JELGAVAS TEHNIKUMS

AGNIJA BENDZIKA

Prototipa izveide mūzikas atskaņošanas aplikācijai *Windows* un *Arch Linux* darbvirsmu vidēm

Kvalifikācijas darbs
kvalifikācijas ieguvei
programmēšanas tehniķis

Darba izpildītājs: ..... 410.gr.izgl. A. Bendzika

Jelgava 2026

ANOTĀCIJA

Bendzika A. Prototipa izveide mūzikas atskaņošanas aplikācijai *Windows* un *Arch Linux* darbvirsmu vidēm: kvalifikācijas darbs. Jelgava: JT, 2026. 179 lpp., 39 attēli, 20 tabulas, 19 pielikumi.

Kvalifikācijas darbā tiek aprakstīta darbvirsmas mūzikas atskaņotāja prototipa izstrāde, kas atbalsta darbību *Windows* un *Linux* operētājsistēmās. Darba mērķis ir izveidot funkcionālu prototipu atbilstoši klienta prasībām, ar uzsvaru uz pielāgošanas iespējām (karstie taustiņi, fontu un krāsu tēmu maiņa) un saskarnes draudzīgumu.

Darba gaitā tika veikta esošo risinājumu analīze, izstrādāts vispārējs lietotāja saskarnes makets un izvērtēti labākie rīki izstrādei, secinot, ka prototipam un klientam vislabāk piemērots pirmkoda brīva palaišana *GitLab* vidē. Pati programmatūra ir realizēta *Java* platformā ar *JavaFX* grafisko saskarni, izmantojot *SQLite* datubāzi datu glabāšanai un pārvaldībai.

Izstrādes laikā tika veiksmīgi ieviestas svarīgākās klienta prasības, piemēram, audio failu atskaņošana, metadatu nolasīšana, kā arī saskarnes pielāgojamības mehānismi. Sistēmas arhitektūra ir balstīta uz trīs slāņu modeli, sekojot veselīga koda uzturēšanas principiem, nodrošinot atbildību nošķiršanu un koda uzturēšanas efektivitāti.

Prototips tika testēts *Windows 10 un 11* un *Arch Linux* sistēmās vairākos izstrādes posmos, kā arī tika veikta lietojamības novērtēšana kopā ar klientu. Klients apstiprināja gala rezultātu.

Rezultātā iegūts darbspējīgs prototips ar divām versijām priekš *Windows* un *Linux* sistēmām. Tas atbilst visām noteiktajām obligātajām prasībām un ir gatavs tālākai attīstībai un pilnveidošanai.

ANNOTATION

Bendzika A. Prototype development for a music listening application for Windows and Arch Linux systems: qualification thesis. Jelgava: JT, 2026. 179 pages, 39 images., 13 tables, 20 attachments.

This thesis describes the development of a desktop music player prototype that is supported on Windows and Linux operating systems. The aim of the work is to create a functional prototype according to the client requirements, with an emphasis on customization options (hotkeys, changing font and color themes) and interface friendliness.

During the planning phase, an analysis of existing products was carried out, a general user interface mockup was developed, and the best development tools were evaluated, concluding that releasing the source code as open-source on GitLab is most suitable solution for the prototype and the client. The software itself is implemented in Java with a JavaFX graphical interface, using an SQLite database for data storage and management.

During development, the most important client requirements were successfully implemented, such as audio file playback, metadata reading, as well as interface customization mechanisms. The system architecture is based on a three-layer model, following healthy code maintenance principles, ensuring separation of responsibilities and code maintainability efficiency.

The prototype was tested on Windows 10 and 11 and Arch Linux systems at several development stages, and usability evaluation was carried out together with the client. The client approved the final result.

The result is a working prototype with two versions for Windows and Linux systems. It meets all specified mandatory requirements and is ready for further development and improvement.

SATURS

ATTĒLU SARAKSTS

TABULU SARAKSTS

IEVADS

Mūzika ir svarīga ikdienas daļa, to lieto ne tikai atpūtai, bet arī darbā, lai uzturētu ritmu un uzmanību veicot uzdevumu. Tāpēc ir svarīgi, ka mūzikas atskaņotāji strādā ātri un nepamanāmi, lai mūzika nekļūtu par traucēkli. Diemžēl, daudzi mūzikas atskaņotāji darbvirsmas vidē ir nepilnīgi, vai arī ar ierobežotu pielāgojamību un personalizāciju. Projekta temats tika izvēlēts, lai apmierinātu klienta prasības pēc darbvirsmas mūzikas atskaņotāja ar paplašinātām pielāgošanas iespējām un pilnīgu funkcionalitāti.

Programma veiks visas pamatfunkcijas, ko veic mūzikas atskaņotāji: atskaņo mūziku, izveido sarakstus, pievieno vāka attēlu, u.tml. Viena no lielajām prioritātēm prototipa izveidē būs pielāgojamības pilnveidošana, lai veicinātu labvēlīgu lietotāja pieredzi: karsto atslēgu ieviešanu, kas dod iespēju lietot aplikāciju bez peles, teksta fonta un lieluma maiņu, kā arī pilnas pamācības ieviešana programmā un spēja darboties gan *Windows*, gan *Arch Linux* vidēs.

Prototipā tiks realizēta pamat funkcionalitāte audio failu atskaņošanai un organizēšanai un pilnveidota saskarnes pielāgojamība un vizuāla pievilcība.

Darba mērķis: Veidot mūzikas atskaņošanas aplikācijas prototipu pēc klienta prasībām.

Uzdevumi:

1. Prasību un esošu risinājumu izpēte;
2. Realizēt audio failu atskaņošanu, filtrāciju un meta datu rediģēšanu;
3. Darbības maketa izveide;
4. Motīvu un vizuālu efektu ieviešana;
5. Ieviest programmas pielāgojamības iespējas;
6. Programmas darbību testēšana *Windows* un *Arch Linux* vidēs;
7. Prototipa vides testēšana pie klienta.

AKRONĪMI UN SAĪSINĀJUMI

MP3	*MPEG-1 Audio Layer 3* - plaši izplatīts audio saspiešanas formāts.
M4A	*MPEG-4 Audio* - audio failu formāts, balstīts uz MPEG-4 konteineru.
AAC	*Advanced Audio Coding* - audio kodēšanas standarts ar augstu kvalitāti un mazāku faila izmēru nekā MP3.
WAV/WAVE	*Waveform Audio File Format* - nesaspiests audio failu formāts ar augstu skaņas kvalitāti.
FLAC	*Free Lossless Audio Codec* - bezpostošas saspiešanas audio kodeks.
OGG	*Ogg Vorbis* — atvērtā pirmkoda audio saspiešanas formāts.
UUID	Akronīms "*Universally unique identifier*".
UI	*User Interface* - lietotāja saskarne, vizuālā vide, ar ko lietotājs mijiedarbojas.
UX	*User Experience* - lietotāja pieredze, vispārējais iespaids no produkta lietošanas.
SQL	*Structured Query Language* - standarta vaicājumu valoda relāciju datubāzu pārvaldībai.
DB	*Database* - datubāze.
ER	*Entity-Relationship* - entītiju-attiecību modelis datubāzes struktūras attēlošanai.
API	*Application Programming Interface* - lietojumprogrammu saskarnes protokols, kas ļauj komponentēm savstarpēji komunicēt.
CSS	*Cascading Style Sheets* - stilu valoda grafiskā izskata definēšanai.
FXML	*FX Markup Language* - *JavaFX* lietotāja saskarnes apraksta valoda, balstīta uz XML.
XML	*Extensible Markup Language* - paplašināma iezīmēšanas valoda datu strukturēšanai.
JVM	*Java Virtual Machine* - *Java* virtuālā mašīna, izpildlaika vide *Java* programmu darbināšanai.
IDE	*Integrated Development Environment* - integrēta izstrādes vide (piem., *IntelliJ IDEA*).
DAO	*Data Access Object* - projektēšanas modelis datu piekļuves loģikas nošķiršanai.
MVC	*Model-View-Controller* - arhitektūras modelis, kas sadala sistēmu datos, skatā un vadības loģikā.
OS	*Operating System* — operētājsistēma.
GUI	*Graphical User Interface* - grafiskā lietotāja saskarne.
CI/CD	*Continuous Integration / Continuous Deployment* - nepārtrauktas integrācijas un izstrādes prakse programmatūras izstrādē.

TERMINI

Prototips	Pirmatnēja produkta parauga versija vai simulācija, ko izmanto ideju pārbaudei un testēšanai pirms galīgā produkta izstrādes. [9]
Metadati	Strukturēta informācija par audio failu, piemēram, dziesmas nosaukums, izpildītājs, albums un žanrs, kas tiek glabāta faila iekšienē.
Datubāze	Organizēta datu kopa, kas ļauj efektīvi glabāt, pārvaldīt un atlasīt informāciju; šajā projektā izmantota *SQLite* datubāze.
SQLite	Viegla relāciju datubāzes pārvaldības sistēma, kas glabā datus vienā failā un ir piemērota lokālām lietojumprogrammām.
Java	Plaši izmantota, platformneatkarīga programmēšanas valoda, uz kuras bāzes izstrādāts šis prototips.
Maven	Java projektu celtniecības un atkarību pārvaldības rīks, kas automatizē bibliotēku lejupielādi un projekta kompilāciju.
Repozitorijs	Krātuve, kurā tiek glabāts projekta pirmkods un tā versiju vēsture, parasti pārvaldīta ar *Git*.
Git	Izplatīta versiju kontroles sistēma, kas ļauj izsekot pirmkoda izmaiņām un sadarboties izstrādē.
GitLab	Tīmekļa platforma *Git* repozitoriju mitināšanai, projektu pārvaldībai un pirmkoda publiskai vai privātai publicēšanai.
Windows	*Microsoft Windows* ir grafiska operētājsistēma, ko izstrādājusi un izdevusi *Microsoft*. [2]
Arch Linux	*Arch Linux* ir neatkarīgi izstrādāta, x86-64 universāla GNU/*Linux* distribūcija, kas ir pietiekami daudzpusīga, lai atbilstu jebkuru lietošanas mērķi. [1]
Karstie taustiņi	Karstie taustiņi (*hotkeys)* ir viena taustiņa vai taustiņu kombinācija, ko nospiežat uz tastatūras, lai veiktu konkrētu darbību vai komandu programmā. [5]
JavaFX	*JavaFX* ir grafisko un multivides pakotņu kopums, kas ļauj izstrādātājiem projektēt, izveidot, testēt, atkļūdot un ieviest bagātīgas klienta lietojumprogrammas, kas darbojas vienādi dažādās platformās. [7]
Google Play	*Google Play* veikals ir digitāls veikals, kurā pieejami dažāda veida mediji. Lietotāji visbiežāk izmanto šo lietotni, lai lejupielādētu lietotnes un spēles. [4]
App Gallery	*HUAWEI AppGallery* ir *HUAWEI* oficiālais lietotņu veikals, kas kalpo kā programmatūras izplatīšanas platforma mobilajām ierīcēm. [8]
Podkāsts	Podkasts ir digitālo audio failu kopums vai sērija, kas ir pieejama lejupielādei vai klausīšanai internetā. Katrs atsevišķs audio ieraksts tiek saukts par podkasta sēriju. [6]
Atskaņošanas rinda (*Queue*)	Secīgs saraksts ar dziesmām, kas gaida atskaņošanu pēc kārtības.
Trīs slāņu arhitektūra	Programmatūras projektēšanas modelis, kas sistēmu sadala prezentācijas, darbības un datu slāņos, nodrošinot atbildību nošķiršanu.
Pirmkods	Cilvēkam lasāms programmas teksts, uzrakstīts programmēšanas valodā, kas tiek kompilēts izpildāmā programmā.
Motīvs (*Theme*)	Saskarnes vizuālā noformējuma kopums, kas nosaka krāsas, fontus un elementu izskatu.

1. UZDEVUMA NOSTĀDNE

1.1. Ievads

Kvalifikācijas darbs ir veltīts audio atskaņošanas programmas prototipa izstrādei, kas tiek veikts pēc klienta pasūtījuma. Paša projekta mērķis ir izstrādāt funkcionālu mūzikas atskaņošanas programmas prototipu balstoties pēc klienta prasībām, kas ieskaita divu versiju izstrādi priekš *Windows* un *Linux* vidēm. Projekta pirmkods ir brīvi pieejams, pēc klienta norādēm.

Konkrētie mērķi:

- pārrunāt nepieciešamās prasības ar klientu;
- veikt līdzīgu produktu analīzi un veikt atkārtotas pārrunas ar klientu;
- izplānot sistēmas vispārējo struktūru, maketus un nepieciešamos izstrādes rīkus;
- izveidot mūzikas atskaņošanas programmu ar atbalstu vairākiem audio failu formātiem un to atskaņošanai;
- realizēt efektīvu mūzikas bibliotēkas pārvaldību ar iespējām meklēt, filtrēt un organizēt audio failus;
- izstrādāt pielāgojamu lietotāja saskarni ar iespējām mainīt vizuālo fontu izmērus un krāsu motīvus;
- integrēt palīgfunkcijas:
    - karstos taustiņus;
    - vietēju lietošanas pamācību;
    - atsaucīgus un draudīgus paziņojumus.
- nodrošināt vairākplatformu saderību;
- veikt secinājums.

1.2. Vispārējas klienta prasības

Klients prototipa izveidei ir noteicis sekojošās prasības:

Obligātās prasības:

- atskaņošanas atbalsts audio formātiem: MP3, M4A, AAC, WAV, FLAC;
- atskaņošanas rindu pārvaldība (organizēšana, dzēšana);
- mūzikas failu meklēšana pēc mūzikas metadatiem, albuma, autoriem;
- saskarnes teksta pielāgošana (teksta fonta, izmēra un krāsas pielāgošana);
- saskarnes krāsu motīvu pielāgošana;

- saskarne lieto angļu valodu;
- saderība ar *Windows* 10 un *Arch Linux* vidēm;
- karsto taustiņi mūzikas atskaņotāja kontrolei.

Vēlamās prasības:

- automātiska audio avotu atjaunošana un statusa pārbaude;
- iekļauta lietotāja pamācība programmā;
- iespēja paplašināt audio formātu atbalstu failu atlasei un atskaņošanai;
- paziņot par nepieejamiem vai pazudušiem failiem;
- iespēja slēpt un dzēst atsevišķus audio failus no bibliotēkas;
- audio failu metadatu pievienošana un rediģēšana;
- paplašināta saderība *Windows 11* un citās *Linux* vidēs.

1.3. Darba uzdevumi

Prototipa izstrādei ir definēti sekojošie uzdevumi, sekojot klienta prasībām:

1) **Plānošanas fāze:**
   - **a.** veikt detalizētu prasību analīzi;
   - **b.** izpētīt esošos mūzikas atskaņotājus un noskaidrot to priekšrocības/trūkumus;
   - **c.** izstrādāt sistēmas arhitektūras plānu;
   - **d.** izveidot lietotāja saskarnes maketu;
   - **e.** izstrādāt funkcionālās specifikācijas.
   - **f.** Izplānot un implementēt datubāzes struktūru;
2) **Izstrādes fāze:**
   - **a.** implementēt audio atskaņošanas funkciju ar formātu atbalstu;
   - **b.** izveidot datu pārvaldības sistēmu audio failu metadatu apstrādei;
   - **c.** realizēt mūzikas bibliotēkas apsekošanu (meklēšana);
   - **d.** izstrādāt lietotāja saskarni ar *JavaFX* un pievienot pielāgošanas iespējas;
   - **e.** ieviest karsto taustiņu atbalstu;
   - **f.** izveidot konfigurācijas sistēmu iestatījumu saglabāšanai.
3) **Testēšanas un optimizācijas fāze:**
   - **a.** veikt pieņemšanas testēšanu ar klientu;
   - **b.** testēt aplikācijas darbību uz *Windows* un *Arch Linux* platformām;
   - **c.** optimizēt resursu lietojumu;
   - **d.** sagatavot lietotāja dokumentāciju.

4) **Piegādes fāze:**
    a. sagatavot programmas instalācijas pakotni, veidojot vienkāršu instalācijas procesu un skaidrus norādījumus, kā to izdarīt;
    b. sagatavot galīgo prototipa versiju klientam;
    c. dokumentēt atziņas.

Gala rezultāts:

- darbspējīgs mūzikas atskaņotāja prototips ar visām obligātajām funkcijām;
- vairākplatformu atbalstu ar stabilu darbību;
- intuitīva un pielāgojama lietotāja saskarne;
- pilnīga tehniskā dokumentācija un lietotāja pamācība.

1.4. Līdzīgu produktu analīze

1.4.1. Ievads

Projekta plānošanas procesā tiek apskatīti līdzīgi mūzikas atskaņošanas programmu un platformu risinājumi. Apskatot citu produktu priekšrocības un trūkumus ir iespējams ietaupīt laiku plānošanas fāzē, redzot reālus rezultātus mūzikas atskaņošanas produktiem un gūt ieskatu lietotāju atsauksmēs un domās par attiecīgām funkcijām. Tirgus izpēte veicina modernas saskarnes un noderīgu funkciju izstrādi.

1.4.2. *HUAWEI Music*

1.4.2.1. Apraksts

HUAWEI Music aplikācijas mērķis ir gādāt augstas kvalitātes un visaptverošu vietējās mūzikas klausīšanas pieredzi.

Produkts ir paredzēts *HUAWEI* mobilo telefonu īpašniekiem un plašām vecuma grupām, izceļot aplikācijas saskarnes draudzīgumu. Šī aplikācija nav pieejama *Google Play* ekosistēmā, tā ir ekskluzīvi atrodama tikai *App Gallery* veikalā.

Aplikācijas galvenās funkcijas ir:

1) vietējās mūzikas atskaņošana un apstrāde – ļaujot rediģēt metadatu informāciju;
2) sarakstu izveide;
3) citas funkcijas:
    a. audio apgriešana;
    b. miega taimeris;
    c. favorīti;

d. dažādas atskaņotāja vizuālie motīvi;

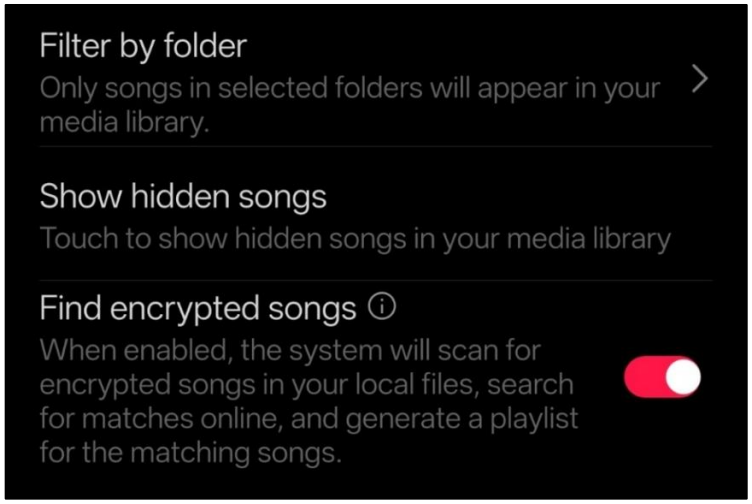
e. audio straumēšana;

f. failu paslēpšana/dzēšana.

Aplikācija ir ļoti labi vizuāli veidota ar intuitīvu un elegantu saskarni, dodot iespēju pielāgot kā tiek rādītas animācijas un uzveidot personisku motīvu. Pieejamas arī dažādās atskaņošanas opcijas, kas paredzētas dažādām klausīšanās vidēm – austiņām, braukšanai mašīnā, 3D efektiem u.tml. Aplikācija ir tikai pieejama mobilajām ierīcēm. (skatīt 1. attēls)

1. attēls ***HUAWEI Music*** **aplikācijas izskats mobilajos telefonos [10]**

Aplikācijai nav pieejamas lietotāja pamācība tiešsaistes vai bezsaistes formā. Paskaidrojumi ir tikai doti sarežģītākām funkcijām. (skatīt 2. attēls)



2. attēls ***HUAWEI Music*** **funkciju apraksti [Autora veidota ekrānkopija]**

1.4.2.2. Salīdzinājums

Apkopojot lietotāju atsauksmes *App Gallery* veikalā - lietotne ir viegli lietojama un saprotama, bet atjauninājumi un izstrādātāju atbalsts liek vilties. Lietotāji pozitīvi atsaucas uz programmas pielāgojamību, izkārtojuma saprotamību un klausīšanās pieredzi, bet atjauninājumu nestabilitāte, kas izraisījusi darbspējas pasliktināšanos, veicina neapmierinātību lietotājos.

Prototips lietotāju pieredzi uzlabo ar plašākas pielāgojamības iespējas nekā konkurents. Jāmin, ka klients specifiski nosauca *HUAWEI Music* kā galveno iedvesmu prototipa vēlamajam saskarnes rezultātam.

1.4.3. *Spotify*

1.4.3.1. Apraksts

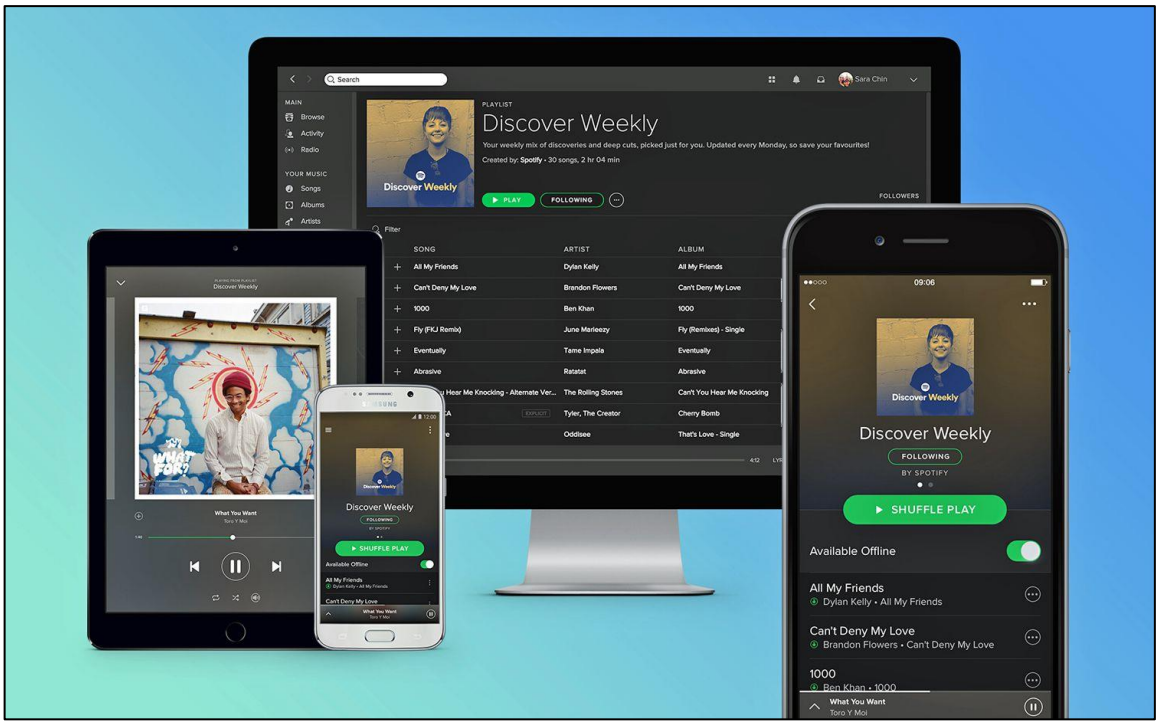
Spotify ir digitālas mūzikas, podkāstu un video straumēšanas serviss, kas nodrošina piekļuvi miljoniem dziesmu un citiem saturiem no visas pasaules.

