Testing): Tika izmantota, lai notestētu sistēmas iekšējo loģiku un struktūru. Šī metode ir svarīga, jo nodrošina detalizētu koda pārbaudi, palīdzot atklāt slēptas kļūdas un uzlabot koda optimizāciju.

## Testpiemēru kopas

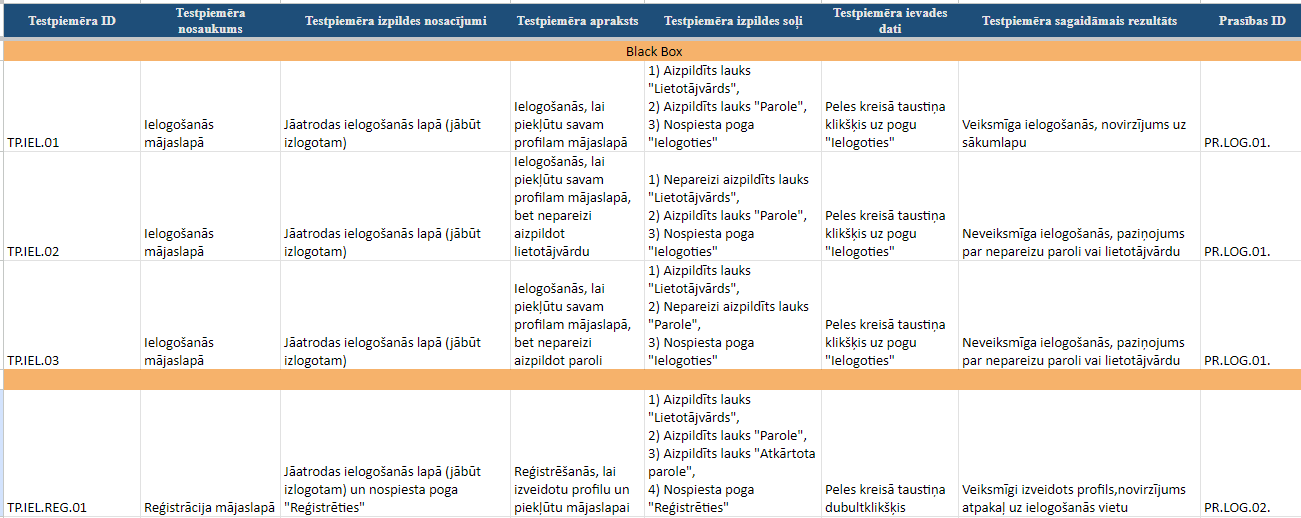
Zemāk redzamas lielākā daļa no testpiemēru kopām. (Skat. 16. attēlu). Kopumā ir apmēram septiņas testpiemēru kopas, kuras testpiemēros sadalās arī dziļāk atkarībā no vajadzīgās specifikas. Piemēram, Ielogošanas lapā tiek iekļauti gadījumi gan par ielogošanos, gan par reģistrēšanos, kur darbības un testi atšķiras.



**16. attēls. Testpiemēru kopas**

## Testēšanas žurnāls

Zemāk redzamajos attēlos var skatīt daļu no pieejamajiem testpiemēriem, kā arī daži no testiem, kas tika veikti testēšanas žurnālā šim projektam.



**17. attēls. Testpiemēri**



**18. attēls. Testēšanas žurnāls**

# Secinājumi

Veidojot projektu “Music player” es sastapos ar vairākām problēmām un izaicinājumiem. Sākot projektu biju izdomājusi izmantot tīru PHP valodu, taču sapratu, ka izmantojot React.js bibliotēku labāks risinājums būtu izmantot Laravel ietvaru PHP, jo tam ir daudz labāka sadarbība ar React.js dēļ tā plašā izmantojuma ar Vue.js, kā arī vairākas citas tehnoloģijas.

1. Izaicinājums Laravel apgūšanā - Laravel pirms šī projekta nebiju izmantojusi, tādēļ bija izaicinājums to apgūt.

2. CSS Bootstrap – Es biju redzējusi to iepriekš, bet aktīvi vēl nebiju to izmantojusi, tādēļ tas aizņēma kādu daļa no laika, lai varētu teikt, ka esmu apguvusi vismaz tā pamatus

3. API – ar API bija vairāki izaicinājumi izvērtējot no sākuma, kuru no tiem es vēlētos pielietot atkarībā no tā piedāvātajām iespējām un tad arī bija svarīgi izprast, kā pareizi izmantot API piedāvātās funkcionalitātes, specificēt to manam projektam un pielāgot to manām izmantotajām tehnoloģijām.

4. React.js – Ar React.js bibliotēku man bija vismazāk problēmas, jo to apguvu ne tikai pašmācības ceļā, bet arī prakses vietā ar to nodarbojos gandrīz katru dienu un minimālus pamatus vēlāk sniedza arī skolā dažās lekcijās, tādēļ man jau bija diezgan plaša izpratne par šīs bibliotēkas pielietošanu.