Produkts ir paredzēts dažāda vecuma personām, kas vēlas klausīties un sinhronizēt mūziku starp dažādām ierīcēm un sekot savu mīļāko mākslinieku jaunajiem un vecajiem singliem.

Spotify ir šādas galvenās funkcijas:

1) mūzikas, video un podkastu straumēšana;
2) atskaņošanas sarakstu veidošana;
3) piekļuve audio grāmatām;
4) citas funkcijas:
    a. vietējās krātuves savienošana ar aplikāciju;
    b. klausīšanās bezsaistē.

Platformas izkārtojums ir atpazīstams un ikonisks. Tomēr saskarnei pielāgojamības iespēju nav plašu, tās dodot tikai maksas lietotājiem. Platforma atbalsta vairums ierīču veidus, palielinot savu lietotāju klāstu. (skatīt 3. attēls)



3. attēls ***Spotify* izskats dažādās ierīcēs [3]**

Spotify nodrošina lietošanas paskaidrojumus viņu mājaslapā tiešsaistē, kas paskaidro dažādas funkcijas un darbības attiecīgi pēc ierīces veida.

1.4.3.2. Salīdzinājums

Apkopojot lietotāju atsauksmes no *Google Play* veikala, aplikācijas stiprās puses ir plašais audio bibliotēku klāsts, intuitīvā saskarne, lieliska skaņas kvalitāte un personalizētie mūzikas ieteikumi. Tomēr lietotāji izsaka neapmierinātību par biežajām reklāmām bezmaksas versijā, nepietiekamu klientu atbalstu tehnisku problēmu gadījumos, mākslīga intelekta (*AI*) integrācijas kvalitātes trūkumiem, viltus mākslinieku klātbūtni katalogā, kā arī par *Premium* versijas nepilnību, turpinot rādīt reklāmas maksājošiem klientiem.

Prototips nesaskarsies ar vairums no šīm problēmām, jo programmas darbība notiek tikai bezsaistē. Salīdzinot ar *Spotify* aplikāciju, projekta darbības sfēra ir daudz vienkāršāka, kas mazina iespējamās neērtības lietošanas laikā – neiekļaujot reklāmas un nevēlamus autoru darbus. Jāmin, ka *Spotify* plašais ierīču atbalsta klāsts ir nesalīdzināms ar prototipa prasībās noteikto, kas noteikti dod priekšrocību konkurentam.

1.4.4. *Windows Media Player*

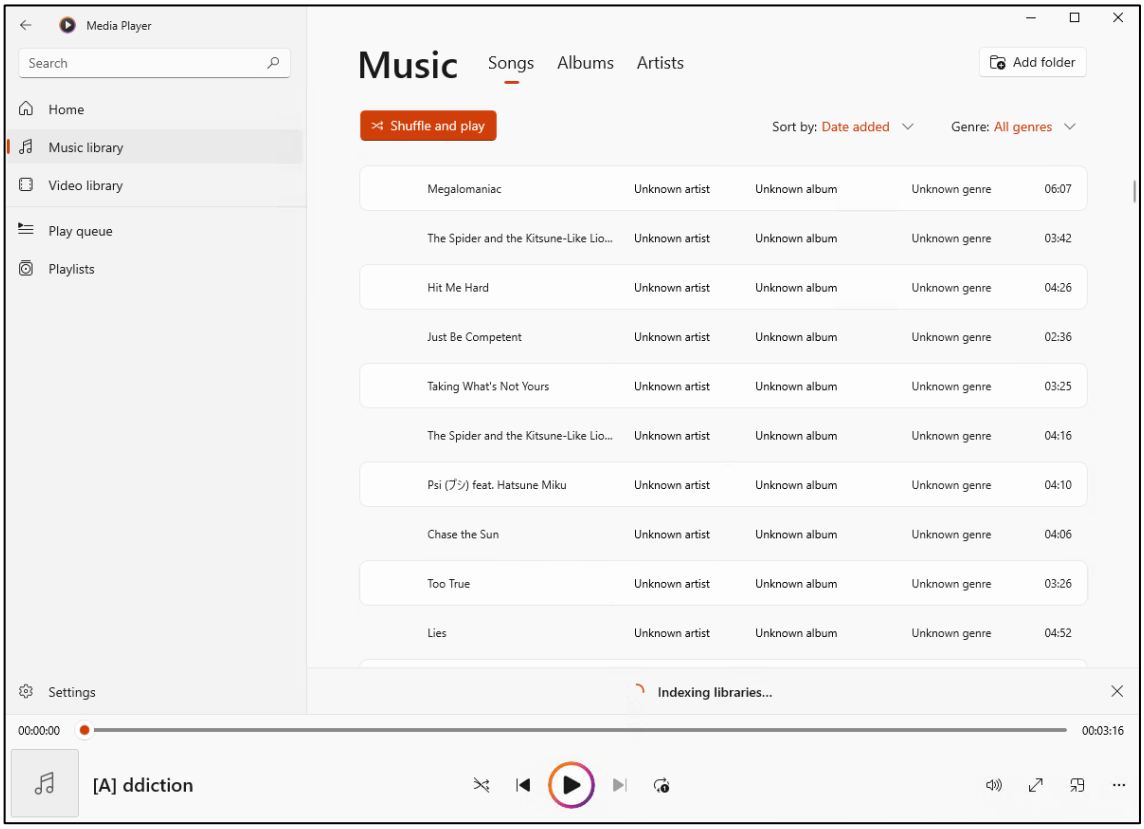
1.4.4.1. Apraksts

Media Player ir multivides atskaņotājs priekš *Windows* sistēmām, kas par savu galveno mērķi uzskata ērtu un patīkamu multimediju satura klausīšanos un skatīšanos *Windows* lietotājiem, aizvietojot to priekšteci *Gorove Music*.

Media Player ir sekojošās galvenās funkcijas:

1) mūzikas un video atskaņošana;
2) sarakstu izveide un pārvaldība;
3) automātiska failu atlase no vietējās datora krātuves;
4) karstie taustiņi un pieejamības funkciju atbalsts;
5) citas funkcijas:
    a. mini atskaņotājs;
    b. automātiska trūkstošo *codec* instalācija no *Microsoft* veikala;
    c. motīvu iestatījumi;
    d. atskaņošanas ātruma un skaļuma regulēšana.

Programmai ir paredzēta plaša vecuma klāstam un ir vizuāli pievilcīga (skatīt 4. attēls). Tā ir veidota vienkāršai lietošanas pieredzei, atbalstot paplašinātas pieejamības opcijas.



4. attēls ***Media player*** **noklusējuma izskats [Autora veidota ekrānkopija]**

Media Player nodrošina lietotāju atbalstu tiešsaistē *Microsoft* atbalsta mājaslapā. Šeit var atrast diskusijas forumos, tehnisku informāciju par programmu un konkurētu darbību paskaidrojumus, meklējot ar atslēgvārdiem.

1.4.4.2. Salīdzinājumi

Apkopojot lietotāju atsauksmes no *Microsoft* veikala un atsevišķiem novērtētājiem, programmas stiprās puses ir vienkāršā un saprotamā saskarne, automātiskā failu atlase un pilnvērtīgs papildu funkciju klāsts. Tomēr lietotāji izsaka neapmierinātību ar programmas optimizāciju priekš sistēmām ar minimāliem resursiem, aizņemot ievērojamu laiku liela daudzuma mediju failu kārtošanā un novērojot darbību atpalikšanu.

Prototips tiek būvēts ar nolūku efektīvi strādāt ar salīdzinoši minimāliem resursiem. Saskarnes izstrādes plānošanā iecerēts ņemt par piemēru *Media Player* saskarni un lietošanas pieredzi.

1.4.5. Secinājums

Līdzīgu produktu analīze atklāj, ka, lai gan katram produktam ir savas stiprās puses, visiem piemīt noteikti trūkumi.

Visos produktos var redzēt kopīgu tendenci, kompromisu starp funkcionalitāti, lietošanas pieredzi vai veiktspējas stabilitāti. *HUAWEI Music*, neskatoties uz tā veiksmīgi izstrādāto saskarni un plašajām personalizācijas iespējām, cieš no atjauninājumu nestabilitātes, kas negatīvi ietekmē lietotāju pieredzi. Līdzīgi, *Windows Media Player*, kam piemīt elegants dizains un pilnīgs funkcionalitātes klāsts, cieš no veiktspējas problēmām sistēmās ar ierobežotiem resursiem. Savukārt *Spotify*, neskatoties uz tā milzīgo popularitāti un plašo satura klāstu, cieš no uzbāzīgām reklāmām, saskarnes limitiem un nevēlamiem aktieriem, kas pasliktina lietotāja pieredzi.

Šī analīze tieši apstiprina prototipa izstrādes mērķa aktualitāti. Prototipu var pasniegt kā risinājumu, apvienojot analizēto produktu stiprās puses – intuitīva saskarne, pilnīgs funkciju klāsts un personalizācijas iespējas – un risinot to galvenos trūkumus. Kopā ar pilnīgu dokumentāciju un palīgfunkciju ieviešanu, prototipu var nostādināt kā konkurētspējīgu risinājumu mūzikas atskaņošanas programmu klāstā.

2. UZDEVUMU RISINĀŠANAS LĪDZEKĻU PAMATOJUMS

2.1. Ievads

Šajā sadaļā tiek aprakstītas un pamatotas galvenās tehnoloģijas un rīki, kas izmantoti prototipa izstrādē. Pamatojums balstās uz tehnoloģiju piemērotību projekta prasībām, izstrādes efektivitāti un ekonomisku izdevīgumu, sekojot klienta vajadzībām.

2.2. Izstrādes vide

Prototipa izstrādes procesam ir specializēta izstrādes vide, kas nodrošina efektīvu darbu un atbilstību projekta prasībām. Galvenā izstrādes vide balstās uz *Arch Linux* operētājsistēmu, kas izvēlēta dēļ tās modernās programmatūras pieejamības, plašās dokumentācijas, iespējām ātri iegūt jaunākās izstrādes rīku versijas un uz projekta izstrādātāja ietekumu.

Pamata izstrādes stacijas uzstādījums:

- operētājsistēma: *Arch Linux* ar *i3wm* darbvirsmas vidi/logu pārvaldītāju;
- procesors: *Intel* i5-10300H (8) @ 4.500GHz;
- atmiņa: 16 GB RAM;
- krātuve: SSD 500 GB;
- grafiskā karte: *Intel CometLake-H* GT2 [UHD *Graphics*] un *NVIDIA GeForce* GTX 1650 *Mobile / Max-Q*.

Sistēma ļauj vienlaicīgi strādāt ar *IDE*, datubāzēm un testēt programmu bez veiktspējas ierobežojumiem un ir pietiekami resursi virtuālo mašīnu darbināšanai testēšanas nolūkiem. *Arch Linux* vide nodrošina stabilu izstrādi ar jaunākajām *Java* un *JavaFX* versijām.

2.3. Produkta izstrādes vides iekārtošana

2.3.1. *IntelliJ IDEA*

Prototipa izstrādei tiek izmantota *IntelliJ IDEA* 2025.2.3 (*Community Edition*) izstrādes vide. Šīs IDE izvēle ir pamatota ar tās piemērotību *Java* un *JavaFX* projektu izstrādei, kā arī ar tās pieejamību bez maksas.

IntelliJ IDEA nodrošina vairākas praktiskas priekšrocības:

- kvalitatīvu koda automātisko pabeigšanu un sintakses analīzi, kas samazina kļūdu skaitu izstrādes laikā;

- integrētu atkļūdošanas vidi, kas atvieglo programmas loģikas pārbaudi un kļūdu novēršanu;
- ērtu *Maven* projekta konfigurēšanu un atkarību pārvaldību;
- atbalstu *JavaFX* lietotņu izstrādei un palaišanai, kas ir būtiski grafiskā lietotāja interfeisa prototipēšanai.

Līdz ar *to IntelliJ IDEA* ir piemērota un pamatota izvēle šī prototipa izstrādei, jo tā uzlabo izstrādes procesa efektivitāti un palīdz nodrošināt stabilāku gala rezultātu.

2.3.2. *SQLite*

Prototipa datu glabāšanai tiek izmantota SQLite datubāze (projekta *Java* pusē caur *sqlite-jdbc* draiveri). Šīs datubāzes izvēle ir pamatota ar tās piemērotību darbvirsmas lietotnei, kurai jāspēj strādāt bezsaistē, kā arī ar vienkāršu integrāciju Java projektā.

SQLite nodrošina vairākas praktiskas priekšrocības:

- datu glabāšanu vienā lokālā failā bez atsevišķa servera;
- vienkāršāku izvietošanu lietotāja datorā, jo nav jāuztur DB serveris;
- pietiekamu veiktspēju bibliotēkas metadatu, atskaņošanas vēstures un sarakstu glabāšanai;
- drošāku un pārskatāmāku lokālo datu pārvaldību prototipa līmenī.

Līdz ar to *SQLite* ir piemērota un pamatota izvēle šī prototipa izstrādei, jo tā nodrošina vienkāršu arhitektūru un atbilst projekta bezsaistes prasībām.

2.3.3. *DBeaver*

Datubāzes struktūras un datu pārvaldībai izstrādes laikā tiek izmantots *DBeaver Community* 25.2. Šī rīka izvēle ir pamatota ar tā universālo pielietojumu *SQL* datubāžu darbā un pieejamību bez maksas.

DBeaver nodrošina vairākas praktiskas priekšrocības:

- ērtu tabulu, kolonnu un indeksu struktūras pārskatīšanu;
- ātru *SQL* vaicājumu izpildi un rezultātu analīzi;
- vienkāršu testa datu ievadi un rediģēšanu;
- pārskatāmu vidi datu modeļa validācijai izstrādes gaitā.

Līdz ar to *DBeaver* ir piemērota un pamatota izvēle šī prototipa izstrādei, jo tas paātrina darbu ar datubāzi un samazina konfigurācijas sarežģītību.

2.3.4. *Git* un *GitLab*

Prototipa koda versiju kontrolei tiek izmantots *Git* 2.51.1 kopā ar *GitLab* platformu. Šīs kombinācijas izvēle ir pamatota ar nepieciešamību uzturēt pārskatāmu izmaiņu vēsturi un nodrošināt strukturētu izstrādes procesu.

Git un *GitLab* nodrošina vairākas praktiskas priekšrocības:

- precīzu izmaiņu izsekojamību un iespēju atgriezties pie iepriekšējām versijām;
- drošu darbu ar zariem (*branches*), atdalot jaunu funkciju izstrādi;
- centralizētu repozitoriju sadarbībai un koda pārskatīšanai;
- CI/CD (*Continued Integration/Continued Development*) procesu integrāciju būvēšanai un testu izpildei.

Līdz ar to *Git* un *GitLab* ir piemērota un pamatota izvēle šī prototipa izstrādei, jo tie uzlabo kvalitātes kontroli un izstrādes procesa pārvaldību.

2.3.5. *Java*

Kā galvenā programmēšanas valoda prototipa izstrādei tiek izmantota *Java* 25. Šīs valodas izvēle ir pamatota ar tās saderību ar projekta tehnoloģisko steku un piemērotību vairākplatformu darbvirsmas lietotnes izstrādei.

Java nodrošina vairākas praktiskas priekšrocības:

- stabilu izpildi *JVM* vidē dažādās operētājsistēmās;
- plašu bibliotēku un rīku ekosistēmu;
- ērtu integrāciju ar *JavaFX*, *Maven* un *JUnit*;
- vienotu izstrādes pieeju gan *Windows*, gan *Linux* vidē.

Līdz ar to *Java* ir piemērota un pamatota izvēle šī prototipa izstrādei, jo tā nodrošina tehnoloģisku saderību un uzticamu pamatu lietotnes realizācijai.

2.3.6. *JavaFX*

Grafiskā lietotāja interfeisa izstrādei tiek izmantots *JavaFX* 21.0.6. Šīs platformas izvēle ir pamatota ar tās piemērotību *Java* darbvirsmas lietotņu izstrādei un nepieciešamo multimediju iespēju pieejamību.

JavaFX nodrošina vairākas praktiskas priekšrocības:

- modernu lietotāja interfeisa komponentu kopu;
- notikumu apstrādes mehānismus interaktīvai saskarnei;
- iebūvētu atbalstu multimediju funkcionalitātei;

- tiešu integrāciju ar *Java* projektu bez papildu sarežģītas konfigurācijas.

Līdz ar to *JavaFX* ir piemērota un pamatota izvēle šī prototipa izstrādei, jo tā ļauj efektīvi veidot funkcionālu un vizuāli pārskatāmu mūzikas atskaņotāja saskarni.

2.3.7. *JUnit*

Prototipa testēšanai tiek izmantota *JUnit* 5.12.1 testēšanas bibliotēka. Šīs bibliotēkas izvēle ir pamatota ar tās plašo pielietojumu Java projektos un piemērotību vienību testu izveidei.

JUnit nodrošina vairākas praktiskas priekšrocības:

- sistemātisku kritisko loģikas daļu pārbaudi;
- agrīnu kļūdu atklāšanu pirms nākamo funkciju ieviešanas;
- drošāku koda pārstrukturēšanu (refaktorēšanu);
- vienotu testu izpildi gan lokāli, gan CI vidē.

Līdz ar to *JUnit* ir piemērota un pamatota izvēle šī prototipa izstrādei, jo tā palīdz paaugstināt koda kvalitāti un samazināt regresijas risku.

2.3.8. *Maven*

Projekta būvēšanai un atkarību pārvaldībai tiek izmantots *Maven* (ar *Maven Wrapper mvnw*). Šī rīka izvēle ir pamatota ar nepieciešamību nodrošināt standartizētu un reproducējamu būvēšanas procesu.

Maven nodrošina vairākas praktiskas priekšrocības:

- centralizētu atkarību pārvaldību;
- automatizētu kompilācijas, testēšanas un pakotņu veidošanas procesu;
- vienotu projekta dzīves cikla pārvaldību izstrādes un CI vidē;
- ērtu integrāciju ar *Java* un *JavaFX* projektu konfigurāciju.

Līdz ar to *Maven* ir piemērota un pamatota izvēle šī prototipa izstrādei, jo tas uzlabo izstrādes procesa stabilitāti un nodrošina konsekventus būvēšanas rezultātus.

2.3.9. *CodeScene.io*

Projekta koda kvalitātes uzraudzībai un regresijas risku agrīnai identificēšanai tiek izmantots *CodeScene.io* rīks. Šī rīka izvēle ir pamatota ar tā spēju analizēt ne tikai statisku kodu, bet arī izmaiņu vēsturi, tādējādi palīdzot savlaicīgi pamanīt riskantās koda zonas.

CodeScene.io nodrošina vairākas praktiskas priekšrocības:

- koda veselības novērtējumu, kas palīdz identificēt tehniskā parāda uzkrāšanos;
- regresijas riska indikācijas, balstoties uz izmaiņu biežumu un koda sarežģītību;
- prioritizētu skatījumu uz failiem un moduļiem, kuriem nepieciešama refaktorēšana;
- pārskatāmus kvalitātes pārskatus, ko iespējams izmantot izstrādes plānošanā un koda pārskatē.

Līdz ar to *CodeScene.io* ir piemērota un pamatota izvēle šī prototipa izstrādei, jo tas palīdz pieņemt datos balstītus lēmumus par koda kvalitātes uzlabošanu un samazina regresijas risku turpmākā attīstībā.

2.4. Tehniskie ierobežojumi

Sekojot projekta prasībām, izmantotajam tehnoloģiskajam stekam un izstrādes vides robežām, tiek definēti šādi tehniskie ierobežojumi.

1) **Ārējie faktori**
   - **a.** izstrādes periods ir ierobežots līdz kvalifikācijas darba iesniegšanas termiņam, tādēļ prioritāri tiek realizētas pamatfunkcijas, bet daļa papildu funkcionalitātes tiek atlikta;
   - **b.** projekta risinājumi tiek izvēlēti ar mērķi nodrošināt stabilu demonstrējamu prototipu noteiktā laikā, nevis pilna mēroga produkta funkcionalitāti.
2) **Resursu lietojuma ierobežojumi**
   - **a.** prototips ir paredzēts darbībai datoros ar vismaz 4 GB operatīvās atmiņas; ieteicams nodrošināt vismaz 1 GB brīvu RAM lietotnes palaišanas un bibliotēkas ielādes brīdī;
   - **b.** mūzikas bibliotēkas ielādes ātrumu būtiski ietekmē diska ievades/izvades (I/O) veiktspēja un datu apjoms;
   - **c.** lielām lokālajām mūzikas kolekcijām sākotnējā skenēšana var aizņemt ilgāku laiku, īpaši uz lēnākiem datu nesējiem.
3) **Platformas un izpildes vides ierobežojumi**
   - **a.** prototips ir testēts un pilnvērtīgi atbalstīts *Windows* 10 un *Arch Linux* (x86_64) vidē - citās operētājsistēmās vai versijās darbība netiek pilnībā garantēta;
   - **b.** izplatāmās pakotnes ir paredzētas x86_64 arhitektūrai (*Windows* `.msi` un *Linux* `.AppImage`), tādēļ citām arhitektūrām (piemēram, ARM) nepieciešama atsevišķa būvēšana un validācija;

c. *Linux* vidē noteiktu audio formātu (īpaši M4A/AAC) atskaņošanai nepieciešami sistēmā uzstādīti *GStreamer* spraudņi;

d. *Linux* `.AppImage` palaišanai dažās sistēmās var būt nepieciešams FUSE atbalsts, ja tas nav pieejams, jāizmanto alternatīvais palaišanas režīms ar `--appimage-extract-and-run`;

e. *JavaFX* grafiskās saskarnes stabilitāte un veiktspēja ir atkarīga no videokartes draiveru saderības, tāpēc novecojušās sistēmās iespējami attēlošanas traucējumi.

4) Izstrādes un piegādes ierobežojumi

a. projekts tiek būvēts ar *Java/JDK* 25 rīkkopu, un izmantotā būvēšanas konfigurācija ir pielāgota mērķa vidēm (*Windows* 10 un *Linux* x86_64);

b. instalatoru/pakotņu būvēšanas laikā var būt nepieciešama interneta piekļuve ārējo resursu (piemēram, fontu pakotnes) lejupielādei;

c. *Windows* `.msi` pakotnei bez koda paraksta var parādīties *SmartScreen* brīdinājums, kas ir tipisks neparakstītām instalācijas pakotnēm.

3. PRASĪBU SPECIFIKĀCIJA

3.1. Ievads

Šajā nodaļā tiek aprakstītas visas ar prasībām saistītās detaļas, iekļaujot aprakstu par funkcionālām un nefunkcionālām prasībām. Sadaļa iekļauj tikai plānošanas fāzē izstrādāto nevis gala rezultātu.

3.1.1. Darbības sfēra

Projekta darbības sfēra ir izstrādāt darbvirsmas audio atskaņotāja prototipu, kas paredzēts mūzikas klausīšanai bezsaistē. Projekta izveide balstās uz vajadzību pēc vienkāršas, bet funkcionālas lietotnes, kas nodrošina visu nepieciešamo audio atskaņošanas iespēju kopumu.

3.2. Mērķauditorija

Programmas izveide fokusējas uz vecuma grupu no 12 līdz 40 gadiem (jaunākas paaudzes), kas regulāri izmanto datorus ikdienas darbībās un ir pieraduši pie dažādām lietotāja saskarnēm. Tomēr, produkts arī ņem vērā saskarnes draudzīguma svarīgumu, tāpēc liels fokuss tiek ņemts uz lietošanas vienkāršumu neatkarīgi no prasmēm.

Nepieciešamās prasmes:

- angļu valodas zināšanas;
- spēja orientēties grafiskā lietotāja saskarnē;
- pamata izpratne par audio atskaņošanas programmu lietotajiem simboliem (favorītu, pauzēt/spēlēt u.tml simboliem);
- pamata datora lietošanas prasmes (peles un klaviatūras taustiņu spiešana, mapju atlase).

Mērķauditoriju veido gan ikdienas lietotāji, kas vēlas vienkāršu un efektīvu mūzikas klausīšanās pieredzi, gan spējīgāki lietotāji, kas novērtē papildus funkcionalitāti kā pielāgojamu saskarni un karstos taustiņus. Lietotāji var būt gan skolēni/studenti, kas izmanto programmu atpūtai, gan pieaugušie, kam nepieciešama fona mūzika darba vidē vai atpūtai mājās.

Lai noteiktu mērķauditoriju un tās vajadzības, tika veikta šādas izpētes metodes:

- esošo mūzikas atskaņotāju lietotāju atsauksmju analīze;
- tiešas konsultācijas ar klientu.

Šīs izpētes rezultāti ļāva definēt funkcionālās un nefunkcionālās prasības, kā arī optimizēt lietotāja saskarni atbilstoši mērķauditorijas sagaidītajai lietošanas pieredzei.

3.3. Vispārīgs apraksts

3.3.1. Sistēmas pārskats

Prototips ir veidots kā alternatīvs populāriem darbvirsmu audio atskaņotājiem. Projektā tiek izstrādāta programma, kas piedāvā lietotājiem intuitīvu dizainu un personalizācijas iespējas.

Sistēmas arhitektūra ir sadalīta trīs loģikas slāņos, kas nodrošina, ka katru sistēmas daļu var attīstīt un uzturēt atsevišķi, netraucējot pārējās programmas daļas. Prezentācijas slānis pārvalda lietotāja saskarni un ievadi. Darbības slānis iekļauj audio un loģikas apstrādi. Visbeidzot, datu slānis pārvalda vietējo krātuvi un tā saistītās darbības.

Prototips darbojas neatkarīgi no citām programmām, nodrošinot pilnīgu funkcionalitāti bezsaistē.

3.3.2. Galvenās sistēmas komponentes

Sistēma sastāv no vairākiem savstarpēji saistītiem moduļiem, kas organizēti pēc slāņu arhitektūras principa: prezentācijas slānis, darbības slānis, datu slānis.

3.3.2.1. Prezentācijas slānis

Prezentācijas slānis pārvalda lietotāja saskarni un ievadi. Tas ietver visus logu, skatu un dialoga elementus, kā arī to mijiedarbības loģiku. Slānī tiek realizēta navigācija starp galvenajiem skatiem, piemēram, dziesmas, albumi, vēsture, u.c. Vizuālais noformējums ir atdalīts no loģikas, izmantojot *CSS* krāsu tēmas, kas ļauj dinamiski mainīt izskatu bez koda izmaiņām un ļauj uzturēt vienotu izskatu. Fontu maiņa un izmēru maiņa ir daļa no vizuāls apstrādes loģikas

3.3.2.2. Darbības slānis

Darbības slānis realizē visu programmas biznesa loģiku, kas nav tieši saistīta ar lietotāja saskarni vai datu glabāšanu. Tas darbojas kā starpnieks starp prezentācijas un datu slāni, apstrādājot lietotāja komandas un sagatavojot datus attēlošanai. Slānis ir organizēts četrās galvenajās funkcionālajās grupās:

- **audio atskaņošanas servisi** – nodrošina audio failu atskaņošanu, pauzēšanu, skaļuma regulēšanu un atskaņošanas pozīcijas pārvaldību.

- **atskaņošanas rindas un vēstures servisi** – pārvalda pašreizējo atskaņošanas rindu (dziesmu pievienošana, izņemšana, pārkārtošana, saglabāšana starp sesijām) un automātiski fiksē noklausīto dziesmu vēsturi ar laika zīmogiem.
- **bibliotēkas pārvaldības servisi** – uzrauga lietotāja norādītās mapes uz failu sistēmas izmaiņām, veic audio failu indeksēšanu, nolasa ID3 tagus un saglabā metadatus datubāzē.
- **papildservisi un pārvaldnieki** – nodrošina mīļāko dziesmu pārvaldību, albuma vāka attēlu kešošanu, globālo karsto taustiņu apstrādi, navigācijas loģiku starp skatiem un atskaņošanas vadības komandu koordināciju no dažādām saskarnes daļām.

Darbības slānis ir pilnīgi neatkarīgs no prezentācijas slāņa – visa komunikācija notiek caur skaidri definētām metožu izsaukumiem un atgriezeniskajiem datu modeļiem.

3.3.3. Lietošanas vide un sistēmas prasības

Programmas lietošanai ir nepieciešamas sekojošās sistēmas prasības pēc operētājsistēmas.

***Windows*:**

- **OS:** Windows (64-bit / x64 ierīces);
- **CPU arhitektūra:** x86_64;
- **Instalācijas prasības:** lietotājam ir jābūt spējīgam instalēt programmas ar `.msi` paplašinājumu (var rādīties brīdinājums, jo instalētājs nav parakstīts).

***Linux*:**

- **OS:** Linux (x64 / x86_64);
- **CPU arhitektūra:** x86_64;
- **Palaišanas prasības:** `.AppImage` aplikācijai jābūt atļaujai būt palaižamai (`chmod +x`);
- **Audio *codec* prasība:** Lai atskaņotu M4A/AAC failus, sistēmā jābūt instalētām *GStreamer* spraudnis (*base, good, libav* pakotnes).

3.4. Funkcionālās prasības

3.4.1. Ievads

Noteikt sistēmas funkcionālās prasības ir svarīgi, lai atvieglotu programmas izstrādes procesu un lai izprastu projekta apjomu. Šajā nodaļā tiek aprakstītas konkrētas darbības un iespējas, kas jārealizē prototipā, lai tas atbilstu klienta vajadzībām un projekta mērķiem. Katra funkcionālā prasība ir strukturēta atsevišķi, norādot tās mērķi un darbības principu.

3.4.2. Funkcionālo prasību saraksts

3.4.2.1. Audio failu atskaņošana

Prasības saturs:

Sistēmai jānodrošina lokālo audio failu atskaņošana no bibliotēkas un rindas, izmantojot *JavaFX media* atskaņošanas mehānismu kā pamata risinājumu. Atskaņošana jāuzsāk no lietotāja izvēlētā ieraksta, un sistēmai korekti jāapstrādā atskaņošanas stāvokļi (atskaņot / pauzēt / apstāties / ieraksta beigas).

Sagaidāmais rezultāts:

Lietotājs var izvēlēties dziesmu un uzreiz to atskaņot; atskaņošanas statuss tiek attēlots saskarnē un korekti mainās darbību laikā.

3.4.2.2. Formātu atbalsts

Prasības saturs:

Sistēmai jāatbalsta vismaz MP3, M4A, AAC, WAV un FLAC formāti. Realizācijā jāņem vērā, ka daļa formātu dekodēšanas *Linux* vidē balstās uz sistēmās *GStreamer* pieejamību (īpaši M4A/AAC gadījumos).

Sagaidāmais rezultāts:

Lietotājs var importēt un atskaņot klienta obligāti prasītos formātus; neatbalstīta vai bojāta faila gadījumā tiek sniegta saprotama kļūdas pazīme/paziņojums.

3.4.2.3. Pamata atskaņošanas vadība

Prasības saturs:

Jārealizē pamatvadība: atskaņot, pauzēt, apturēt, nākamais un iepriekšējais ieraksts, kā arī skaļuma un progresa pozīcijas kontrole. Vadības elementi jāsinhronizē ar aktuālo atskaņotāja stāvokli.

Sagaidāmais rezultāts:

Lietotājs var pilnībā kontrolēt atskaņošanu no galvenās saskarnes; darbības notiek bez nekonsekventiem stāvokļiem un ar paredzamu uzvedību.

3.4.2.4. Atskaņošanas rindu (*queue*) pārvaldība

Prasības saturs:

Sistēmai jānodrošina *queue* veidošana un pārvaldība: ierakstu pievienošana rindai, secības maiņa, atsevišķu ierakstu izņemšana un rindas tīrīšana. Atskaņošanas sistēmai jāstrādā ar aktuālo rindas secību.

Sagaidāmais rezultāts:

Lietotājs var izveidot sev vēlamu atskaņošanas secību un dinamiski to mainīt; atskaņotājs pāriet uz nākamo ierakstu pēc rindas loģikas.

3.4.2.5. Mūzikas bibliotēkas pārvaldība

Prasības saturs:

Jānodrošina bibliotēkas uzturēšana no lokālajiem avotiem (mapju/failu pievienošana), ierakstu indeksēšana un glabāšana lokālajā datu slānī (*SQLite*). Jāparedz bibliotēkas stāvokļa atjaunošana starp palaišanas reizēm un lietotāja izraisītu atjaunošanu.

Sagaidāmais rezultāts:

Pēc avotu pievienošanas bibliotēka satur pieejamos audio ierakstus ar to pamatinformāciju, pēc programmas pārstartēšanas bibliotēkas dati saglabājas.

3.4.2.6. Meklēšana bibliotēkā

Prasības saturs:

Jārealizē meklēšana pēc metadatiem (nosaukums, albums, autori), izmantojot bibliotēkā saglabātos datus. Meklēšanas rezultāti jāatgriež ātri un lietotājam saprotamā veidā.

Sagaidāmais rezultāts:

Ievadot atslēgvārdu, lietotājs saņem atbilstošu ierakstu sarakstu un var nekavējoties sākt atskaņošanu vai pievienot rezultātus *queue*, kā arī redzēt rezultātus par autoriem, albumiem un atskaņošanas sarakstiem.

3.4.2.7. Metadatu nolasīšana

Prasības saturs:

Importa laikā sistēmai jānolasa audio failu metadati (piem., nosaukums, izpildītājs, albums, ilgums), izmantojot izvēlēto *Java* bibliotēku kombināciju metadatu apstrādei, un jānormalizē dati glabāšanai datubāzē.

Sagaidāmais rezultāts:

Bibliotēkā ieraksti tiek attēloti ar lietotājam jēgpilniem metadatiem, kas tālāk tiek izmantoti meklēšanai un atskaņošanas skatiem.

3.4.2.8. Saskarnes motīvu pielāgošana

Prasības saturs:

Jānodrošina tēmu (krāsu motīvu) maiņa lietotāja saskarnē, balstoties uz *JavaFX* stilu apstrādes pieeju (CSS resursi). Motīva izvēle jāpielieto visam interfeisam konsekventi.

Sagaidāmais rezultāts:

Lietotājs var pārslēgt motīvu, un interfeiss uzreiz pielāgojas izvēlētajam vizuālajam stilam. Jābūt iespējai atgriezties uz noklusējuma stilu.

3.4.2.9. Teksta noformējuma pielāgošana

Prasības saturs:

Sistēmai jāļauj lietotājam pielāgot teksta noformējumu (fonts, izmērs) pieejamības un lietojamības uzlabošanai. Risinājumam jāstrādā kopā ar tēmu sistēmu, neradot konfliktējošu stilu.

Sagaidāmais rezultāts:

Lietotājs var pielāgot teksta attēlojumu savām vajadzībām, un izmaiņas korekti atspoguļojas visās galvenajās saskarnes zonās.

3.4.2.10. Karsto taustiņu atbalsts

Prasības saturs:

Jāievieš karstie taustiņi biežākajām atskaņošanas darbībām (vismaz *play/pause, next, previous*; pēc vajadzības arī papildu komandas). Taustiņu apstrādei jābūt stabilai un paredzamai aktīvā lietotnes loga kontekstā.

Sagaidāmais rezultāts:

Lietotājs var kontrolēt atskaņošanu ar tastatūru bez peles izmantošanas, tā paātrinot ikdienas lietošanu.

3.4.2.11. Lietotāja iestatījumu saglabāšana

Prasības saturs:

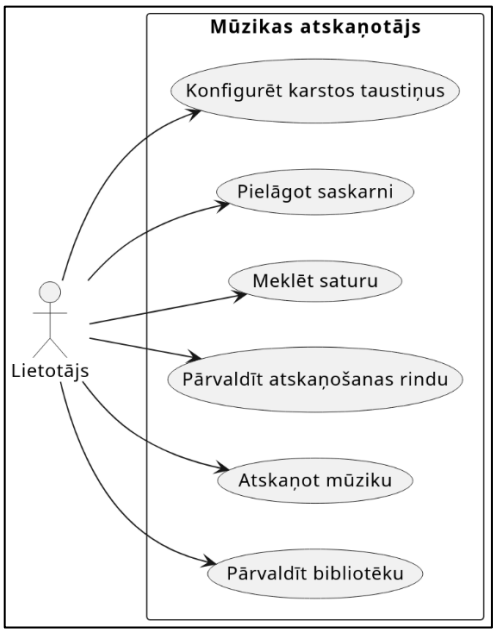
Jārealizē konfigurācijas sistēma, kas saglabā lietotāja izvēles (piem., tēma, teksta parametri, atskaņošanas iestatījumi un citi būtiski parametri) lokāli starp sesijām.

Sagaidāmais rezultāts:

Pēc programmas aizvēršanas un atkārtotas palaišanas lietotājs saņem iepriekšējo konfigurāciju bez atkārtotas manuālas iestatīšanas.

3.4.3. Lietojumgadījumu diagrammas

Lietojumgadījumu diagrammas dod vizuālu priekšstatu par lietotāja mijiedarbību ar programmu, pārskatāmā veidā atspoguļojot sistēmas funkcionālās prasības. Sadaļa apskatīs piecus galvenos lietojuma scenārijus: vispārīgo pārskatu (skatīt 5. attēls), detalizētās diagrammas (skatīt no 6. attēls līdz 10. attēls). Visu diagrammu aktieris ir Lietotājs. Attiecības `<<include>>` un `<<extend>>` attēlo attiecīgi obligātas apakšdarbības un papildu/alternatīvus scenārijus.



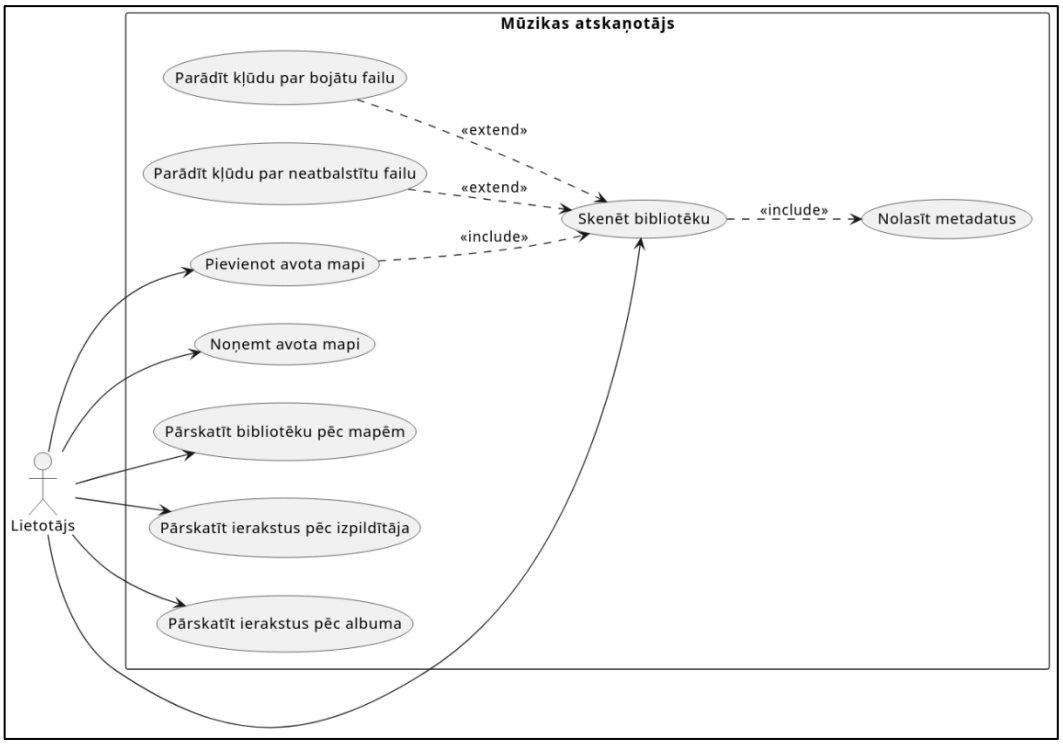
5. attēls **Vispārīga lietojumgadījuma diagramma [Autora veidots attēls]**

Galveno lietojumgadījumu pārskats:

- Pārvaldīt bibliotēku;
- Atskaņot mūziku;
- Pārvaldīt atskaņošanas rindu;
- Meklēt saturu;
- Pielāgot saskarni;
- Konfigurēt karstos taustiņus.

3.4.3.1. Bibliotēkas iestatīšanas un uzturēšanas diagramma

Šī diagramma attēlo pamata darbības, kas nepieciešamas, lai lietotājs varētu uzsākt programmas lietošanu un uzturēt mūzikas bibliotēku. Tās ir balsta funkcijas, no kurām ir atkarīgas pārējās lietotnes iespējas, īpaši pirmajā palaišanas reizē (skatīt 6. attēls).



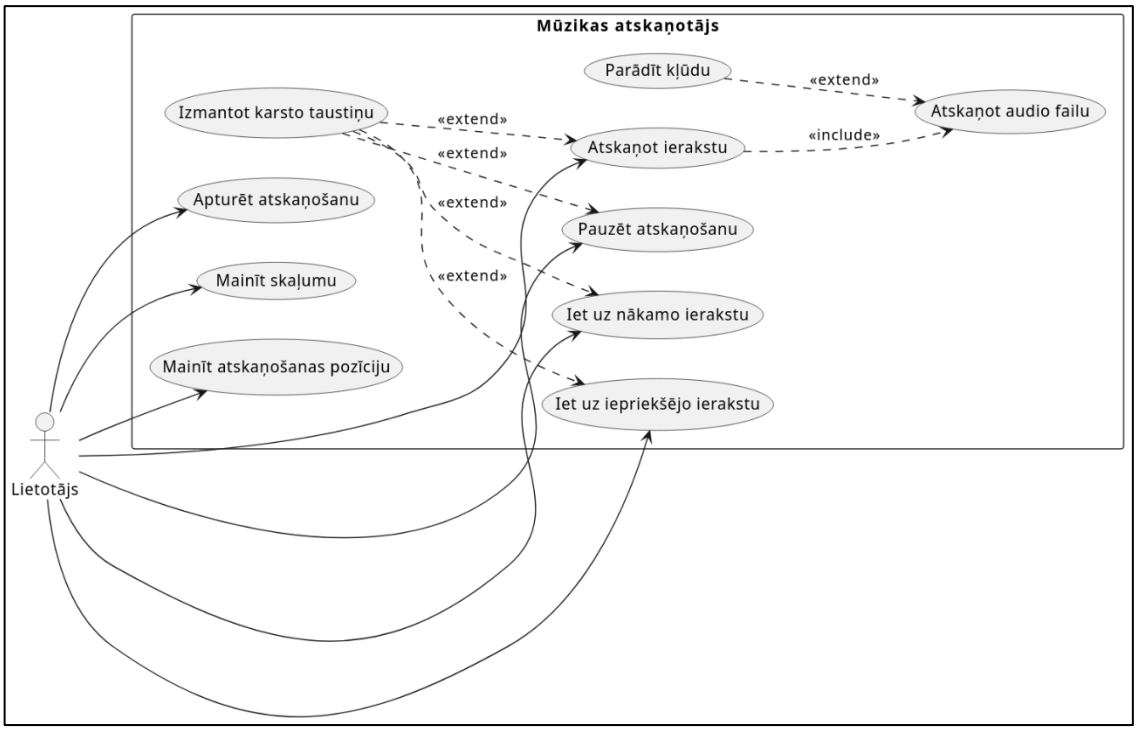
6. attēls **Bibliotēkas iestatīšanas un uzturēšanas diagramma [Autora veidots attēls]**

Diagrammā iekļautie lietojumgadījumi:

- Pievienot avota mapi;
- Skenēt bibliotēku;
- Noņemt avota mapi;
- Pārskatīt bibliotēku pēc mapēm;
- Pārskatīt ierakstus pēc izpildītāja;
- Pārskatīt ierakstus pēc albuma;
- Nolasīt metadatus (<> - obligāta darbība skenēšanas laikā);
- Parādīt kļūdu par neatbalstītu failu (<>);
- Parādīt kļūdu par bojātu failu (<>).

3.4.3.2. Atskaņošanas un vadības diagramma

Diagramma apraksta biežākās lietotāja darbības audio atskaņošanas vadībā - gan ieraksta palaišanu, gan tā pārtraukšanu un citus vadības parametrus (skatīt 7. attēls).



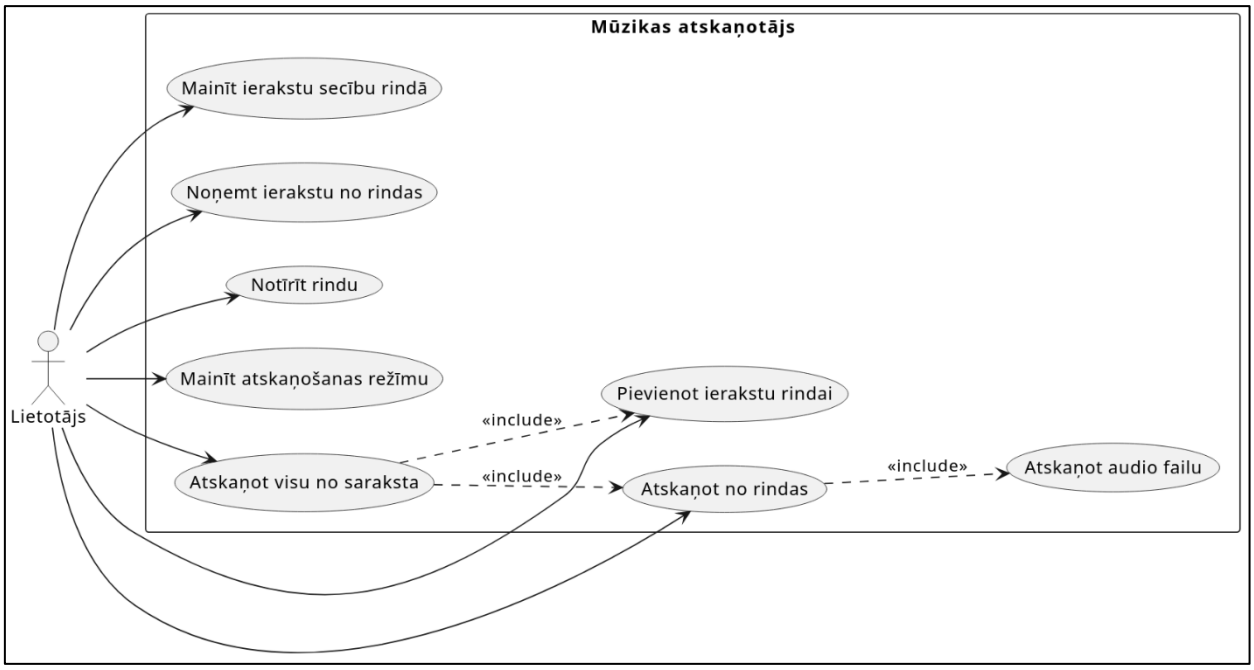
7. attēls **Atskaņošanas un vadības diagramma [Autora veidots attēls]**

Diagrammā iekļautie lietojumgadījumi:

- Atskaņot ierakstu;
- Atskaņot audio failu (<> - ietver faktisko atskaņošanu);
- Pauzēt atskaņošanu;
- Apturēt atskaņošanu;
- Pāriet uz nākamo ierakstu;
- Pāriet uz iepriekšējo ierakstu;
- Mainīt skaļumu;
- Mainīt atskaņošanas pozīciju;
- Izmantot karsto taustiņu (<> - papildu veids vadības darbībām);
- Parādīt atskaņošanas kļūdu (<>).