5. Sasniegumi Laravel – Es nevarētu teikt, ka pilnībā pārvaldu Laravel, taču tas man bija diezgan liels sasniegums to pielietot šajā projektā.

6. Audio faili – Iepriekšēja pieredze ar audio failu (MP3) apstrādi un atskaņošanu man nebija bijusi līdz šī projekta veidošanai, tādēļ tas bija ievērojams sasniegums to visu izdarīt un pielietot šajā projektā ar minimālām grūtībām.

7. Nākotne – Veidojot šo projektu man radās diezgan laba izpratne par audio failu apstrādi un glabāšanu un nākotnē noteikti būtu interese papildināt šo projektu piemērojot iespējams papildus API, iespējas, plašāku administrēšanu, ja to palaistu publikā. Vērts ir arī pieminēt, ka man pašai ikdienas dzīvē šī mājaslapa var būt diezgan noderīga, jo varu piekļūt savai mūzikai no jebkuras vietas, ja to tur esmu augšupielādējusi.

# Lietoto terminu saīsinājumu skaidrojumi

**Redux** – React.js datu plūsmas pārvaldīs

**WEB** - Globālais tīmeklis (World Wide Web)

**Front-end** - klienta puse

**Back-end** - servera puse

**API** - Lietojumprogrammu interfeiss (Application Programming Interface)

**Black box** – melnās kastes testēšanas metode (neiesaista kodu)

**White box** – baltās kastes testēšanas metode, kas iesaista kodu

**Framework** – Ietvars (Piemēram Laravel PHP ietvars)

**CSS** - Cascading Style Sheets (Stila lapu kaskādēšana)

**Developer tools** - Izstrādātāju rīki (piemēram interneta pārlūkā)

**Max.** – Maksimums

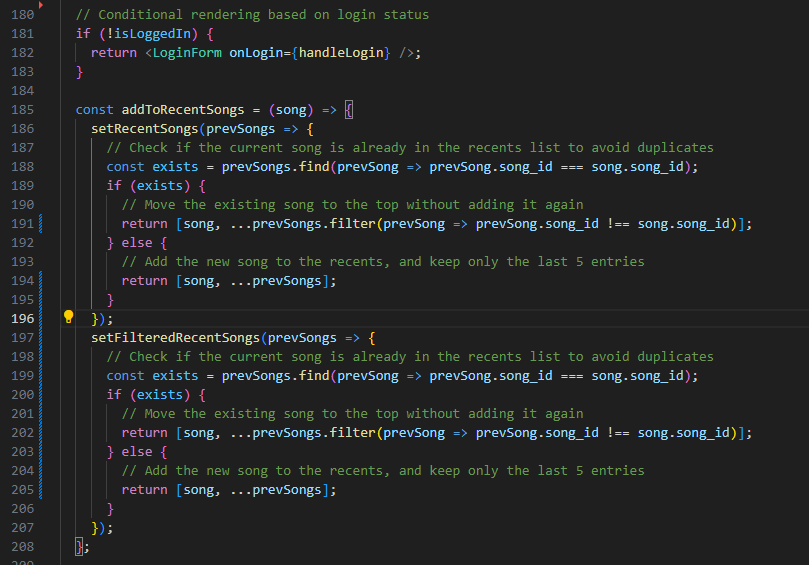
# Literatūras un informācijas avotu saraksts

1. <https://react.dev/>
2. <https://testsigma.com/blog/test-suite-vs-test-case/>
3. <https://www.w3schools.com/>react/
4. <https://react-bootstrap.netlify.app/>
5. <https://laravel.com/>
6. <https://ant.design/components/overview/>
7. <https://www.php.net/>
8. <https://www.w3schools.com/php/>
9. <https://www.lucidchart.com/blog/data-flow-diagram-tutorial>
10. <https://www.visual-paradigm.com/tutorials/how-to-create-data-flow-diagram/>