3.4.3.3. Atskaņošanas rindas (*queue*) pārvaldības diagramma

Diagramma attēlo lietojumgadījumus, kas saistīti ar aktīvās atskaņošanas rindas veidošanu un pārvaldību, kā arī ar atskaņošanas režīmu maiņu. Tā papildina 3.4.3.2 sadaļā aprakstīto atskaņošanas loģiku (skatīt 8. attēls).



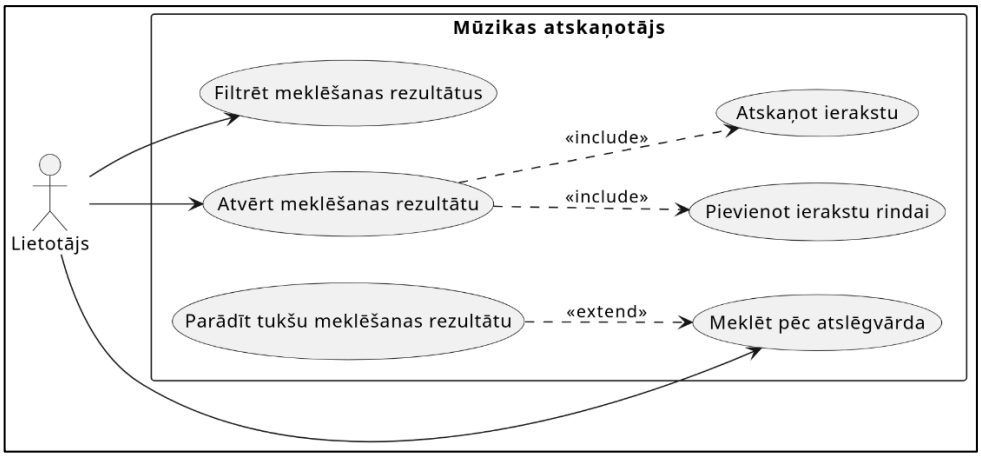
8. attēls **Atskaņošanas rindas (*queue*) pārvaldības diagramma [Autora veidots attēls]**

Diagrammā iekļautie lietojumgadījumi:

- Pievienot ierakstu rindai;
- Mainīt ierakstu secību rindā;
- Noņemt ierakstu no rindas;
- Notīrīt rindu;
- Mainīt atskaņošanas režīmu;
- Atskaņot no rindas;
- Atskaņot visu no saraksta;
- Atskaņot audio failu (<> - saistīts ar atskaņošanu no rindas);
- Pievienot ierakstu rindai (<> - saistīts ar "Atskaņot visu no saraksta").

3.4.3.4. Meklēšanas un satura atrašanas diagramma

Diagramma attēlo lietotāja darbības, kas ļauj atrast un izmantot saturu no lielākās bibliotēkas – meklēšanu, rezultātu filtrēšanu un turpmākās darbības ar atrasto saturu (skatīt 9. attēls)



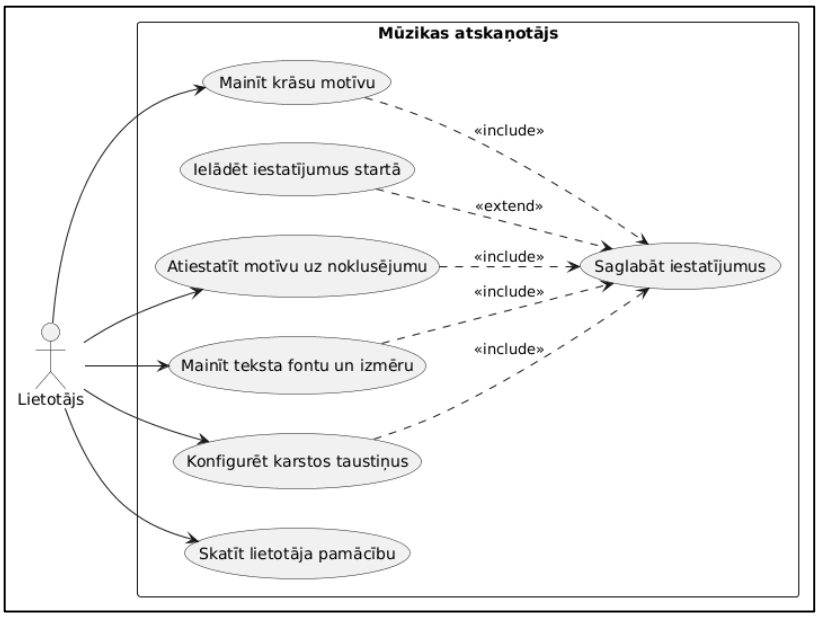
9. attēls **Meklēšanas un satura atrašanas diagramma [Autora veidots attēls]**

Diagrammā iekļautie lietojumgadījumi:

- Meklēt pēc atslēgvārda;
- Filtrēt meklēšanas rezultātus;
- Atvērt meklēšanas rezultātu;
- Atskaņot ierakstu (<> - pēc rezultāta atvēršanas);
- Pievienot ierakstu rindai (<> - pēc rezultāta atvēršanas);
- Parādīt tukšu meklēšanas rezultātu (<>).

3.4.3.5. Personalizācijas un iestatījumu diagramma

Diagramma attēlo lietotāja darbības, kas saistītas ar saskarnes pielāgošanu, karsto taustiņu konfigurāciju un iestatījumu saglabāšanu starp programmas palaišanas reizēm (skatīt 10. attēls).



10. attēls **Personalizācijas un iestatījumu diagramma [Autora veidots attēls]**

Diagrammā iekļautie lietojumgadījumi:

- Mainīt krāsu motīvu;
- Atiestatīt motīvu uz noklusējumu;
- Mainīt teksta fontu un izmēru;
- Konfigurēt karstos taustiņus;
- Saglabāt iestatījumus (<> - saistīts ar izmaiņu veikšanu);
- Ielādēt iestatījumus programmas startā (<> - sistēmas darbība palaišanas brīdī);
- Skatīt lietotāja pamācību.

3.5. Nefunkcionālās prasības

3.5.1. Ievads

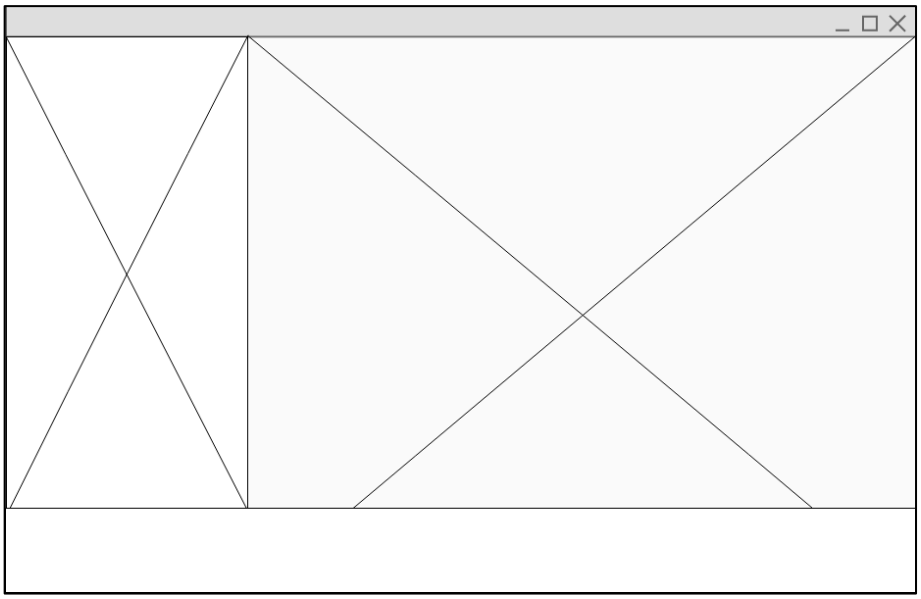
Nodaļa apraksta prototipa nefunkcionālās prasības - tās raksturo, cik labi un cik ērti sistēmai jādarbojas, nevis ko tā dara, kas aprakstīts 3.4 sadaļā. Šeit iekļautas lietojamība un saskarnes struktūra, vizuālā noformējuma prasības, veiktspējas un uzturamība.

Šīs prasības ir būtiskas, lai nodrošinātu, ka prototips darbojas efektīvi reālā vidē, uztur vienotu stilu un atbilst klienta prasībām.

3.5.2. Lietojamība un UI struktūra

Šajā apakšnodaļā tiek aprakstīti galvenie programmas struktūras elementi un navigācijas sistēma, kas nodrošina intuitīvu lietotāja pieredzi. Visi elementi ir izvietoti viegli apskatāmā izkārtojumā un atbilst mūsdienu lietotāja saskarnes standartiem.

Programmas struktūru var sadalīt 3 galvenajās daļās: navigācijas panelis, satura apgabals un mūzikas vadības panelis. Šīs tvertnes ir vienmēr redzamas programmas darbības laikā, tāpēc tās var uzskatīt par saskarnes struktūru. To izvietojums ir apskatāms attēlā (skatīt 11. attēls).



11. attēls **Vispārēja prototipa izkārtojuma struktūrskice [Autora veidots attēls]**

Navigācijas panelis (pa kreisi) kopā ar satura apgabalu (pa labi) ir struktūras elementi, kuru platumi ir lietotājam maināmi caur navigācijas paneļa labās malas vilkšanu. Satura apgabals attiecīgi pielāgo savu saturu šīm izmaiņām. Navigācijas paneļa platums nekad nedrīkst pārsniegt vairāk par pusi no programmas loga platuma. Audio vadības panelis atrodas skices apakšā. Tas savu izkārtojumu un izmēru neizmaina navigācijas paneļa manipulācijas ietekmē, tikai paša programmas loga izmēra manipulācijas laikā.

Papildus struktūras sadalījumam tiek ņemtas vērā šādas lietojamības prasības:

- **Saskarnes valoda:**

Prototipa lietotāja saskarne tiek plānota angļu valodā, atbilstoši klienta obligātajai prasībai. Visi virsraksti, pogas, ziņojumi un palīdzības teksts ir angliski, lai nodrošinātu vienotu pieredzi Windows un Linux vidēs.

- **Pārklājumi:**

Daļa funkciju netiek atvērta kā jauna navigācijas lapa, bet parādās virs pašreizējā satura apgabala. Tie ietver meklēšanu, atskaņošanas rindas paneli, lielo atskaņošanas skatu un grupu dziesmu sarakstu (atverot izpildītāju, albumu, mapi vai atskaņošanas sarakstu). Šie skati ir sīkāk aprakstīti sadaļās 3.5.5.4 - 3.5.5.6.

- **Vienots vizuālais stils:**

Saskarnei plānots konsekvents izskats, izmantojot vairākus krāsu motīvus un pielāgojamu teksta fontu un izmēru, kā aprakstīts funkcionālajās prasībās 3.4.2.8 un 3.4.2.9 sadaļās.

- **Loga izmērs un ritināšana:**

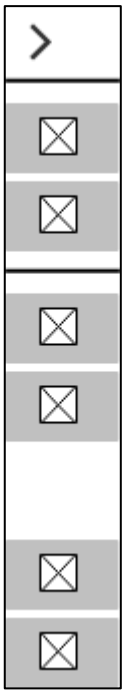
Prototipam jābūt lietojamam arī vidēji mazos logos. Navigācijas paneļa platums nevar pārsniegt aptuveni pusi no loga platuma. Bibliotēkas lapās galvenokārt ritinās dziesmu vai grupu saraksts, iestatījumu lapā ritinās viss saturs kā viena vertikāla kolonna un lietotāja pamācības lapā ritinās tēmu kartīšu saraksts.

3.5.3. Navigācijas panelis

Navigācijas panelis ir galvenais elements, ko lieto lietotājs, lai pārskatītu lapu saturu programmā (skatīt 12. attēls).

12. attēls **Navigācijas paneļa struktūrskice parastajā režīmā [Autora veidots attēls]**

Tas sevī iekļauj tādas pogas kā pogu navigācijas paneļa samazināšanai ikonu režīmā (augšā pa kreisi) un pogas, kas nomaina satura apgabalu uz attiecīgo lapu pogas nosaukumā. Ikonu režīms ir navigācijas paneļa mazākais iespējamais platums, ko lietotājs var uzstādīt, attiecīgi šajā režīmā navigācijas pogas vairs nerāda tekstu, tikai savas ikonas (skatīt 13. attēls).



13. attēls **Navigācijas paneļa struktūrskice ikonu režīmā [Autora veidots attēls]**

Ikonu režīms pāriet uz parasto režīmu (pogas rāda tekstu) kad lietotājs uzspiež augšējo bultas pogu, kas palielina navigācijas platumu, vai velkot navigācijas labo malu uz labo pusi, kas arī palielina navigācijas platumu.

Navigācijas panelī pieejamas šādas galvenās lapas:

- "*All Songs*" - galvenā bibliotēka (dziesmas, izpildītāji, albumi, mapes);
- "*Playlists*" - lietotāja atskaņošanas saraksti;
- "*Favorites*" - favorītu ieraksti;
- "*Recently Played*" - nesen atskaņotie ieraksti;
- "*User Guide*" - iebūvēta lietošanas pamācība;
- "*Settings*" - iestatījumi un bibliotēkas uzturēšana.

Vienlaikus var būt aktīva tikai viena navigācijas poga, pārējās darbojas kā izslēgtas, lai lietotājs vienmēr redzētu, kurš skats ir atvērts.

3.5.4. Mūzikas vadības elementu panelis

Visi audio vadības elementi atrodas uz atsevišķi paneļa (skatīt 14. attēls). Vadības panelī galvenokārt tiek rādīta atskaņošanas pozīcija un ilgums. Pilnāks dziesmas nosaukums un izpildītājs ir redzams atvērtā atskaņošanas rindas panelī ("*Now Playing"*) un lielajā atskaņošanas skatā (skatīt 3.5.5.4 un 3.5.5.5).

14. attēls **Mūzikas vadības paneļa struktūrskice [Autora veidots attēls]**

Tā augšā atrodas laika līnija, caur kuru lietotājs spēj mainīt atskaņošanas pozīciju šobrīd atskaņojošam singlam. Lietotājs var uzklikšķināt uz laika joslas, lai pārietu uz izvēlēto vietu pašreizējā ierakstā, ne tikai vilkt slīdni. Pa kreisi no laika līnijas atrodas taimeris, kas rāda šobrīdējo pozīciju skalā, pa labi taimeris rāda attiecīgā mūzikas ieraksta ilgumu.

Zem laika līnijas, sākot no kreisās, atrodas šādas ikonu pogas:

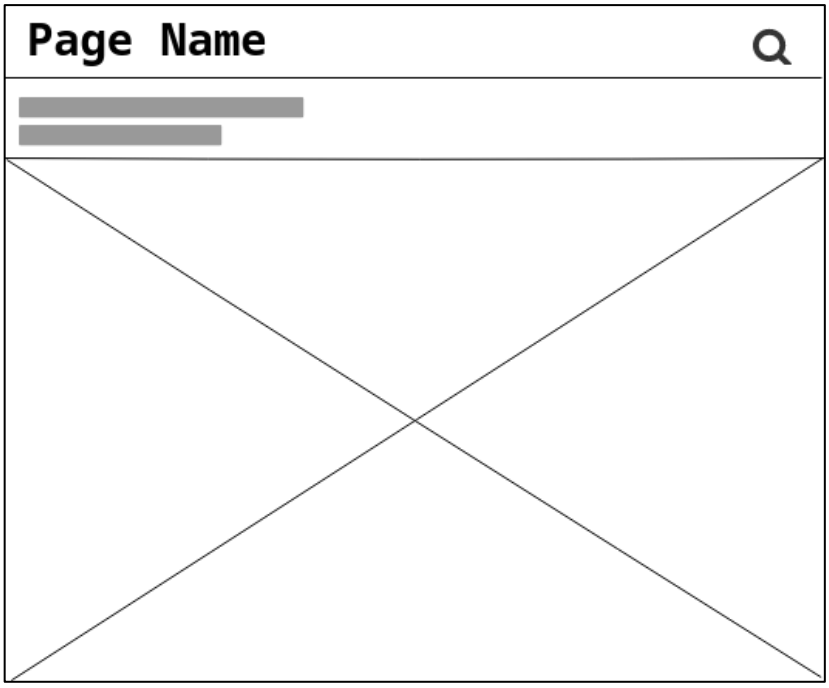
- lielā atskaņošanas skata poga (ikona var būt disks, kas rotē spēlēšanas laikā, vai cita ikona, kas labāk parāda pogas nozīmi). Poga atver lielo atskaņošanas skatu;
- skaļuma poga, tā ikona mainās atkarībā no blakus esošās skalas vērtības. To uzspiežot tiek izslēgta skaņa (ikonai attiecīgi jāpielāgojas skaļuma maiņas darbībām);
- skaļuma skala, tās vērtība var iet no 0% līdz 125%, kad skaļums ir pie 100% un pāri ir jāizceļ skalas izskats. Blakus skalai var atrasties procentu rādītājs, kas parāda šobrīdējo skaļuma vērtību;
- atskaņošanas režīmu poga. Spiežot to, tā maina savu ikonu attiecīgi režīmam, kas cikliski izvēlēts;
- iet uz iepriekšējo poga. Tā nogādā lietotāju uz iepriekšējo ierakstu rindā, tās ikonai jābūt pašsaprotamai;
- spēlēt/pauzēt poga. Tās ikona mainās attiecīgi, kad atskaņotājā tiek spēlēta mūzika un kad tā netiek. Ikonai jābūt pašsaprotamai;
- iet uz nākamo poga. Tā nogādā lietotāju uz nākamo ierakstu rindā, tās ikonai ir jābūt pašsaprotamai;
- patīk poga. Poga atzīmē ierakstu kā ‘patīk’ to uzspiežot, tāpēc ikonai attiecīgi tā izmaiņa ir jāparāda;
- atvērt atskaņošanas rindas (*queue*) poga. Poga atver atskaņošanas rindas paneli.

Lielākā daļa vadības darbību (atskaņot/pauzēt, nākamais/iepriekšējais, rinda, favorīts u.c.) ir pieejama arī ar tastatūru. Konfigurācija aprakstīta iestatījumos un funkcionālajā prasībā 3.4.2.10.

Visu pogu darbībām, kas neatver acīm redzamu skatu vai ievērojamu darbību, ir attiecīgi jāizsauc karoga paziņojums virs atskaņošanas vadības paneļa un jāinformē par veikto darbību.

3.5.5. Satura apgabala struktūra

Satura panelis ietekmējas no navigācijā un audio vadības paneļa veiktajām darbībām, mainot savu saturu uz attiecīgām lapām vai atjaunojot attiecīgus sarakstus vai to rindas pēc lietotāja darbībām. Gandrīz visām lapām satura apgabalā piemīt vienojošs izkārtojums, kas redzams struktūrskicē (skatīt 15. attēls).



15. attēls **Satura lapas vispārīga izkārtojuma struktūrskice [Autora veidots attēls]**

Visas lapas un apakšlapas seko vienādam izkārtojuma principam:

- Lapas augša atrodas galvene ar lapas nosaukumu, informējot lietotāju, kur atrodas;
- Kopā ar lapas nosaukumu, otrā pusē no tā, var atrasties arī meklēšanas poga priekš dziesmu, albumu, u.c. elementu meklēšanas. Šādai funkcijai nevajadzētu atrasties lapās, kur tas nav aktuāli (piem., iestatījumos).
- Zem galvenes atrodas papildinājumu sekcija, kas nav obligāta lapās, kur tās papildinājums nav vajadzīgs. Sekcijā atrodas:
    - apakšlapu cilnes;
    - informācija par autoru, albumu, singlu, u.c. skaitu attiecīgā lapā;
    - darbības pogas, kas veic kaut ko specifisku lapā, piemēram, pievieno attiecīgā saraksta visas dziesmas atskaņošanas rindai (*queue*) vai lapas iekšējā meklēšana.

- Zem augšējiem elementiem atrodas lapas saturs, tas visbiežāk iekļauj sarakstus ar dziesmām vai grupām (albumi, atskaņošanas saraksti, u.tml.), kas tiek dinamiski papildināti no datubāzes. Saturam arī var būt statisks izkārtojums, kur tas ir aktuāls, piemēram, lietotāja pamācībai un iestatījumiem.

Logu vienotais stils atvieglina gan izstrādi, gan veido skaidru programmas identitāti. Lapās, kas nāk no navigācijas paneļa darbībām, mainīgākā daļa ir papildinājuma sekcija, kur, kā jau iepriekš aprakstīts, ir ievērojama variācija. To variācija starp programmā nepieciešamajām lapām ir aprakstīta sadaļā 3.5.5.1 Variācija lapu galvenēs.

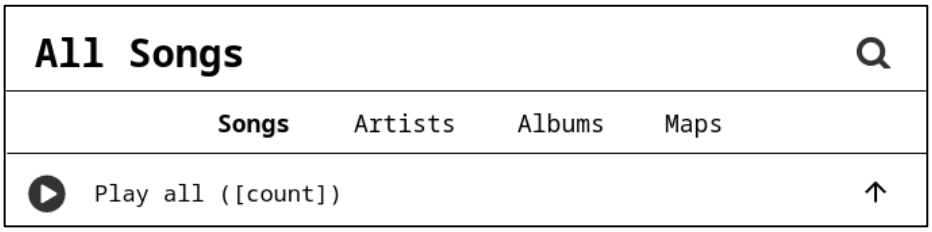
No visām satura lapām, kas nāk no navigācijas paneļa darbībām, visatšķirīgākie izkārtojumi ir lietotāja pamācības lapai un iestatījumu lapai (skatīt 3.5.5.2 un 3.5.5.3 sadaļas). Abu lapu saturs, atšķirībā no pārējām lapām, nav dinamiski ielasīts no datubāzes.

Satura apgabalā arī parādās struktūras, kas pārklāj citas lapas. Šīs struktūras parasti tiek izsauktas no darbībām iekšā satura apgabalā vai no mūzikas vadības paneļa (skatīt no 3.5.5.4 līdz 3.5.5.6 sadaļai).

Bibliotēkas sarakstos (dziesmas, grupas u.c.) plānota labās peles taustiņa atverama konteksta izvēlne ar biežākajām darbībām, piemēram: atskaņot, pievienot atskaņošanas rindai, atzīmēt kā favorītu, noņemt no rindas vai pārskatīt saistīto saturu. Tas atbilst ierastai darbvirsmas mūzikas programmu pieredzei un papildina pogas galvenē un vadības panelī, nepārslogojot galveno izkārtojumu.

3.5.5.1. Variācija lapu galvenēs

Galvenēm ir galvenās variācijas (skatīt no 16. attēls līdz 18. attēls).



16. attēls **"*All Songs*" lapas galvenes skice [Autora veidots attēls]**

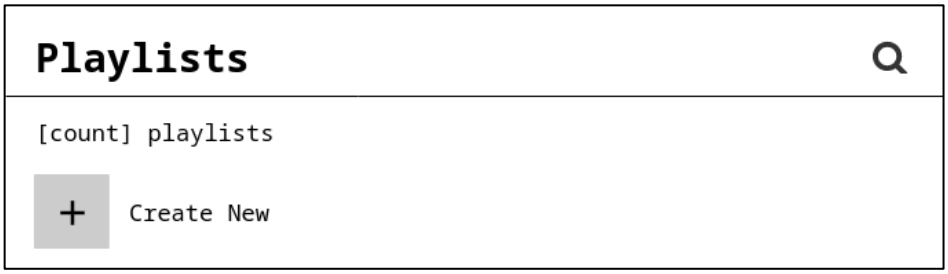
Augšējā galvenes skice ir tikai priekš "*All Songs*" lapas – galvenā bibliotēkas lapa. Tās augšā atrodas tā nosaukums, šķērsus no tā - meklēšanas poga, zem tiem atrodas ciļņu izvēlne starp apakšlapām: dziesmas, izpildītāji, albumi un mapes (direktorijas) un visbeidzot zem tiem atrodas 'spēlēt visus' poga ar tekstu, kas to nosauc un pasaka cik ierakstu tiks pievienoti atskaņošanas rindai, šķērsus tam atrodas poga, kas ļauj ātri uzskrullēt uz lapas augšu. Apakšlapas arī dala šo izskatu, vienīgais, kas mainās ir ieraksti, kas ielasās lapas saturā. Ja

bibliotēkā vēl nav ierakstu, "*All Songs"* lapā tiek parādīts norādījums ar saiti, lai pievienotu mūzikas direktoriju (parasti caur *Settings* → *File Options*), tādējādi atvieglojot pirmo lietošanu.

Pastāv arī vienkāršāka galvene, kas balstās uz "*All Songs*" galveni. Vienkāršotā galvene (skatīt 17. attēls) tiek izmantota albumu un izpildītāju skatos, kā arī lapās *Favorites* un *Recently Played*, specifiski, kur saturs ir viens filtrēts ierakstu saraksts bez apakšcilnēm.

17. attēls **Vienkāršota galvenes skice [Autora veidots attēls]**

"*All Songs*" lapa nav vienīgā ar speciālu galveni, arī atskaņošanas sarakstu lapai ir savs galvenes izskats (skatīt 18. attēls).



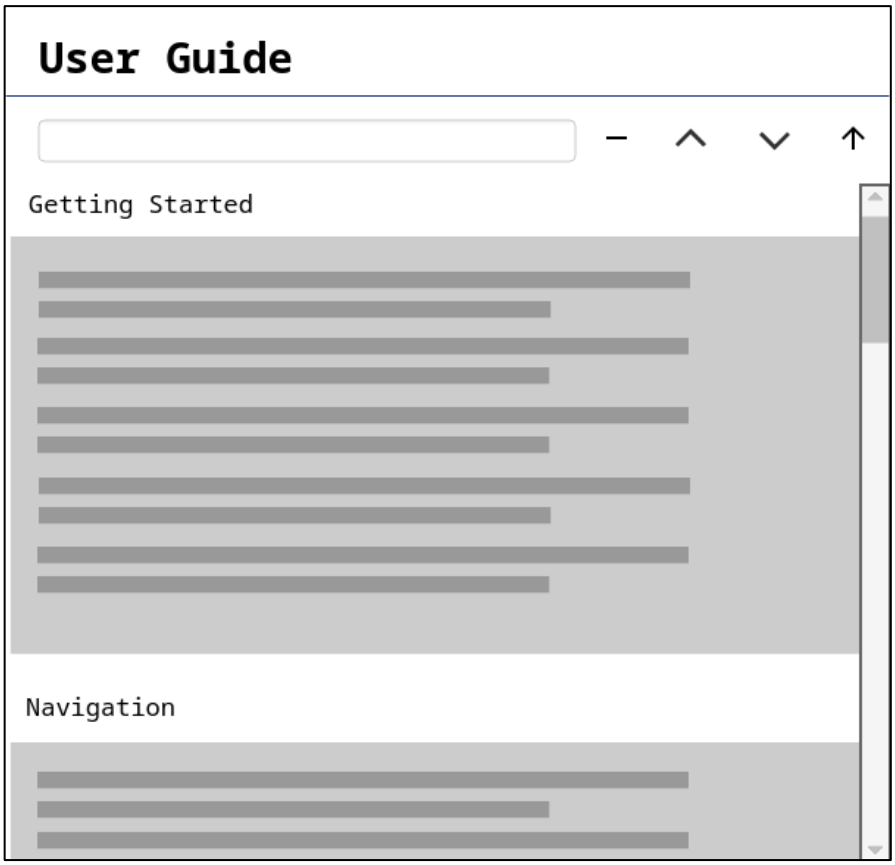
18. attēls **"*Playlist*" lapas galvenes skice [Autora veidots attēls]**

Galvene pievieno sev unikālu pogu – izveidot jaunu. Poga ir novietota līdzīgi kā 'spēlēt visus' poga. Tātad ir divi līmeņi: *Playlists* lapā ir saraksts ar visiem lietotāja veidotiem sarakstiem (18. attēls, galvenē iespēja izveidot jaunu sarakstu) un atvērts viena saraksta skats ar vienkāršoto galveni (17. attēls) un attiecīgā saraksta dziesmām.

Pamācības lapas un iestatījumu lapas galvenes ir atsevišķi aprakstītas sadaļās 3.5.5.2 un 3.5.5.3.

3.5.5.2. Lietotāja pamācības lapa

Lietotāja pamācības lapa izmanto vienkāršotu satura izkārtojumu (skatīt 19. attēls).



19. attēls **Lietotāja pamācības lapas struktūrskice [Autora veidots attēls]**

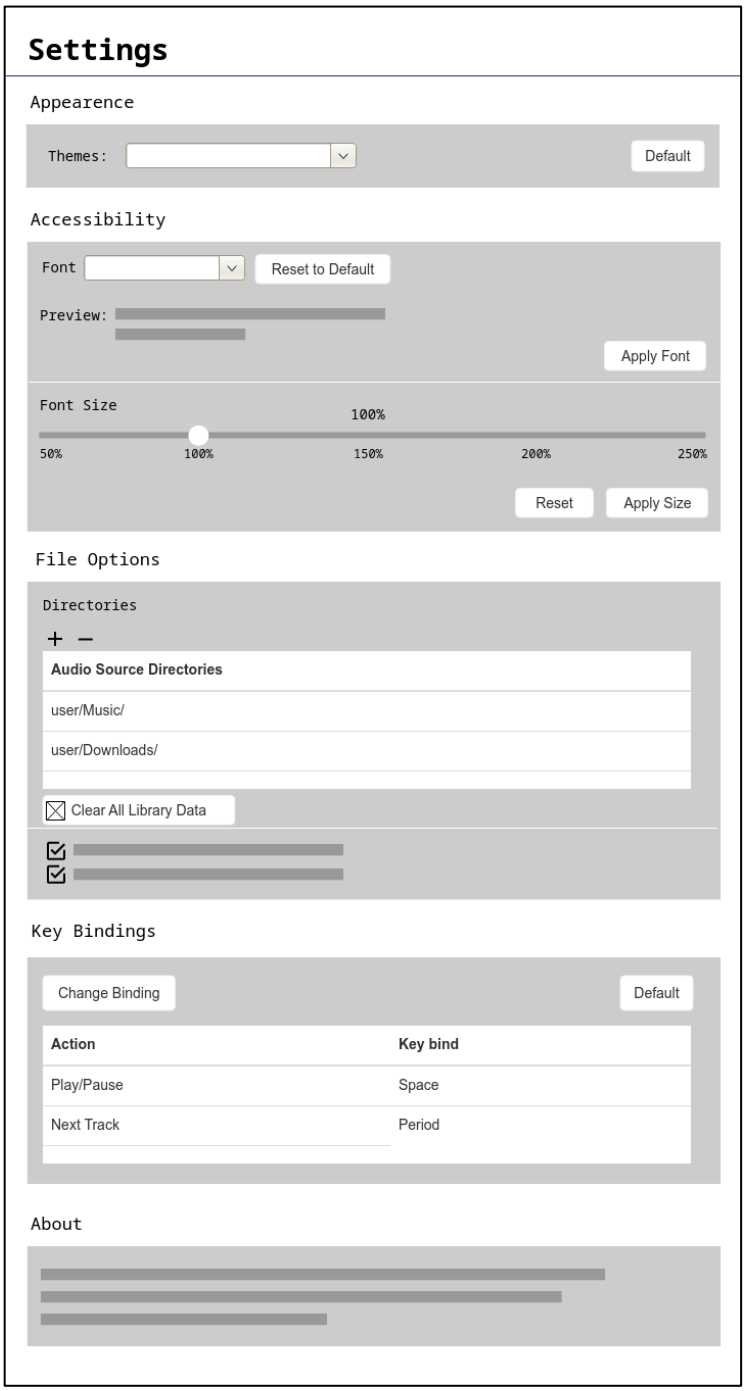
Galvenē atrodas tikai lapas nosaukums bez papildus pogām. Tūlīt zem tā, joslā atrodas pilna platuma meklēšanas lauks, rezultātu skaitītājs, iepriekšējās un nākamās rezultāta pogas, kā arī vadības poga „Pārvietoties uz augšu". Galvenā zona ir vertikāli pārvietojama palīdzības tēmu kolonna. Tēmas ir sagrupētas sadaļu virsrakstos, katra tēma parādās kā atsevišķa kartīte ar treknrakstā izcelto virsrakstu un paskaidrojošo tekstu. Lapas iekšējā meklēšana filtrē, kuras kartes tiek parādītas, un izceļ meklējamo tekstu un tekstā, lietotājs var pārvietoties pa rezultātiem ar blakus esošajām vadības pogām, aktīvais rezultāts tiek izcelts un parādīts skatā. Ja nekas nesakrīt, ritināšanas saraksts tiek aizstāts ar īsu ziņojumu par tukšo stāvokli.

Pamācības saturs ir angļu valodā, saskaņā ar vispārējo saskarnes valodu. Meklēšanas laikā lietotājs var ar tastatūru pārvietoties starp rezultātiem, izmantojot blakus esošās pogas vai atbilstošos karstos taustiņus.

3.5.5.3. Iestatījumu lapa

Iestatījumu lapa ir centrālā vieta lietotnes konfigurācijai. Tā atveras no navigācijas paneļa, izmantojot pogu "*Settings*", un satur visas galvenās preferences vienuviet. Atšķirībā no vairākām bibliotēkas lapām, šī lapa izmanto vertikālu ritināšanu visā satura apgabalā: satura

augstums atbilst visu sadaļu kopējai summai, un lietotājs ritina uz leju, lai sasniegtu apakšējās sadaļas (skatīt 20. attēls).



20. attēls **Iestatījum lapas pilna struktūrskice [Autora veidots attēls]**

Lapas galvenē atrodas tikai virsraksts "*Settings*" bez papildu darbību pogām. Zem virsraksta ir horizontāla atdalošā līnija, kas vizuāli atšķir galveni no konfigurācijas satura.

Zem galvenes atrodas vertikāli sakārtots satura bloks. Tajā secīgi ir piecas iestatījumu sadaļas, katra ar savu virsrakstu un satura konteineru. Sadaļas ir šādas:

- Izskats (*Appearance*) - krāsu motīva izvēle. Rindā ir etiķete "*Theme*:" priekš nolaižamā saraksta ar pieejamajiem motīviem un poga "*Default*", kas atgriež noklusējuma motīvu.
- Pieejamība (*Accessibility*) - teksta noformējums. Augšējā daļā: fonta ģimenes izvēle, pogas "*Reset to Default*" un "*Apply Font*", kā arī priekšskatījuma zona ar parauga tekstu vairākās rakstībās. Zem atdalošās līnijas – "*Font Size*" ar procentuālo vērtību, slīdni un pogām "*Reset*" un "*Apply Size*".
- Failu opcijas (*File Options*) - bibliotēkas avoti un uzturēšana. Sadaļā "*Directories*" ir rīkjosla ar pogām "pievienot" (+) un "noņemt" (−), kā arī tabula ar kolonnu "Mūzikas avota direktorijas" (*Audio source directories*), kur redzami pievienotās direktorijas. Zem tabulas atrodas poga "*Clear All Library Data*", kas dzēš bibliotēkas datus datubāzē. Pēc tā, divas izvēles rūtiņas: "*Check for library changes on startup*" (noklusējums ieslēgts) un "*Apply library changes on startup without asking*" (kļūst pieejama, ja pirmā rūtiņa ir aktīva).
- Karstie taustiņi (*Key Bindings*) - tastatūras saīsinājumi. Augšā ir pogas "*Change Binding*" un "*Default*", zem tām tabula ar kolonnām "*Action*" un "*Key Bind*", kur lietotājs redz un var mainīt saistības starp darbību un taustiņu.
- Par (*About*) - informatīvs bloks ar īsu prototipa aprakstu, autora norādi un hipersaiti uz projekta repozitoriju.

3.5.5.4. Atskaņošanas rindu (*queue*) panelis

Atskaņošanas rindas panelis ir pārklājuma elements, kas parādās virs satura apgabala un mūzikas vadības paneļa, nepārklājot galveno navigācijas lapu. Panelis izslīd no apakšas ar animāciju un sākotnēji aizņem aptuveni pusi satura apgabala augstuma (skatīt 21. attēls).

21. attēls **Atskaņošanas rindas struktūrskice [Autora veidots attēls]**

Panelis augšpusē satur horizontālu "vilkšanas" joslu, ar kuru lietotājs var mainīt paneļa augstumu (līdzīgi kā navigācijas panelim). Ja pēc atlaišanas augstums kļūst mazāks par noteiktu slieksni, panelis automātiski tiek aizvērts.

Zem vilkšanas joslas atrodas galvene ar rīku rindu. No kreisās puses tur ir:

- poga "Aizvērt" - paslēpj paneli;
- virsraksts ar rindas informāciju;
- poga atskaņošanas režīma maiņai (secīgi / jauktā secībā / atkārtot sarakstu / atkārtot vienu ierakstu);
- poga rindas tīrīšanai (atkritumu tvertne).

Zem rīku rindas ir horizontāla atdalošā līnija.

Tagad atskaņo zonā ir īss apakšvirsraksts "*Now Playing*", zem tā - pašlaik atskaņotā ieraksta nosaukums un izpildītājs. Ja nekas neatskaņojas, tiek rādīts atbilstošs noklusējuma teksts.

"Turpinājumā" saraksts: otra atdalošā līnija un apakšvirsraksts "*Up Next*". Galvenā zona ir vertikāli ritināms saraksts ar rindā esošajiem ierakstiem. Katrai rindai ir:

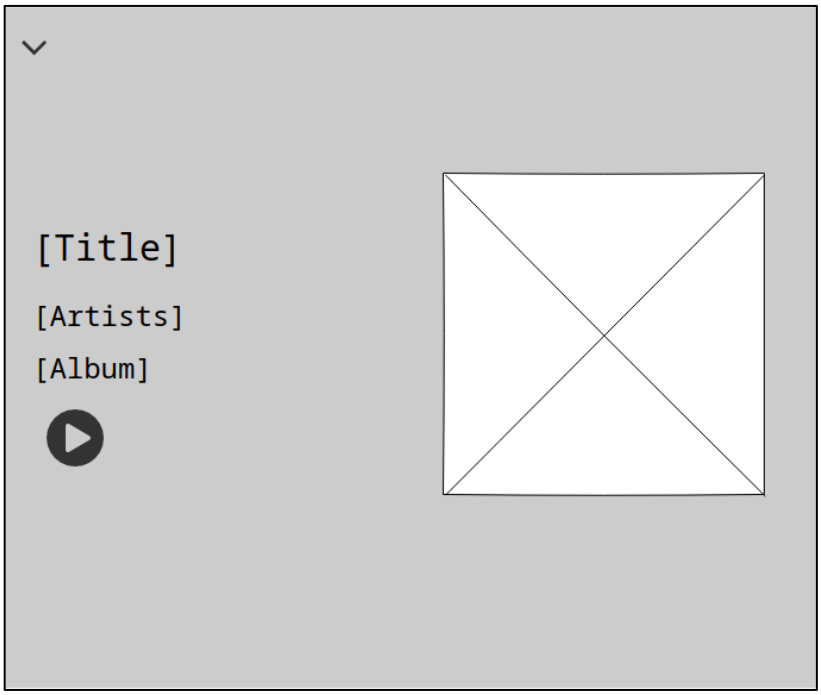
- vilkšanas turis (⠿) secības maiņai;
- dziesmas nosaukums un izpildītājs;
- ilgums;

- poga ieraksta noņemšanai no rindas.

Pašlaik atskaņotais ieraksts sarakstā tiek vizuāli izcelts. Ja rinda ir tukša, tiek parādīts ziņojums "*Queue is empty*".

3.5.5.5. Lielais atskaņošanas skats

Lielais atskaņošanas skats (*Big Player*) ir pilna platuma pārklājuma skats, kas parāda pašlaik atskaņotā ieraksta informāciju lielākā, fokusētā formātā. Skats aizņem satura apgabalu virs bibliotēkas lapām, bet nepārklāj navigācijas paneli un atskaņošanas vadības paneli (skatīt 22. attēls).



22. attēls **Lielā atskaņošanas skata struktūrskice [Autora veidots attēls]**

Augšējā kreisajā stūrī ir viena poga ar ikonu uz leju, kas aizver lielo skatu un atgriež lietotāju iepriekšējā satura skatā.

Galvenajā zonā atrodas horizontāli sadalīts saturs:

- Kreisā puse - metadati un vadība: dziesmas nosaukums, izpildītājs, albums, kā arī liela atskaņot/pauzēt poga ar atbilstošu ikonu.
- Labā puse - albuma vāka attēlu.

Informācija un vāks sinhronizējas ar aktuālo atskaņošanas stāvokli. Atskaņošanas laikā skata atvēršanas poga atskaņošanas vadības panelī var spēlēt rotācijas animāciju vai ko līdzīgu, lai vizuāli norādītu aktivitāti.

3.5.5.6. Meklēšanas lapa

Meklēšanas lapa ir pārklājuma skats, kas atveras virs pašreizēji atvērtās lapas. To izsauc ar meklēšanas ikonu lapas galvenē vai no citām atbilstošām vietām programmā. Aizvēršana notiek ar pogu atpakaļ galvenes kreisajā pusē, kas noņem pārklājumu un atjauno iepriekšējo saturu (skatīt 23. attēls).

23. attēls **Meklēšanas lapas struktūrskice [Autora veidots attēls]**

Lapas augšdaļā horizontāli izvietoti:

- poga atpakaļ (aizver meklēšanu);
- centrā - meklēšanas ikona, teksta lauks ar aizpildītāju "*Search songs, artists, albums…*" un poga notīrīt (x), kas iztukšo vaicājumu.

Zem galvenes ir atdalošā līnija, kur atrodas meklēšanas rezultāti, kas ir vertikāli ritināms saraksts. Katrs rezultāts tiek attēlots atbilstošā rindas veidā (dziesma vai grupas elements). Meklējamais teksts rezultātos tiek izcelts. Dziesmu rindās pieejamas tās pašās papildus darbības kā bibliotēkā. Ja meklēšanas lauks ir tukšs, rezultātu saraksts tiek notīrīts. Ja pēc meklēšanas nekas neatbilst, tad saraksts arī paliek tukšs.

3.5.6. Vizualizācija un tēmu sistēma

Vizuālajām prasībām jānodrošina vienots un pārskatāms izskats visā programmā, vienlaikus ļaujot lietotājam pielāgot izskatu pēc vajadzības.

Plānoti vismaz pieci alternatīvi krāsu motīvi (piem., tumši un gaiši varianti), ko lietotājs izvēlas iestatījumos (*Appearance → Theme*). Motīva maiņa jāpiemēro uzreiz visā atvērtajā saskarnē un jāsaglabā nākamajām lietošanas reizēm. Jābūt iespējai atgriezties pie noklusējuma motīva vienas pogas darbībā.

Vizualizācija balstās uz centralizētu stilu failu pieeju: kopīga bāzes struktūra un atsevišķi motīvu faili ar vienādiem stila mainīgajiem (krāsām, foniem, izcelšanai u.c.). Tas ļauj mainīt paleti, nemainot katras lapas izkārtojumu.

Teksta fonts un izmērs tiek pielāgoti atsevišķi (sadaļā *Accessibility*), ar priekšskatījumu pirms saglabāšanas. Fonta un izmēra izmaiņām jābūt pamanāmām galvenajās zonās: navigācija, saraksti, galvenes, iestatījumi un palīdzības teksts. Izmaiņas nedrīkst padarīt tekstu nelasāmu vai pārklāt citus elementus.

Svarīgas darbības (piem., favorīts, atskaņošanas režīms, kļūdas) jāatspoguļo ar ikonu atjaunošanos vai vizuālu izcelšanu, lai lietotājs saprastu sistēmas stāvokli bez papildu tehniskās informācijas.

3.5.7. Veiktspēja

Veiktspējas prasības nosaka, ka prototipam jābūt lietojamam ikdienā tipiskā personiskā datorā. Sistēmai jāspēj darboties vidē ar vismaz 4 GB RAM (skatīt 2.4 sadaļu), ieteicams, lai palaišanas un bibliotēkas ielādes brīdī būtu pietiekami brīvas atmiņas.

Mapju pievienošana un pārskenēšana var aizņemt laiku atkarībā no failu skaita un diska ātruma, procesam jābūt progresa indikācijai un iespējai atcelt ilgstošu skenēšanu. Pēc skenēšanas saraksti jāatjaunina bez manuālas programmas pārstartēšanas.

Meklēšanai pār bibliotēku jāatgriež rezultāti bez pārlieku lielas gaidīšanas. Meklēšanas lauka ievadei plānota īsa aizkave pēc rakstīšanas, lai samazinātu lieku slodzi, kamēr lietotājs vēl raksta vaicājumu.

Pāreja starp dziesmām, pauze un turpināšana jāuztver kā tūlītēja lietotāja darbība; ilgākas aizkaves pieļaujamas tikai faila ielādē vai neatbalstīta formāta gadījumā, ar saprotamu paziņojumu.

Bibliotēkas metadati un vāku attēli tiek glabāti lokāli (*SQLite*). Lielas kolekcijas datubāzē un diska ievades/izvades ātrums ietekmē ielādes laiku, tas ir plānots ierobežojums.

3.5.8. Uzturamība un paplašināmība

Šīs prasības nodrošina, ka prototipu var turpināt attīstīt un pielāgot klienta vēlamajām funkcijām, nesākot projektu no jauna.

Sistēma plānota ar atdalītu prezentācijas un darbības slāni (skatīt 3.3.2 sadaļu):

- saskarne (FXML, kontrolieri);
- biznesa loģika (servisi, DAO, *SQLite*).

Tas atvieglo kļūdu labošanu, jaunu logu pievienošanu un testēšanu.

Galvenais logs ir sadalīts nemainīgās zonās (navigācija, saturs, vadības panelis). Atsevišķas lapas un iestatījumu sadaļas veidotas kā atkārtojami bloki (piem., iestatījumu sekcijas, pārklājumi). Jaunas bibliotēkas skata variācijas var pievienot, ievērojot kopējo lapas struktūru (skatīt 3.5.5 sadaļu).

Pašas bibliotēkas dati tiek glabāti strukturētā *SQLite* shēmā (skatīt 3.6 sadaļu), skaidri sadalot ierakstus no metadatiem, atskaņošanas sarakstiem, rindu un atskaņotāja stāvokli. Tas

atbalsta jaunu filtru, meklēšanas lauku un papildu attiecību veidošanu bez pilnīgas arhitektūras maiņas.

Arhitektūrai jāļauj nākotnē paplašināt šādas funkcijas:

- papildu audio formātus;
- metadatu rediģēšanu;
- paziņojumus par pazudušiem failiem;
- plašāku OS atbalstu un automātiskāku bibliotēkas uztveršanu.

Pat ja šīs funkcijas prototipa obligātajā apjomā nav pilnībā realizētas, to pievienošanai nedrīkst būt pretrunā ar esošo datu un UI modeli.

Lietotāja izvēles (motīvs, fonts, karstie taustiņi, daļa bibliotēkas uzvedības) glabājas lokālās konfigurācijas formātā, lai tās var mainīt bez programmas pārbūvēšanas. Tas atbilst prasībai par iestatījumu saglabāšanu (skatīt 3.4.2.11 sadaļu).

3.5.9. Uzticamībā

Uzticamības prasības nosaka, cik stabili un prognozējami prototipam jādarbojas ikdienas lietošanā, īpaši kā lokālai, bezsaistes mūzikas bibliotēkai.