**Pielikumi**

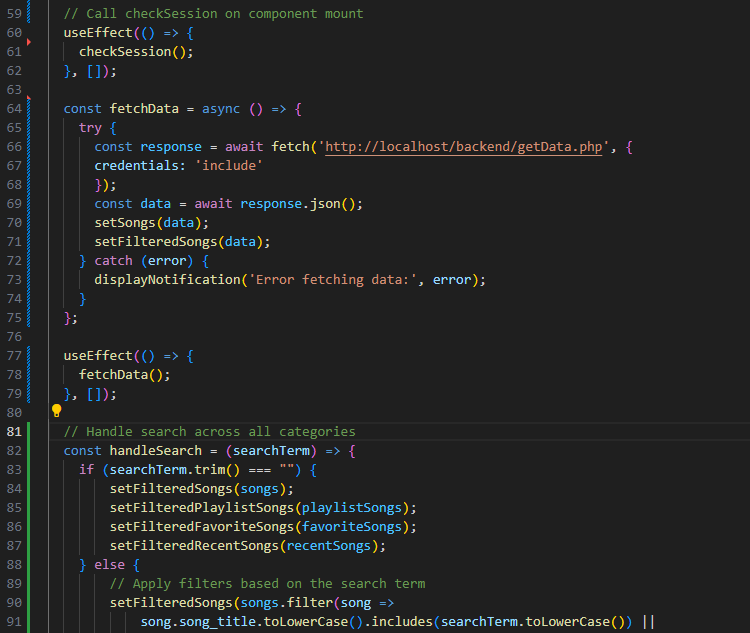
**1. pielikums**

**Programmas kods React.js**



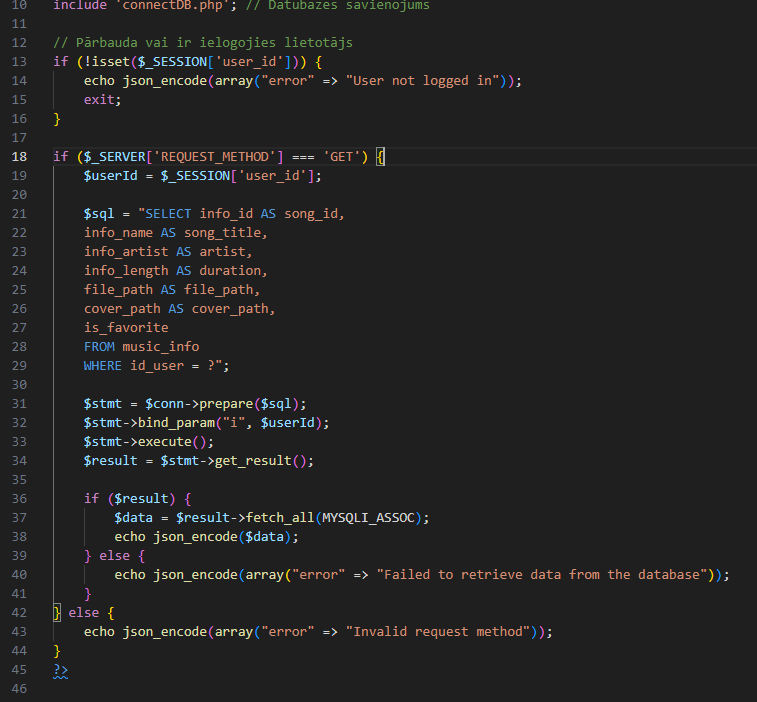
**2. pielikums**

**Programmas kods React.js**



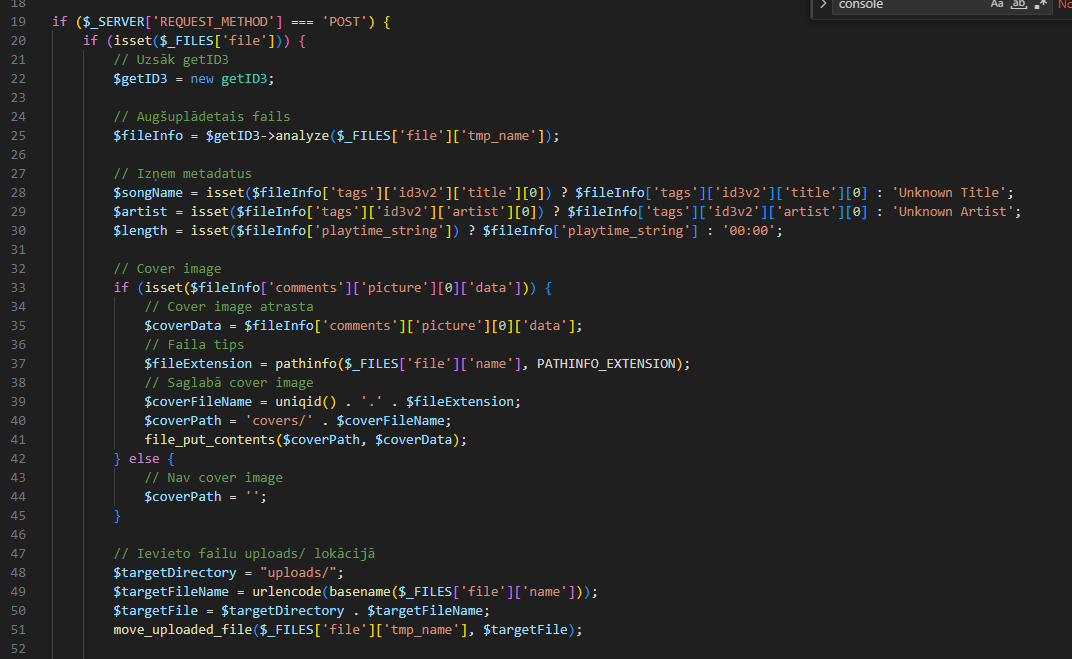
**3. pielikums**

**Programmas kods PHP**



**4. pielikums**

**Programmas kods PHP**



**5. pielikums**

**Programmas kods PHP**