Programmai jāspēj pilnvērtīgi darboties bez interneta pieslēguma pēc instalācijas un bibliotēkas izveides. Nav nepieciešams serveris, konts vai tiešsaistes autentifikācija. Visi būtiskie dati (bibliotēka, iestatījumi, rinda, atskaņotāja stāvoklis) tiek glabāti lokāli lietotāja datorā.

Pēc programmas aizvēršanas un atkārtotas palaišanas jāsaglabājas:

- bibliotēkas indekss un metadati;
- lietotāja atskaņošanas saraksti un favorīti;
- atskaņošanas rinda un pēdējais atskaņotāja stāvoklis (ja iespējams);
- izvēlētais motīvs, fonti un karsto taustiņu konfigurācija.

Lietotājam nav jāveic atkārtota manuāla iestatīšana, ja vien viņš pats nemaina avotus vai neizdzēš datus.

Ja bibliotēka saistīts fails ir nepieejams, pārvietots vai bojāts, sistēmai jāreaģē saprotami (piem., paziņojums vai vizuāla indikācija). Neatbalstīti formāti vai atskaņošanas problēmas jākomunicē tā, lai lietotājs saprastu iemeslu un iespējamo risinājumu (piem., *Linux* vidē - *GStreamer* spraudņa nepieciešamība M4A/AAC formātiem).

Noņemot avota mapi vai atsevišķus ierakstus no bibliotēkas, datubāzē jāatspoguļo izmaiņas konsekventi (arī attiecībā uz rindu un atskaņošanu, ja ieraksts vairs nav pieejams). Pēc

avotu pārskenēšanas bibliotēkai jāatspoguļo jauni, mainīti un noņemti faili atbilstoši lietotāja izvēlei (manuāla pārlādēšana vai starta pārbaude).

Atskaņošanas vadība, navigācija un saglabātie iestatījumi nedrīkst nejauši "atcelties" bez lietotāja darbības. Darbības, kas nemaina redzamu stāvokli (piem., favorīts, režīma maiņa), jāapstiprina ar īsu, netraucējošu paziņojumu, kā aprakstīts 3.5.4 sadaļā.

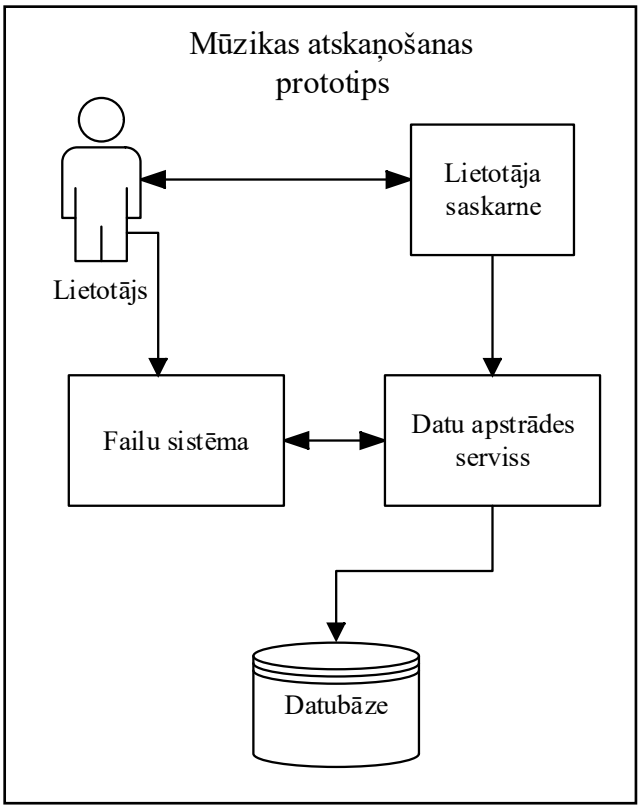
3.6. Datu struktūru plāns

3.6.1. Ievads

Šajā nodaļā tiek aprakstīta prototipa plānotā datu arhitektūra, to nozīme produktā kā arī nepieciešamie datu un to tipi. Prototips pielieto *SQLite* relācijas datubāzes tehnoloģiju, lai spētu veikt nepieciešamo datu organizāciju un apstrādi bez interneta pieslēguma nepieciešamības.

3.6.2. Datu plūsma produktā

Produkta datu plūsmas attēlošana palīdz izprast datu nepieciešamību programmas veiksmīgai darbībai un galvenos datu apstrādes procesus. Datu plūsma starp lietotāju, programmu un datubāzi ir attēlota diagrammā (skatīt 24. attēls).



24. attēls **Pirmā līmeņa konteksta diagramma [Autora veidots attēls]**

Lietotāja saskarne nosūta pieprasījumus darbības slāņa servisiem (bibliotēkas pārvaldība, atskaņošana, rindas pārvaldība), servisi izmanto DAO (*Data Access Object*) slāni, kas izpilda

SQL vaicājumus pret datubāzi. Inicializāciju veic datubāzes palīgs: izveido savienojumu, ieslēdz ārējās atslēgas un, ja datubāze ir jauna, izpilda shēmas izveidi, kas programmā atsevišķi definēta.

3.6.3. Datubāzes apraksts

Prototipa datu glabāšanai tiek izmantota *SQLite* relāciju datubāze – viens lokāls fails, piemēram, *music_library.db* lietotāja datu mapē (`~/.soufone`). Šī pieeja atbilst prasībai par darbību bezsaistē: nav nepieciešams atsevišķs datubāzes serveris, pieslēgšanās parametri vai tīkla savienojums lietošanas laikā.

Datubāzē tiek glabāts bibliotēkas indekss un lietotnes stāvoklis, nevis paši audio faili. Faktiskie ieraksti atrodas failu sistēmā, tabulā tiek saglabāts unikāls ceļš, ar kuru programma atver failu atskaņošanai. Metadati (izpildītājs, albums, ilgums, formāts u.c.) tiek iegūti importa/skenēšanas laikā un normalizēti attiecīgajās tabulās.

Kas glabājas datubāzē?

- Audio ierakstu metadati un ceļi. Izpildītājiem un albumiem (grupas tabulas) ir jāglabājas atsevišķās tabulās. Starp grupu tabulām atlases brīdī automātiski jāizpildās arī saistošās tabulas ar daudz-pret-daudziem saiti.
- Vāka attēli no audio failu metadatiem ir jātur atsevišķā tabulā, lai tos spētu sasaistīt ar citiem elementiem.
- Lietotāja veidoti atskaņošanas saraksti. Vienā tabulā uzglabā vairākus sarakstus, atsevišķā daudz-pret-daudziem tabulā savienojas ar pievienotiem singliem. Nosaukumi var atkārtoties.
- Aktīvā atskaņošanas rinda (queue) glabājas kā viena atsevišķa tabula. Tā regulāri atjaunojas caur lietotāja darbībām un manipulācijām. Uzturas starp sesijām, ja nav iztīrīta.
- Atskaņotāja stāvoklis starp sesijām. Glabā informāciju par pēdējo aktīvo dziesmu pirms programmas aizvēršanas, skaļuma skalas vērtību, atskaņošanas laika līnijas pozīciju. Tabulā vienmēr ir tika viena ieraksta rinda, kas atjaunojas ar pārmaiņām.
- Atskaņošanas vēsture. Katru reizi, kad atskaņota dziesma, glabā dziesmas informāciju. Atjauno tā paša ieraksta pozīciju par jaunāko, ja atkārti tiek spēlēts, nevis veido kopiju.

Kas netiek glabāts datubāzē?

- paši audio faili (tikai ceļš uz failu);

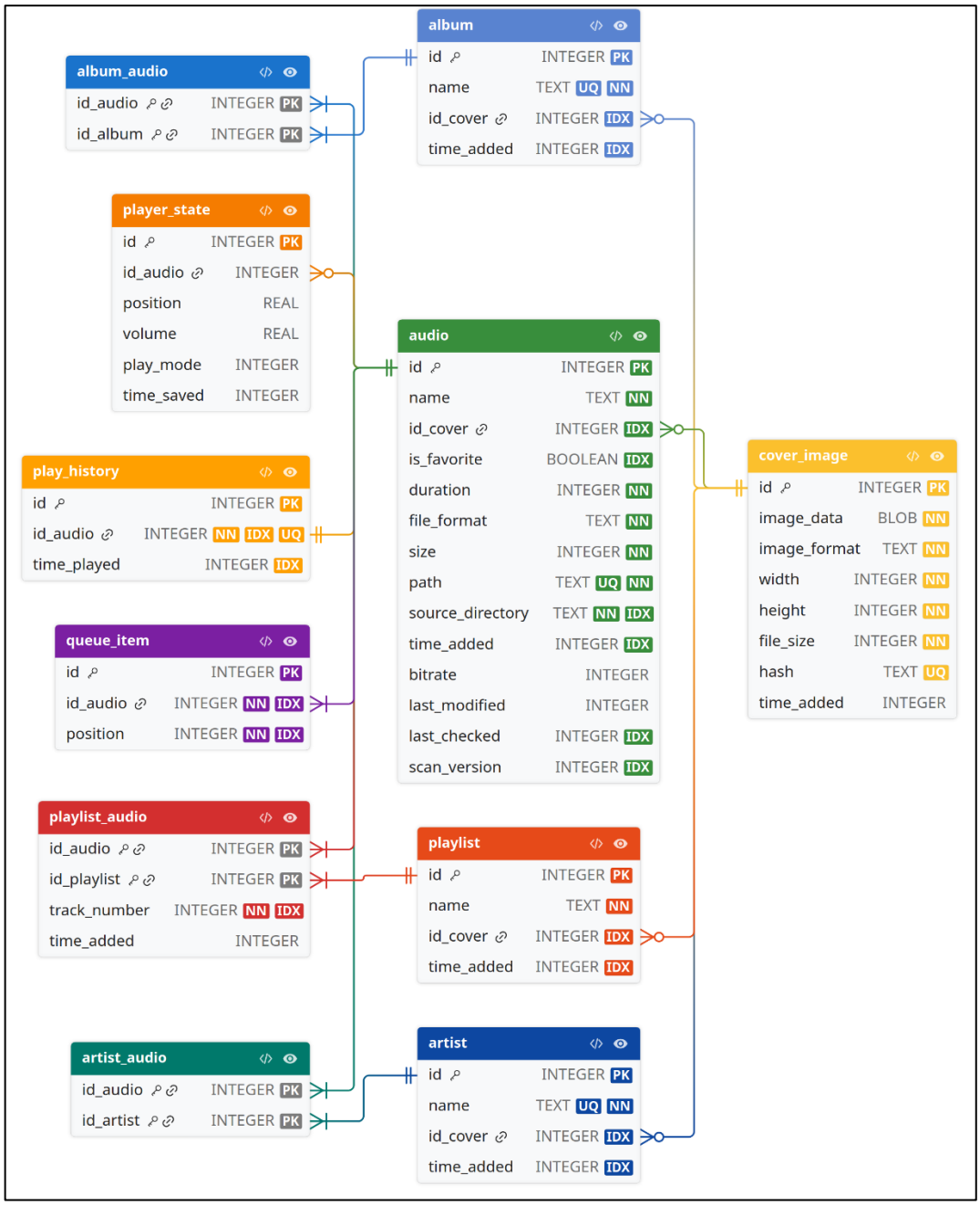
- saskarnes tēmas, fonti, karstie taustiņi un daļa vispārīgo iestatījumu (atsevišķi konfigurācijas faili lietotāja datu mapē);
- bibliotēkas avotu mapes kā atsevišķa tabula - avoti tiek atspoguļoti kā atšķirīgi ieraksti, lietotājs caur tiem var dzēst mūziku no bibliotēkas pēc avotu mapēm.

Svarīgi nejaukt divas atskaņošanas sarkstu koncepcijas:

- atskaņošanas rindas (*track queue*) - aktīvās atskaņošanas rindas, atrodas *queue* panelī (skatīt 3.5.5.4). Glabājas tabulā kā viens vesels saraksts, ko lietotājs var mainīt, bet nevar glabāt vairākas atskaņošanas rindas. Visa tabula ir viens saraksts;
- atskaņošanas saraksts (*playlist*) - saglabātie lietotāja saraksti, kurus var atkārtoti izmantot un organizēt. Glabā vairākus sarakstus.

3.6.3.1. ER modelis

Datubāzes struktūra ir vizuāli parādīta diagrammā (skatīt 25. attēls).



25. attēls **ER diagramma priekš *SQLite* datubāzes [Autora veidots attēls]**

Diagrammā redzamas visas 11 tabulas un to saites:

- "audio" ir centrālais elements, ar "artist" un "album" saistās caur "artist_audio" un "album_audio" (daudzi-pret-daudziem saite);
- "cover_image" ir neobligāts resurss vairākām entitījām (viens-pret-daudziem saites tabula ar vāku uz albumu, izpildītāju, audio, atskaņošanas sarakstu);
- "playlist" un "audio" – daudzi-pret-daudziem saite caur "playlist_audio" ar secību "track_number";

- “queue_item” un “play_history” ir tieši saistīti ar audio (viens-pret-daudziem saite);
- “player_state” satur ne vairāk kā vienu rindu un pēc vajadzības norāda uz vienu “audio” ierakstu.

Šī struktūra atbalsta 3.6.2 aprakstīto plūsmu: servisi strādā ar entitījām caur DAO, bet fiziskie faili paliek ārpus datubāzes, un *SQLite* nodrošina vienotu, lokālu un bezsaistes datu modeli.

3.6.3.2. Tabula “audio”

Tabulas “audio” specifiskie parametri ir redzami zemāk:

1. tabula **"audio" tabulas specifikācija [Autora veidota tabula]**

Nr.	Nosaukums	Tips	Noklusējums	Atribūti	Komentārs
1.	id	INTEGER	-	PRIMARY KEY, NOT NULL, AUTOINCREMENT	Unikāls ieraksta identifikators
2.	name	TEXT	-	NOT NULL	Dziesmas vai faila attēlojamais nosaukums.
3.	id_cover	INTEGER	NULL	FOREIGN KEY → cover_image(id), ON DELETE SET NULL	Neobligāta saite uz vāka attēlu
4.	is_favorite	BOOLEAN	FALSE	-	Vai ieraksts atzīmēts kā favorīts (*SQLite* glabā kā 0 vai 1)
5.	duration	INTEGER	-	NOT NULL	Ieraksta ilgums (sekundēs)
6.	file_format	TEXT	-	NOT NULL	Audio formāts (piem., mp3, flac, wav)
7.	size	INTEGER	-	NOT NULL	Faila izmērs baitos
8.	path	TEXT	-	NOT NULL, UNIQUE	Absolūtais ceļš uz failu sistēmā
9.	source_directory	TEXT	-	NOT NULL	Bibliotēkas avota mapes ceļš, no kuras fails importēts
10.	time_added	INTEGER	NULL	-	Laiks, kad ieraksts pievienots bibliotēkai (*Unix* laiks)

Nr.	Nosaukums	Tips	Noklusējums	Atribūti	Komentārs
11.	bitrate	INTEGER	NULL	-	*Bitrate* (kbps), ja pieejama no metadatiem
12.	last_modified	INTEGER	NULL	-	Faila pēdējās modificēšanas laiks (*Unix* laiks)
13.	last_checked	INTEGER	NULL	-	Pēdējā reize, kad ieraksts pārbaudīts bibliotēkas skenēšanā
14.	scan_version	INTEGER	0	-	Skenēšanas versija bibliotēkas atjaunināšanas kontrolei

3.6.3.3. Tabula “artist”

Tabulas “artist” specifiskie parametri ir redzami zemāk:

2. tabula **“artist” tabulas specifikācija [Autora veidota tabula]**

Nr.	Nosaukums	Tips	Noklusējums	Atribūti	Komentārs
1.	id	INTEGER	-	PRIMARY KEY, NOT NULL, AUTOINCREMENT	Unikāls ieraksta identifikators
2.	name	TEXT	-	NOT NULL	Izpildītāja nosaukums (unikāls visā tabulā)
3.	id_cover	INTEGER	NULL	FOREIGN KEY → cover_image(id), ON DELETE SET NULL	Neobligāta saite uz vāka attēlu
4.	time_added	INTEGER	`strftime('%s', 'now')`	-	Ieraksta pievienošanas laiks (*Unix* laiks)

3.6.3.4. Tabula “artist_audio”

Daudzi-pret-daudziem tabulas “artist_audio” specifiskie parametri ir redzami zemāk:

3. tabula **“artist_audio” tabulas specifikācija [Autora veidota tabula]**

Nr.	Nosaukums	Tips	Noklusējums	Atribūti	Komentārs
1.	id_audio	INTEGER	-	NOT NULL, PRIMARY KEY (kopā ar id_artist), FOREIGN KEY → audio(id), ON DELETE CASCADE	Saite uz audio ierakstu
2.	id_artist	INTEGER	-	NOT NULL, PRIMARY KEY (kopā ar id_audio), FOREIGN KEY → artist(id), ON DELETE CASCADE	Saite uz izpildītāju

3.6.3.5. Tabula “album”

Tabulas “album” specifiskie parametri ir redzami zemāk:

4. tabula **“album” tabulas specifikācija [Autora veidota tabula]**

Nr.	Nosaukums	Tips	Noklusējums	Atribūti	Komentārs
1.	id	INTEGER	-	PRIMARY KEY, NOT NULL, AUTOINCREMENT	Unikāls ieraksta identifikators
2.	name	TEXT	-	NOT NULL	Albuma nosaukums (unikāls, reģistram nejūtīgs)
3.	id_cover	INTEGER	NULL	FOREIGN KEY → cover_image(id), ON DELETE SET NULL	Neobligāta saite uz vāka attēlu
4.	time_added	INTEGER	`strftime('%s', 'now')`	-	Ieraksta pievienošanas laiks (*Unix* laiks)

3.6.3.6. Tabula “album_audio”

Daudzi-pret-daudziem tabulas “album_audio” specifiskie parametri ir redzami zemāk:

5. tabula **“album_audio” tabulas specifikācija [Autora veidota tabula]**

Nr.	Nosaukums	Tips	Noklusējums	Atribūti	Komentārs
1.	id_audio	INTEGER	-	NOT NULL, PRIMARY KEY (kopā ar id_album), FOREIGN KEY → audio(id), ON DELETE CASCADE	Saite uz audio ierakstu
2.	id_album	INTEGER	-	NOT NULL, PRIMARY KEY (kopā ar id_audio), FOREIGN KEY → album(id), ON DELETE CASCADE	Saite uz albumu

3.6.3.7. Tabula “playlist”

Tabulas “playlist” specifiskie parametri ir redzami zemāk:

6. tabula **“playlist” tabulas specifikācija [Autora veidota tabula]**

Nr.	Nosaukums	Tips	Noklusējums	Atribūti	Komentārs
1.	id	INTEGER	-	PRIMARY KEY, NOT NULL, AUTOINCREMENT	Unikāls ieraksta identifikators
2.	name	TEXT	-	NOT NULL	Atskaņošanas saraksta nosaukums
3.	id_cover	INTEGER	NULL	FOREIGN KEY → cover_image(id), ON DELETE SET NULL	Neobligāta saite uz vāka attēlu
4.	time_added	INTEGER	`strftime('%s', 'now')`	-	Ieraksta pievienošanas laiks (*Unix* laiks)

3.6.3.8. Tabula “playlist_audio”

Daudzi-pret-daudziem tabulas “playlist_audio” specifiskie parametri ir redzami zemāk:

7. tabula **“playlist_audio” tabulas specifikācija [Autora veidota tabula]**

Nr.	Nosaukums	Tips	Noklusējums	Atribūti	Komentārs
1.	id_audio	INTEGER	-	NOT NULL, PRIMARY KEY (kopā ar id_playlist), FOREIGN KEY → audio(id), ON DELETE CASCADE	Saite uz audio ierakstu
2.	id_playlist	INTEGER	-	NOT NULL, PRIMARY KEY (kopā ar id_audio), FOREIGN KEY → playlist(id), ON DELETE CASCADE	Saite uz atskaņošanas sarakstu
3.	track_number	INTEGER	-	NOT NULL	Ieraksta secības numurs sarakstā
4.	time_added	INTEGER	`strftime('%s', 'now')`	-	Ieraksta pievienošanas laiks (*Unix* laiks)

3.6.3.9. Tabula “cover_image”

Tabulas “cover_image” specifiskie parametri ir redzami zemāk:

8. tabula **"cover_image" tabulas specifikācija [Autora veidota tabula]**

Nr.	Nosaukums	Tips	Noklusējums	Atribūti	Komentārs
1.	id	INTEGER	-	PRIMARY KEY, NOT NULL, AUTOINCREMENT	Unikāls ieraksta identifikators
2.	image_data	BLOB	-	NOT NULL	Attēla binārie dati
3.	image_format	TEXT	-	NOT NULL	Attēla formāts (piem., jpeg, png)
4.	width	INTEGER	-	NOT NULL	Attēla platums pikseļos
5.	height	INTEGER	-	NOT NULL	Attēla augstums pikseļos
6.	file_size	INTEGER	-	NOT NULL	Attēla izmērs baitos
7.	hash	TEXT	NULL	UNIQUE	Jaucējvērtība dublikātu noteikšanai un atkārtotas glabāšanas samazināšanai
8.	time_added	INTEGER	`strftime('%s', 'now')`	-	Ieraksta pievienošanas laiks (*Unix* laiks)

3.6.3.10. Tabula “queue_item”

Tabulas “queue_item” specifiskie parametri ir redzami zemāk:

9. tabula **“queue_item” tabulas specifikācija [Autora veidota tabula]**

Nr.	Nosaukums	Tips	Noklusējums	Atribūti	Komentārs
1.	id	INTEGER	-	PRIMARY KEY, NOT NULL, AUTOINCREMENT	Unikāls ieraksta identifikators
2.	id_audio	INTEGER	-	NOT NULL, FOREIGN KEY → audio(id), ON DELETE CASCADE	Saite uz audio ierakstu aktīvajā rindā
3.	position	INTEGER	-	NOT NULL	Ieraksta secības numurs atskaņošanas rindā

3.6.3.11. Tabula “play_history”

Tabulas “play_history” specifiskie parametri ir redzami zemāk:

10. tabula **“play_history” tabulas specifikācija [Autora veidota tabula]**

Nr.	Nosaukums	Tips	Noklusējums	Atribūti	Komentārs
1.	id	INTEGER	-	PRIMARY KEY, NOT NULL, AUTOINCREMENT	Unikāls ieraksta identifikators
2.	id_audio	INTEGER	-	NOT NULL, FOREIGN KEY → audio(id), ON DELETE CASCADE, UNIQUE	Saite uz audio ierakstu atskaņošanas vēsturē
3.	time_played	INTEGER	`strftime('%s', 'now')`	-	Pēdējās atskaņošanas laiks (*Unix* laiks)

3.6.3.12. Tabula “player_state”

Tabulas “player_state” specifiskie parametri ir redzami zemāk:

11. tabula **"player_state" tabulas specifikācija [Autora veidota tabula]**

Nr.	Nosaukums	Tips	Noklusējums	Atribūti	Komentārs
1.	id	INTEGER	-	PRIMARY KEY, NOT NULL, CHECK (id = 1)	Vienīgais ieraksts tabulā (*singleton*)
2.	id_audio	INTEGER	NULL	FOREIGN KEY → audio(id), ON DELETE SET NULL	Pašlaik vai pēdējais atskaņotais ieraksts
3.	position	REAL	0	-	Atskaņošanas pozīcija ierakstā (sekundēs)
4.	volume	REAL	0,5	-	Skaļums (0,0 – 1,0)
5.	play_mode	INTEGER	0	-	Atskaņošanas režīms: 0 - secīgi, 1 - jauktā secībā, 2 - atkārtot sarakstu, 3 - atkārtot vienu ierakstu
6.	time_saved	INTEGER	`strftime('%s', 'now')`	-	Pēdējā stāvokļa saglabāšanas laiks (*Unix* laiks)

3.6.4. Struktūras ārpus datubāzes

Prototips papildus *SQLite* datubāzei izmanto vienotu lietotāja datu mapi (`~/.soufone` *Linux* vidē, `C:\Users\<lietotājvārds>\.soufone` *Windows* vidē), kur glabājas gan bibliotēkas fails, gan konfigurācija. Tas atbalsta bezsaistes darbību un iestatījumu saglabāšanu starp palaišanām bez centralizēta servera.

Galvenais fails *music_library.db* satur bibliotēkas indeksus, atskaņošanas stāvokli un atsauces uz attiecīgo audio failu un tā saistīto saturu (skatīt 3.6.3 sadaļu).

Iestatījumi pielieto vienkāršus teksta failus ar atslēga-vērtība pāriem priekš konfigurācijām:

12. tabula **Konfigurācijas faili [Autora veidota tabula]**

Fails	Mērķis	Saturs
`theme_config.properties`	Vizualizācija	Pašreizējais krāsu motīvs, izvēlētā fontu mape, fonta izmērs procentos
`hot_key_config.properties`	Karstie taustiņi	Darbību (piem., spēlēt/pauzēt) saistība ar tastatūras taustiņiem
`app_preferences.properties`	Lietotnes uzvedība	Bibliotēkas pārbaude startā, automātiska izmaiņu piemērošana

Šie faili tiek izveidoti ar noklusējuma vērtībām pirmajā palaišanas brīdī un atjaunināti, kad lietotājs maina iestatījumus (3.5.5.3). Tie glabā tikai lietotāja preferences.

Krāsu motīvu CSS un fontu faili nav lietotāja mapē – tie tiek iekļauti izplatāmajā pakotnē. `theme_config.properties` norāda tikai kuru motīvu un fontu lietotājs ir izvēlējies.

4. SISTĒMAS REALIZĀCIJA

4.1. Ievads

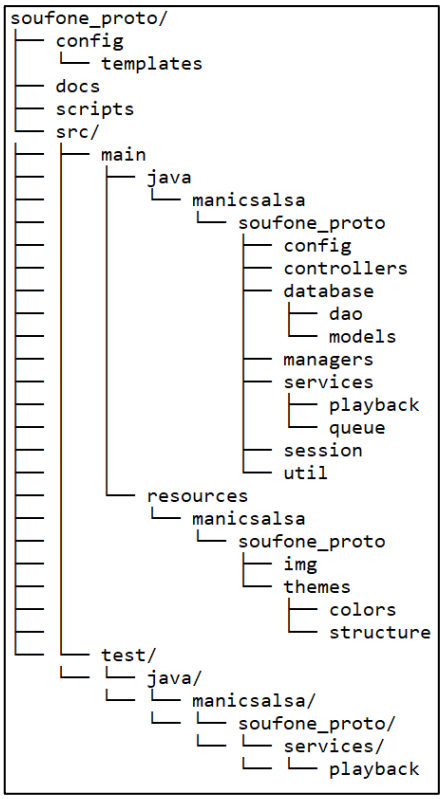
Trešā nodaļā tika definētas prototipa prasības: funkcionālās un nefunkcionālās prasības, lietojumgadījumi, plānotā lietotāja saskarne un datu struktūra. Šī nodaļa apraksta realizēto risinājumu - kā šīs prasības ir ieviestas programmatūrā, kādas tehnoloģijas un arhitektūras pieejas tika izmantotas, un kā sistēmas daļas sadarbojas praksē.

Nodaļas mērķis ir parādīt gatavo prototipu, kas realizē 3. nodaļā definētās prasības. Sadaļās tiek izklāstīts svarīgākais no izstrādes rezultāta - mazākas nozīmes implementācijas detaļas var tikt tikai īsi pieminētas šīs nodaļas saturā, bet netiks izklāstītas atsevišķās apakšnodaļās un netiks atkārtotas pielikumos. Pilnu projekta avota kodu ar *Javadoc* komentāriem var apskatīt projekta *GitLab* repozitorijā [11].

4.2. Projekta uzbūve

Projekta avota kods ir organizēts pēc *Maven* standarta projekta izkārtojuma, kas nodrošina vienu vietu atkarību pārvaldību, kompilāciju, testēšanu un izplatāmo pakotņu veidošanu. Šāda struktūra atbalsta 3. nodaļā aprakstīto trīsslāņu arhitektūru (skatīt 3.3.2 sadaļu): prezentācijas, darbības un datu slāņa kods ir fiziski atdalīts pakotnēs un resursos, bet palaišanas brīdī tiek salikts vienā *JavaFX* lietotnē.

Repozitorija augstākā līmeņa uzbūve ir attēlota struktūrskicē (skatīt 26. attēls).



26. attēls **Projekta repozitorijas struktūrskice [Autora veidots attēls]**

Galvenās mapes un to loma ir šāda:

- `src/main/java/` - visa lietojumprogrammas *Java* implementācija;
- `src/main/resources/` - FXML skati, CSS motīvi, fonti, noklusējuma attēli;
- `src/test/java/` - vienību testi (*JUnit* 5);
- `config/templates/` - konfigurācijas veidnes primārai palaišanai;
- `docs/` - Instalācijas un izstrādes instrukcijas;
- `scripts/` - palīgskripti (piem., *.AppImage* faila veidošanai).

Lietotāja dati (*SQLite* datubāze un konfigurācijas faili) netiek glabāti repozitorijā, tie tiek izveidoti lietotāja datu mapē `~/.soufone` (Linux) vai `C:\Users\<lietotājvārds>\.soufone (Windows)`, kā plānots 3.6.4 sadaļā.

4.2.1. *Maven* projekta definīcija

Projekta identifikators un atkarības ir definētas `pom.xml` failā (skatīt 1. pielikumu).

Failā definētās atkarības tieši atbalsta 3. nodaļas prasības:

13. tabula **Projekta atkarības [Autora veidota tabula]**

Nr.	Tehnoloģija (*Maven*)	Loma prototipā
1.	*JavaFX* 21 (darbības, *fxml*, mēdiji)	Saskarne un audio atskaņošana
2.	*SQLite JDBC, ORMLite, HikariCP*	Lokālā datubāze un DAO
3.	*jaudiotagger*	ID3 / metadatu nolasīšana
4.	*mp3spi, vorbisspi, jflac-codec*	Papildu formātu atbalsts kopā ar *JavaFX Media*
5.	*ikonli-**	Navigācijas un vadības ikonas
6.	*TwelveMonkeys ImageIO*	Vāku attēlu apstrāde
7.	*JUnit 5, Mockito*	Automatizētie testi

Būvniecības spraudņi:

- `maven-compiler-plugin` - *Java* avota kompilācija;
- `javafx-maven-plugin` - izstrādes palaišana ar *Launcher* kā ieejas punktu;
- `maven-shade-plugin` - vienots „*fat*" JAR izplatīšanai (*package* fāzē), manifestā norādot *Launcher*.

Profili izplatīšanai (atbilst 3.3.3 sistēmas prasībām *Windows* un *Linux*):

`installer-win` - *Windows* portatīvā *app-image (jpackage);*

`installer-win-msi` - *Windows .msi* instalētājs (nepieciešams *WiX* rīks);

`installer-linux` - *Linux app-image* mape;

`installer-linux-appimage` - *Linux .AppImage* (ar `scripts/build-appimage.sh`).

Detalizētās komandas un *GStreamer* pakotņu nosaukumi par izplatāmo pakotņu izveidi dokumentēti repozitorijas `docs` mapē [12].

4.2.2. Programmas palaišanas ķēde

Lietotnes starta secība realizē plānoto atdalījumu starp *JavaFX* moduļa ieraksta punktu un faktisko *Application* klasi:

Launcher → SoufoneApplication → MainLayout.fxml / MainLayoutController

- *Launcher* - minimāla klase, kas izsauc *SoufoneApplication*. Tā nepieciešama, jo *JavaFX* moduļu sistēmā *Application* apakšklase nedrīkst būt vienlaikus moduļa galvenā klase *shade/JAR* manifestā (skatīt 2. pielikumu).
- *SoufoneApplication* - *Application* apakšklase (skatīt 3. pielikumu), kas inicializē datubāzi, atjauno rindu un atskaņotāja stāvokli, ielādē galveno izkārtojumu, piemēro tēmu un parāda logu (noklusējuma izmērs 1200×800px).
- *MainLayoutController* - saista navigācijas paneli, satura zonu un mūzikas vadības paneli (skatīt 3.5.2 - 3.5.4 sadaļas), pārvalda lapu un pārklājumu maiņu (skatīt 4. pielikumu).

Šī secība nodrošina, ka datu slānis un biznesa servisi ir gatavi pirms galvenās saskarnes parādīšanas, kas atbilst 3.5.9. sadaļas uzticamības prasībai par stāvokļa saglabāšanu starp sesijām.

4.2.3. *Java* pakotņu organizācija

Visa lietojumprogrammas loģika atrodas pakotnē `java.manicsalsa.soufone_proto`: 146 *Java* avota faili + `module-info.java` (skatīt 5. pielikumu) [13]. Pakotņu sadalījums atbilst 3.3.2. sadaļas izteiktajam slāņu principam un ir izteikts 14. tabula:

14. tabula **Java pakotnes un to nozīme [Autora veidota tabula]**

Nr.	Pakotne / apakšpakotne	Slānis	Galvenā atbildība
1.	`controllers` (43 faili) [14]	Prezentācija	FXML kontrolieri: lapas, pārklājumi, iestatījumu sekcijas, sarakstu šūnas
2.	`managers` (10 faili) [15]	Prezentācija / koordinācija	Navigācija, vadības josla, karstie taustiņi, pārklājumu steks, bibliotēkas pārlāde
3.	`services` (38 faili) [16]	Darbība	Atskaņošana, rinda, bibliotēka, metadati, meklēšana, importēšana
4.	`services.playback` un `services.queue` (kopā 12 faili) [16]	Darbība	Atskaņošanas režīmi, rindas indeksi un pastāvības definējumi
5.	`database`, `database.dao`, `database.models` (kopā 28 faili) [19]	Dati	*SQLite* shēma, DAO, entitīju modeļi (11 tabulas pēc 3.6.3.1 sadaļas)
6.	`config` (11 faili) [20]	Šķērslīnija	Tēmas, fonti, ceļi uz `~/.soufone`, DB konfigurācija
7.	`util` [21] un `session` [22] (kopā 13 faili)	Palīgfunkcijas	Sarakstu ritināšana, ceļu normalizācija, lapu sesijas stāvoklis

Saknes pakotnē atrodas *Launcher* un *SoufoneApplication*. Modulis `module-info.java` skaidri deklarē *JavaFX, SQLite, jaudiotagger* un citas atkarības, kā arī atver kontrolieru pakotnes FXML ielādei - tas atbalsta 3.5.8 uzturamības prasību par skaidru atkarību robežu.

DAO slānis (`database.dao`) implementē SQL piekļuvi tabulām, kas atbilst 3.6.3 sadaļas ER modelim (*AudioDAO, QueueItemDAO, PlayerStateDAO* u.c.). Modeļi (`database.models`) atspoguļo tabulas kolonnas un saites. Prezentācijas slānis ar datiem tiek saistīts caur servisiem un kontrolieru fabrikām (*SoufoneApplication* iestata *AudioDAO* u.c. atkarības *MainLayoutController* ielādei), nevis tiešiem SQL vaicājumiem no FXML - tas atbilst 3.3.2.2 sadaļas prasībai par starpniecību starp slāņiem.

4.2.4. Resursi

Resursi atrodas pakotnē `resources.manicsalsa.soufone_proto` [23]. Šeit atrodas FXML skati, kas atbilst 3.5 sadaļā plānotajām lapām un pārklājumiem:

- Galvenais ietvars ir *MainLayout.fxml*. Tas ietver navigācijas paneli, satura apgabalu un atskaņošanas vadības paneli (skatīt 6. pielikumu);
- Bibliotēkas navigācijas lapas: *AllSongsPage.fxml* (skatīt 7. pielikumu), *SongsPage.fxml* [24], *ArtistsPage.fxml* [25], *AlbumsPage.fxml* [26], *MapsPage.fxml* [27], *FavoritesPage.fxml* [28], *HistoryPage.fxml* [29], *PlaylistPage.fxml* [30];
- Pārklājumi*: QueuePanel.fxml* (skatīt 8. pielikumu), *BigPlayerPage.fxml* [31], *SearchPage.fxml* [32], *GroupTrackListPage.fxml* [33], *LibraryReloadOverlay.fxml* [34];
- Iestatījumu un pamācības lapas: *SettingsPage.fxml* (skatīt 9. pielikumu), iestatījumu sadaļas paneļi (*AppearanceSectionPane.fxml* [35], *AccessibilitySectionPane.fxml* [36], *FileOptionsSectionPane.fxml* [37], *KeyBindingsSectionPane.fxml* [38], *AboutSectionPane.fxml* [39]) un *GuidePage.fxml* (skatīt 10. pielikumu);
- Bibliotēkas saturu sarakstu elementi: *AudioItem.fxml* (skatīt 11. pielikumu), *GroupItem.fxml* [40], *QueueItem.fxml* [41], *GlobalContextMenu.fxml* (skatīt 12. pielikumu).

Pakotnē arī atrodas vizuālā noformējuma sistēma (3.4.2.8 un 3.5.6):

- `themes/structure/base-structure.css` – kopīgais sistēmas izkārtojums (skatīt 7. pielikumu);
- `themes/colors/` - pieci krāsu motīvi: *Dark-Abyss, Light-Dawn, Deep-Ember, Evening-Tea* un *Violet-Dusk* (atbilst prasībai par vismaz pieciem motīviem) [42];
- `themes/fonts/` - satur iebūvētos *Noto* fontus (CJK/*Serif* varianti), to pilnā fontu kopums izstrādājot instalētājus var tikt papildināts no ārēja ZIP, ko *Maven* profili lejupielādē no projekta repozitorijas pakotnēm. Ja tas netiek darīts, direktorija ir tukša.

Citi resursi: *img/default_cover.png* - noklusējuma albuma vāks, ko lieto kad audio faila metadatos nav attēla [43].

Pēc 3.6.4 sadaļas norādēm, lietotāja datorā ārpus pakotnes tiek glabāti:

- *music_library.db* – SQLite datubāze;

- *theme_config.properties* – izskata preferences;
- *hot_key_config.properties* – karsto taustiņu preferences;
- *app_preferences.properties* – aplikācijas procesu preferences.

Repozitorija `config/templates/` satur noklusējuma preferences veidnes, ko programma kopē un izmanto pirmajā programmas palaišanas brīdī [44]. Šīs faktiskās vērtības tiek rakstītas `.soufone/`, kas definētas *DatabaseConfig.java* (skatīt 14. pielikumu) un *AppConfigPaths.java* (skatīt 15. pielikumu).

4.3. Arhitektūras realizācija

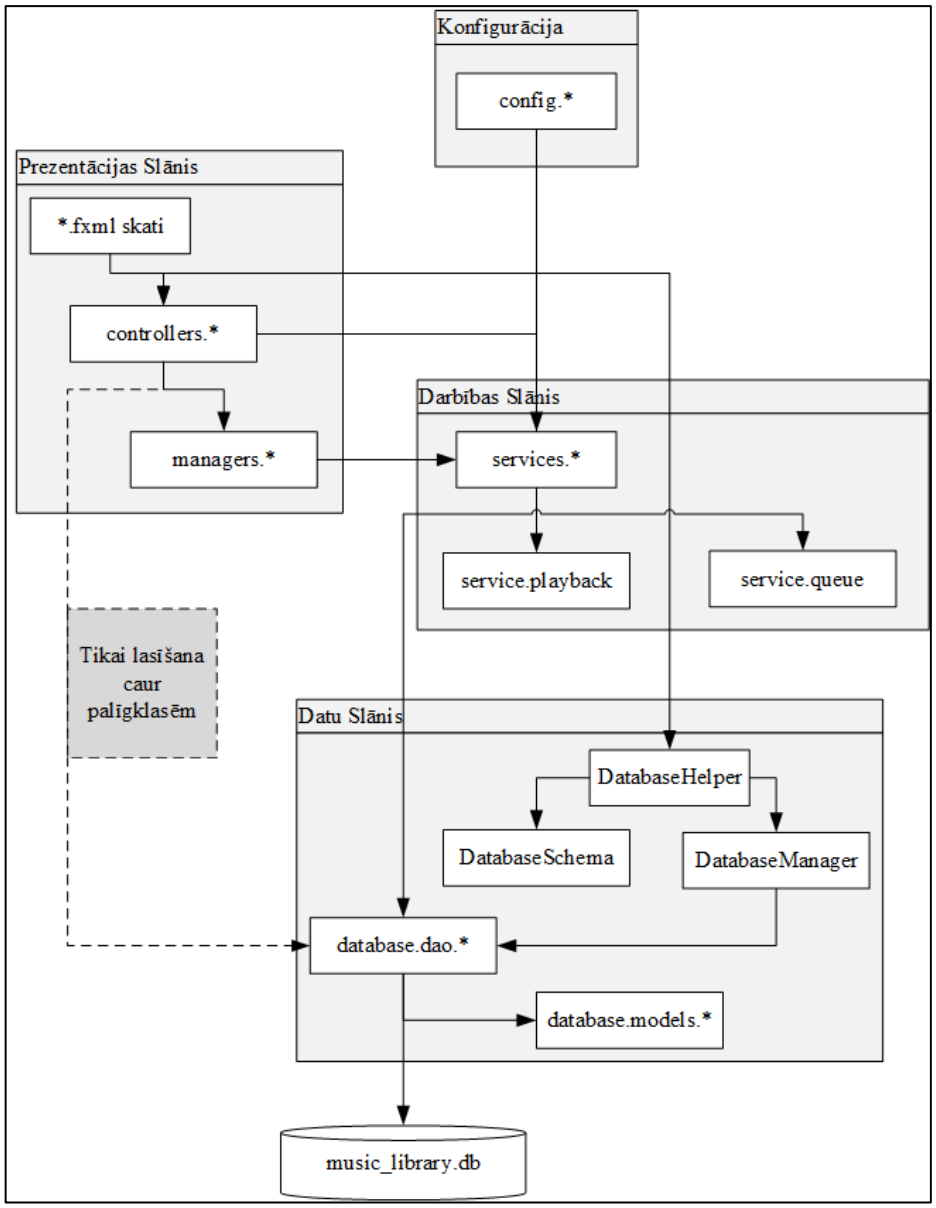
Trešajā nodaļā sistēma tika plānota kā trīs loģiski atdalīti slāņi - prezentācija, darbība un dati - ar skaidru atbildību sadalījumu (skatīt 3.3.2 sadaļu). Realizācijā šis princips ir saglabāts pakotņu līmenī: *Java* kods ir sadalīts tā, lai lietotāja saskarne, biznesa loģika un *SQLite* piekļuve varētu attīstīties neatkarīgi, bet palaišanas brīdī sadarbotos caur definētām metožu izsaukumu ķēdēm.

Arhitektūras pamatā ir vienvirziena datu plūsma:

- prezentācijas komponenti (kontrolieri un pārvaldnieki) apstrādā lietotāja ievadi un atjaunina skatus;
- darbības slāņa servisi koordinē atskaņošanu, bibliotēku un rindu;
- datu slānis nodrošina uzstātību caur DAO un entitīju modeļiem.

SQL vaicājumi atrodas tikai DAO klasēs - FXML un kontrolieros tie netiek rakstīti, kas atbilst 3.5.8 sadaļas uzturamības prasībai.

Slāņu attiecības prototipā ir attēlotas diagrammā (skatīt 27. attēls).



27. attēls **Realizētā trīsslāņu arhitektūra ar galvenajām klasēm [Autora veidots attēls]**

Lietotāja darbības tiek virzītas secībā kontrolieris / pārvaldnieks → serviss → DAO. Rakstīšanas un biznesa operācijas (bibliotēkas skenēšana, rindas maiņa, favorītu atzīmēšana, atskaņošanas stāvokļa saglabāšana) vienmēr iet caur servisiem. Kontrolieri var tieši izmantot DAO tikai lasāmos scenārijos - galvenokārt sarakstu attēlošanai, izmantojot palīgklases kā *LazyAudioList* (skatīt . pielikumu) un *LibraryIndexCache* (skatīt . pielikumu), lai saistītu *JavaFX* sarakstus ar datubāzi bez SQL dublēšanas UI kodā.

4.3.1. Prezentācijas slānis

Prezentācijas slānis realizē 3.5 sadaļā plānoto saskarni: navigācijas paneli, satura apgabalu, mūzikas vadības paneli un pārklājumus. Tas sastāv no trim savstarpēji saistītiem elementu veidiem.

FXML skati (`resources` pakotne [23]) definē vizuālo izkārtojumu atsevišķām lapām un atkārtojamiem UI elementiem. Galvenais ietvars ir *MainLayout.fxml*, kas ietver trīs nemainīgās zonas (skatīt 3.5.2 - 3.5.4). Pārējās lapas tiek ielādētas dinamiski satura zonā vai kā pārklājumi virs tās.

Kontrolieri (`controllers` pakotne [14]) saista FXML elementus ar programmas loģiku. Katram skatam atbilst kontrolieris, kas implementē *Initializable* un apstrādā pogas, sarakstus un ievades laukus. Tipiski piemēri:

15. tabula **Kontrolieri un to loma [Autora veidota tabula]**

Nr.	Kontrolieris	Loma
1.	*MainLayoutController* (skatīt 4. pielikumu)	Lietotnes čaula: lapu ielāde, pārklājumu steks, globālā konteksta izvēlne
2.	*SongController* [45]	„*All Songs*" bibliotēkas skats (dziesmas, izpildītāji, albumi, mapes)
3.	*PlaylistController* [46]	Lietotāja atskaņošanas saraksti
4.	*QueueController* [47]	Atskaņošanas rindas panelis
5.	*SearchController* [48]	Meklēšanas pārklājums
6.	*SettingsController* [49]	Iestatījumu lapa un sekciju koordinācija
7.	*GuideController* [50]	Iebūvētā lietotāja pamācība
8.	*AppearanceSectionController, FileOptionsSectionController, KeyBindingsSectionController* u.c. [14]	Atsevišķas iestatījumu sadaļas

MainLayoutController tiek injicēts ar *AudioDAO* caur *SoufoneApplication* kontrolieru fabriku (skatīt 4.2.2), lai čaula varētu nodrošināt bibliotēkas lapu ielādi bez globāla stāvokļa.

Pārvaldnieki (`managers` pakotne [15]) izdala sarežģītu UI koordināciju no lapu kontrolieriem. Tie nav atsevišķs arhitektūras slānis plānošanas dokumentācijas izpratnē, bet praksē darbojas kā prezentācijas slāņa koordinatori starp FXML, kontrolieriem un servisiem:

16. tabula **Pārvaldnieki un to atbildība [Autora veidota tabula]**

Nr.	Pārvaldnieks	Atbildība
1.	*NavigationManager* [51]	Navigācijas paneļa maršrutēšana, ikonu/teksta režīms, platuma animācija (3.5.3)
2.	*PlayerControlsManager* [52]	Apakšējā vadības josla: atskaņošana, skaļums, progress, favorīts, režīms (3.5.4)
3.	*PageRegistry* [53]	FXML ceļu un lapu identifikatoru reģistrs
4.	*ContentOverlayStack* [54]	Pārklājumu (*SearchPage*, *BigPlayerPage*, *GroupTrackListPage*) empilēšana
5.	*LibraryReloadManager* [55]	Bibliotēkas pārlādes progresa pārklājums un atcelšana
6.	*HotKeyManager* [56]	Globālo karsto taustiņu reģistrācija un izpilde (3.4.2.10)

7.	*SearchOverlayOpener* [57], *ContextMenuHost* [58]	Meklēšanas un konteksta izvēlnes atvēršana

PlayerControlsManager ir centrālais koordinators atskaņošanas vadībā, tas saista UI pogas ar *AudioPlayerService* (transporta kontrole) un *QueueService* (rindas loģika), piemēram, pārejot uz nākamo ierakstu pēc ieraksta beigām.

Prezentācijas slānis neglabā biznesa stāvokli ilgtermiņā - tas atspoguļo servisu un DAO datus, klausās to notikumus un nodod lietotāja komandas atpakaļ servisiem.

4.3.2. Darbības slānis

Darbības slānis (`services` pakotne un apakšpakotnes `playback`, `queue` [16]) realizē 3.3.2.2 aprakstītās funkcionālās grupas: audio atskaņošanu, rindu un vēsturi, bibliotēkas pārvaldību un papildservisus. Lielākā daļa servisu izmanto *singleton* pieeju, lai visā lietotnē būtu viens koordinēts stāvoklis - tas atbilst prasībai par konsekventu atskaņotāja un rindas uzvedību (3.4.2.3, 3.4.2.4 sadaļas).

Galvenie servisi un to saistība ar 3. nodaļas prasībām:

17. tabula **Galvenie servisi [Autora veidota tabula]**

Nr.	Serviss	Funkcionālā prasība	Īss apraksts
1.	*AudioPlayerService* (skatīt 18. pielikumu)	3.4.2.1, 3.4.2.2, 3.4.2.3	Atskaņošana, pauze, *seek*, skaļums; izmanto *Mp3Player / JavaFxMediaPlayerAdapter*
2.	*QueueService* [60]	3.4.2.4	Aktīvā rinda, režīmi (*PlayMode*), persistences caur *QueuePersistence*
3.	*DirectoryService* [61]	3.4.2.5, 3.4.2.7	Mapju skenēšana, importēšana, avotu uzturēšana
4.	*MetadataService* [62]	3.4.2.7	ID3 un iegulto tagu nolasīšana (*jaudiotagger*)
5.	*AudioLoaderService* [63]	3.4.2.1	Faila ielāde un formāta pārbaude pirms atskaņošanas
6.	*LibraryChangeService* [64]	3.4.2.5	Bibliotēkas izmaiņu paziņojumi UI slānim pēc skenēšanas
7.	*LibraryBootstrap* [65]	3.5.7	Fona bibliotēkas indeksa priekšielāde startā
8.	*PlayHistoryService* [66]	3.6.3.11	Nesen atskaņoto ierakstu reģistrācija
9.	*FavoriteService* [67]	3.5.4	Favorītu stāvokļa maiņas un klausītāji
10.	*CoverImageCache* [68]	3.6.3.9	Albuma vāku kešošana atskaņošanas skatos
11.	*DatabaseSetupService* [69]	3.6.2	Datubāzes gatavības pārbaude palaišanā
12.	*PlayerActionFeedbackService* [70]	3.5.4	Īsi paziņojumi virs vadības paneļa
13.	*MissingFileRegistry* [71]	3.5.9	Nepieejamu failu reģistrācija

Apakšpakotne `services.playback` satur atskaņošanas politikas detaļas, kas atdalītas no *AudioPlayerService* transporta loģikas: *PlayMode* (secīgi / jauktā secībā / atkārtot), *PlayingQueueIndexResolver*, *ShuffleNavigation*, *PlaybackVolumeCurve*, *PlaybackTrackRef* [17].

Apakšpakotne `services.queue` nodrošina rindas persistenci (*QueuePersistence*) un rindas-audio saites (*QueueAudioStubs*) [18].

Tipisks darbības plūsmas piemērs - ieraksta atskaņošana no bibliotēkas:

1. *SongController* vai *GlobalContextMenuController* izsauc *QueueService* (pievienošana rindai / sākt atskaņošanu).
2. *PlayerControlsManager* koordinē *AudioPlayerService.play(...)* ar aktuālo rindas indeksu.
3. *AudioPlayerService* ielādē faila ceļu caur *AudioDAO*, izveido Mp3Player un atjaunina *PlayerStateDAO*.
4. Ieraksta beigās *AudioPlayerService* paziņo *PlayerControlsManager*, kas izsauc nākamo rindas loģiku atbilstoši *PlayMode*.

Bibliotēkas skenēšanas plūsma iet caur *DirectoryService*, kas izmanto *MetadataService* katra faila apstrādei un vairākus *DAO* (AudioDAO, ArtistDAO, AlbumDAO, CoverImageDAO u.c.) [59] datu saglabāšanai - tas realizē 3.4.3.1 lietojumgadījumu „Nolasīt metadatus" kā obligātu apakšdarbību.

4.3.3. Datu slānis

Datu slānis (`database` pakotne [19]) realizē 3.6 sadaļas *SQLite* modeli. Tas nodrošina, ka bibliotēkas indekss, rinda, atskaņotāja stāvoklis un atskaņošanas vēsture tiek glabāti lokāli failā ~/.soufone/music_library.db (skatīt 3.6.4).

Galvenās klases:

- *DatabaseConfig* (skatīt 14. pielikumu) - datubāzes un lietotāja datu mapes ceļš (`~/.soufone`);
- *DatabaseHelper* [72] - procesa vienīgais *SQLite* fasāde: savienojums, shēmas inicializācija, transakcijas;
- *DatabaseSchema* [73] - DDL: 11 tabulu izveide, indeksi, noklusējuma dati (atbilst 3.6.3.1 ER modelim);
- *DatabaseManager* [74] - viena JDBC savienojuma DAO konteiners, inicializē visus DAO un ārējās atslēgas.

DAO slānis (`database.dao` pakotne [59], visas manto *BaseDAO* (skatīt 19. pielikumu)) iekļauj: *AudioDAO, ArtistDAO, AlbumDAO, ArtistAudioDAO, AlbumAudioDAO, CoverImageDAO, PlaylistDAO, PlaylistAudioDAO, QueueItemDAO, PlayerStateDAO, PlayHistoryDAO* [59].

Katra DAO klase kapsulē SQL vaicājumus vienai tabulai vai saistību tabulai. Servisi un, reti, kontrolieri izsauc DAO metodes, nevis raksta SQL tieši.

Modeļi (`database.models` pakotne [75]) atspoguļo tabulu rindas: *Audio, Artist, Album, Playlist, QueueItem, PlayerState, PlayHistory, CoverImage* u.c. Tie tiek izmantoti kā datu pārvadātāji starp DAO un servisiem/kontrolieriem.

Datu plūsma atbilst 3.6.2 konteksta diagrammai.

4.3.4. Konfigurācijas komponentes

Lietotāja preferences un vizuālās izvēles, kas nav daļa no *SQLite* shēmas, realizētas atsevišķā `config` pakotnē [20]. Tās darbojas šķērslīņā starp visiem slāņiem un atbalsta 3.4.2.8 - 3.4.2.11 un 3.6.4 prasības.

18. tabula **Galvenās konfigurācijas [Autora veidota tabula]**

Nr.	Klase	Loma
1.	*DatabaseConfig* (skatīt 14. pielikumu)	Datubāzes ceļš un mapes izveide
2.	*AppConfigPaths* (skatīt 15. pielikumu)	Konfigurācijas failu ceļu risināšana
3.	*ThemeConfig* [76]	Krāsu motīvu un fontu pielietošana visai ainai (3.4.2.8, 3.5.6)
4.	*AppPreferences* [77]	Bibliotēkas pārbaude startā, atskaņošanas preferences (3.4.2.11)
5.	*HotKeyManager* [56] + *AppConfigPaths.hotKeyConfig()* (skatīt 15. pielikumu)	Karsto taustiņu saistības (3.4.2.10)
6.	*FontSizePercent, ScriptFontResolver,* u.c. [20]	Teksta izmēra un fontu pielāgošana (3.4.2.9)

ThemeConfig ielādē *base-structure.css* un izvēlēto motīvu no `themes/colors/`, reģistrē fontus no resursiem un saglabā lietotāja izvēli *theme_config.properties*. *AppPreferences* un karsto taustiņu konfigurācija tiek ielādēta startā un saglabāta izmaiņu brīdī - tas realizē 3.4.3.5 lietojumgadījumu „Saglabāt iestatījumus” un „Ielādēt iestatījumus programmas startā”.

Repozitorija `config/templates/` satur noklusējuma *.properties* veidnes, pirmajā palaišanā programma izveido faktiskos failus lietotāja mapē (skatīt 4.2.4).

4.4. Testēšana

Prototipa testēšana nodrošina, ka kritiskās sistēmas daļas - datu glabāšana, bibliotēkas apstrāde un atskaņošanas rindas loģika - darbojas prognozējami, mazinot regresijas iespējamību ritošā izstrādes modelī. Testēšana aptver gan automatizētos vienību testus, gan lietotāju testēšanu.

4.4.1. Testēšanas pieeja un rīki

Automatizētie testi atrodas `src/test/java` pakotnē [78]. Testēšanai izmantots *JUnit* 5 (skatīt 2.3.7).

Testu sadalījums pēc mērķa:

- vienību tests - izolētas loģikas testēšana bez DB un UI (PlayingQueueIndexResolverTest [], QueueIndexAdjustmentsTest []);
- integrācijas tests - pārbauda *SQLite* shēmu, DAO, transakciju pareizumu (DatabaseBaseTest []);
- servisu tests - bibliotēkas importa un pārlādes noteikumu pārbaude (DirectoryServiceTest [], DirectoryServicePreviewTest []).

JavaFX saskarne (FXML, kontrolieri, vizuālie pārklājumi) netiek testēta ar automatizētiem UI testiem. Šīs funkcijas tika pārbaudītas lietojamības testos ar klientu.

Ritošās izstrādes ietvaros tiek pielietos *CodeScene.io* rīks (skatīt 2.3.9 sadaļu), lai laicīgo novērtu regresijas pazīmes un novērstu koda ožu laicīgi.

4.4.2. Lietojamības testēšana

Sekojot prasībām, kas definētas 3. nodaļā, tiek pārbaudītas implementēto funkciju pareizums, saskarnes draudzīgums kopā ar klientu.

4.5. Realizācijas secinājumi

Šī sadaļa sniedz kopēju vērtējumu par izstrādes rezultātu.

4.5.1. Prasību izpildes kopsavilkums

Prototips realizē klienta uzdevumā noteikto darbības sfēru - bezsaistes darbvirsmas audio atskaņotāju ar bibliotēkas pārvaldību, atskaņošanas vadību un personalizāciju. Galvenās 3.4.2 sadaļas noteiktās prasības ir ieviestas:

- lokāla atskaņošana no bibliotēkas un rindas, ar vairāku formātu atbalstu;
- pilna pamatvadība un atskaņošanas rindas pārvaldība ar persistenci starp sesijām;
- *SQLite* bibliotēka ar metadatu indeksēšanu un meklēšanu;
- pieci krāsu motīvi, pielāgojami fonti un saglabāti lietotāja iestatījumi;
- karstie taustiņi un angliskā saskarne.

Nefunkcionālās prasības no 3.5 sadaļas ir realizētas galvenajā apjomā: trīszonu UI struktūra, pārklājumi, veiktspējas optimizācijas (optimizēta ierakstu ielāde, meklēšanas optimizācija, fona ielāde) un lokāla datu uzticamība bez tīkla atkarības.

4.5.2. Izstrādes secinājumi un ierobežojumi

Projekta rezultāts ir darbojošs mūzikas atskaņošanas prototips, kas demonstrē 3. nodaļā plānoto sistēmu reālā implementācijā. Prasības ir izstrādātas konkrētā *JavaFX* lietotnē ar lokālu *SQLite* bibliotēku, modulāru kodu struktūru un izplatāmiem instalētājiem *Windows* un *Linux* vidēm.

Izstrādes process apstiprināja plānošanas fāzē izvēlēto pieeju: slāņu atdalījums atviegloja gan funkciju pievienošanu (rinda, meklēšana, motīvi), gan kļūdu labošanu, bet *Maven* un *GitLab* repozitorijs nodrošināja atkārtojamu būvējumu un versiju kontroli. Automatizētie testi deva drošību kritiskajās vietās - datubāzē, bibliotēkas importā un rindas loģikā, savukārt manuālā testēšana pārbaudīja lietotāja pieredzi atbilstoši 3.4.3 un 3.5 prasībām.

Neskatoties uz to, ka prototips izpilda galvenās prasības, realizācijā ir šādi ierobežojumi:

- testēšanas segums - automatizēti testi koncentrējas uz datu slāni un biznesa loģiku; UI un audio pipeline lielākoties pārbaudīti manuāli. Nākotnē varētu paplašināt ar UI testiem vai integrācijas testiem ar testa audio failiem;
- platformu atbalsts - *Linux* testēšana notika *Arch Linux* vidē, citās distribūcijās *GStreamer* pakotņu nosaukumi var atšķirties (dokumentēts [12]);
- prototipa statuss - risinājums ir funkcionāls prototips, ne komerciāli gatavs produkts (piem., *Windows MSI* nav parakstīts, kā norādīts 3.3.3).

Kopumā prototips atbilst kvalifikācijas darba mērķim: parādīt spēju izstrādāt strukturētu, funkcionālu un bezsaistē lietojamu programmatūru, balstoties uz klienta prasībām, lietojamības principiem un mūsdienīgu *Java* izstrādes praksi.

5. LIETOTĀJA CEĻVEDIS

5.1. Ievads

Šī nodaļa apraksta, kā lietotājam uzstādīt un izmantot prototipu ikdienas darbībās. Ceļvedis ir strukturēts secīgi: vispirms instalācija un pirmā bibliotēkas izveide, pēc tam saskarnes orientēšanās, atskaņošanas vadība, bibliotēkas skati, meklēšana, konteksta izvēlne un iestatījumi.

Prototipa lietotāja saskarne ir angļu valodā (skatīt 3.5.2 sadaļu). Šajā nodaļā funkciju nosaukumi tiek doti tādā formā, kādā tie parādās programmā, lai lietotājs varētu tos viegli atrast. Detalizētāka, meklēšanai pielāgojama palīdzība ir pieejama arī pašā lietotnē sadaļā *User Guide* (skatīt 5.10. sadaļu).

5.2. Instalācija un sistēmas prasības

Prototips ir paredzēts *Windows 10* un *Linux* (x86_64) vidēm. Izplatāmās pakotnes ir pieejamas projekta *GitLab* izlaidumu lapā [79].

No šīs izlaiduma lapas var lejupielādēt gatavu instalācijas pakotni attiecīgajai operētājsistēmai. Pēc lejupielādes nav nepieciešams atsevišķi instalēt *Java* - izplatāmajās pakotnēs ir iekļauta nepieciešamā izpildvide.

Ja nepieciešams būvēt programmu no avota koda (izstrādei, pielāgotai pakotnei vai jaunākai versijai), repozitorija mapē `docs/` [12] ir pieejamas instrukcijas: *setup.md* (vispārīga iestatīšana), *install-windows.md* un *install-linux.md* (instalācijas pakotņu veidošana ar *Maven* profiliem).

5.2.1. Windows instalācija

1. Atveriet v1.0.0 izlaiduma lapu un lejupielādējiet *Windows* instalācijas failu (*Audio Player Prototype Windows x86_64.msi*).
2. Palaidiet lejupielādēto instalētāju un sekojiet instalācijas vednim.
3. Pēc instalācijas palaidiet programmu no izvēlnes *Audio Player Prototype* vai darbvirsmas saīsnes.

Svarīgi: ja instalētājs nav digitāli parakstīts, *Windows SmartScreen* var parādīt brīdinājumu. Tas ir tipisks neparakstītām instalācijas pakotnēm (skatīt 2.4 un 3.3.3. sadaļas). Šādā gadījumā izvēlieties papildu informāciju un apstipriniet palaišanu.

Sistēmas prasības: *Windows* 64 bit (x86_64), vismaz 4 GB operatīvās atmiņas; ieteicams vismaz 1 GB brīvas RAM lietotnes palaišanas brīdī.

5.2.2. Linux instalācija

1. Atveriet v1.0.0 izlaiduma lapu un lejupielādējiet *AppImage* failu (*Audio-Player-Prototype-Linux-x86_64.AppImage*).
2. Piešķiriet failam izpildes atļauju un palaidiet:

```
`chmod +x Audio-Player-Prototype-Linux-x86_64.AppImage
   ./Audio-Player-Prototype-Linux-x86_64.AppImage`
```

3. Ja sistēmā nav pieejams FUSE atbalsts, izmantojiet alternatīvo palaišanas režīmu:

```
`./Audio-Player-Prototype-Linux-x86_64.AppImage --appimage-extract-and-run`
```

Audio formātu atbalsts: M4A/AAC atskaņošanai *Linux* vidē nepieciešami sistēmā uzstādīti *GStreamer* spraudņi. *Arch Linux* piemēram:

```
`sudo pacman -S gst-plugins-base gst-plugins-good gst-libav`
```

Citu distribūciju pakotņu nosaukumi ir norādīti *docs/install-linux.md* [80].

Sistēmas prasības: *Linux* x86_64, vismaz 4 GB RAM; ieteicams pietiekami brīvas atmiņas bibliotēkas ielādei.

5.2.3. Lietotāja datu vieta

Pēc pirmās palaišanas programma izveido lokālu datu mapi:

- Linux: `~/.soufone`
- Windows: `C:\Users\<lietotājvārds>\.soufone`

Šeit tiek glabāta *SQLite* bibliotēka (*music_library.db*) un iestatījumu faili. Audio faili paliek jūsu izvēlētajās mapēs - programma tos nedublē.

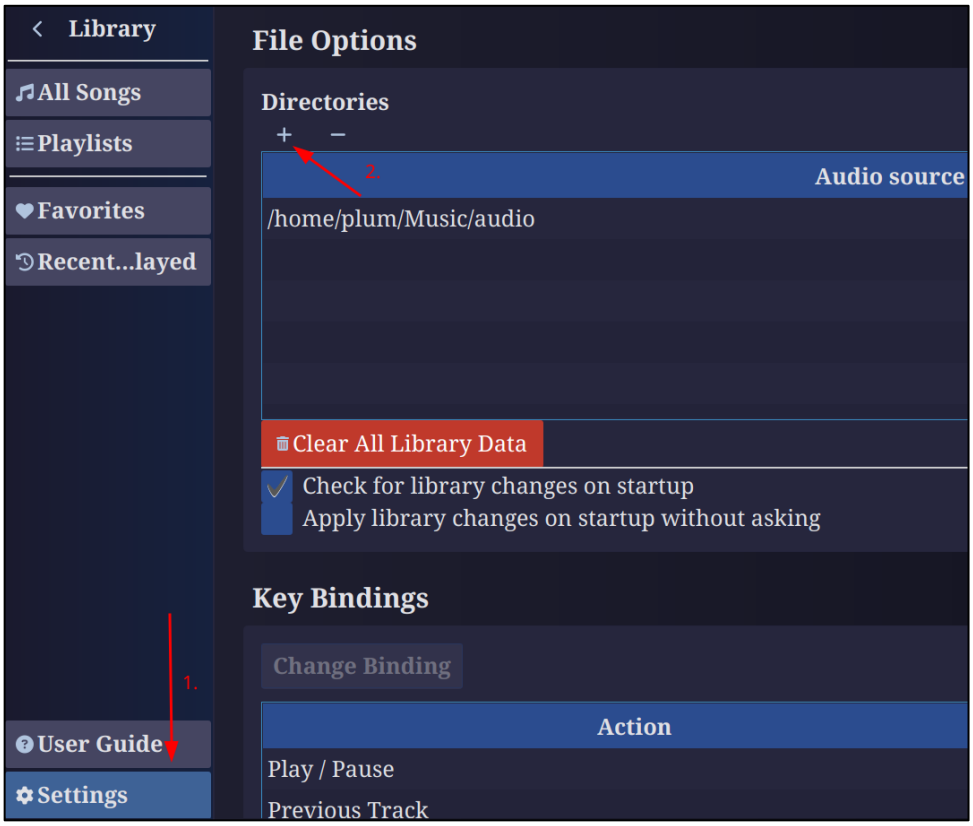
5.3. Pirmie soļi - mūzikas bibliotēkas izveide

Prototips atskaņo audio failus no lietotāja norādītām mapēm. Bez avota mapes bibliotēka ir tukša.

5.3.1. Avota mapes pievienošana

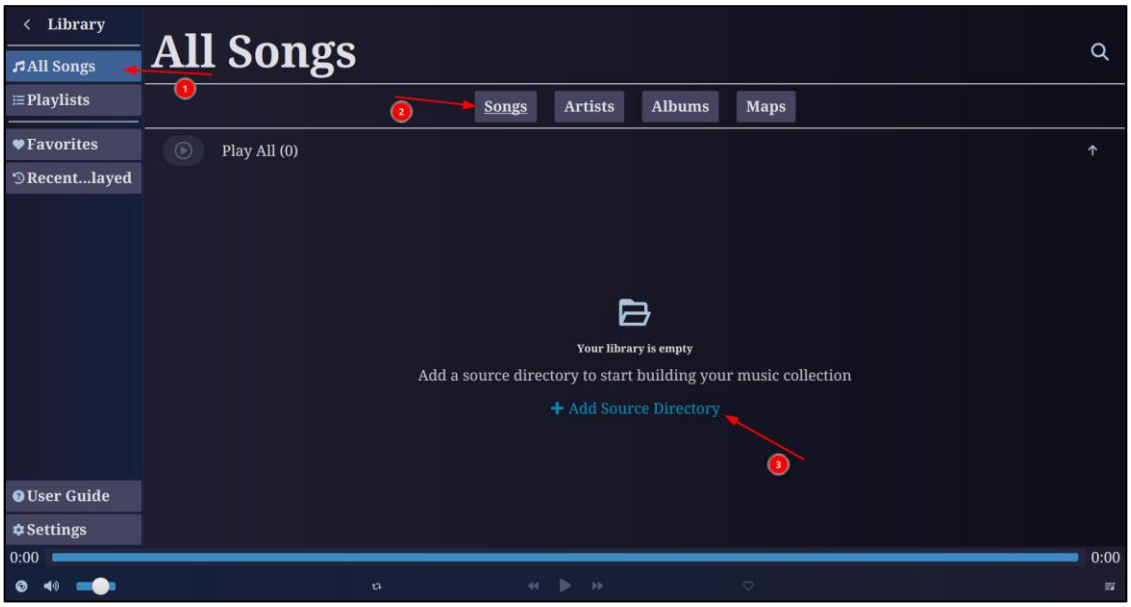
Bibliotēku var izveidot divos veidos:

1. *Settings → File Options → Directories* — izmantojiet pogu *Add (+)* un izvēlieties mapi ar mūziku (skatīt 28. attēls).



28. attēls **Mapes izvēle no iestatījumiem [Autora veidots attēls]**

2. *All Songs* lapā, ja bibliotēka ir tukša, sekojiet saitei *Add Source Directory*, kas ved uz iestatījumiem (skatīt 29. attēls).



29. attēls ***Add Source Directory*** **novietojums [Autora veidots attēls]**

Programma rekursīvi skenē izvēlēto mapi, nolasa metadatus un aizpilda skatus *Songs, Artists, Albums* un *Maps*. Skenēšanas laikā tiek rādīts progresa dialogs.

Atbalstītie formāti: MP3, WAV, FLAC, M4A, AAC, OGG un WMA.

Ierobežojumi pievienošanai:

- Ja pievienojat vecāku mapi, kurai jau ir apakšmapes kā avoti, programma var piedāvāt aizstāt apakšmapes ar vecāku mapi.
- Apakšmapes pievienošana, ja tās vecākmāte jau ir avots, netiek atļauta.

5.3.2. Bibliotēkas pārlāde

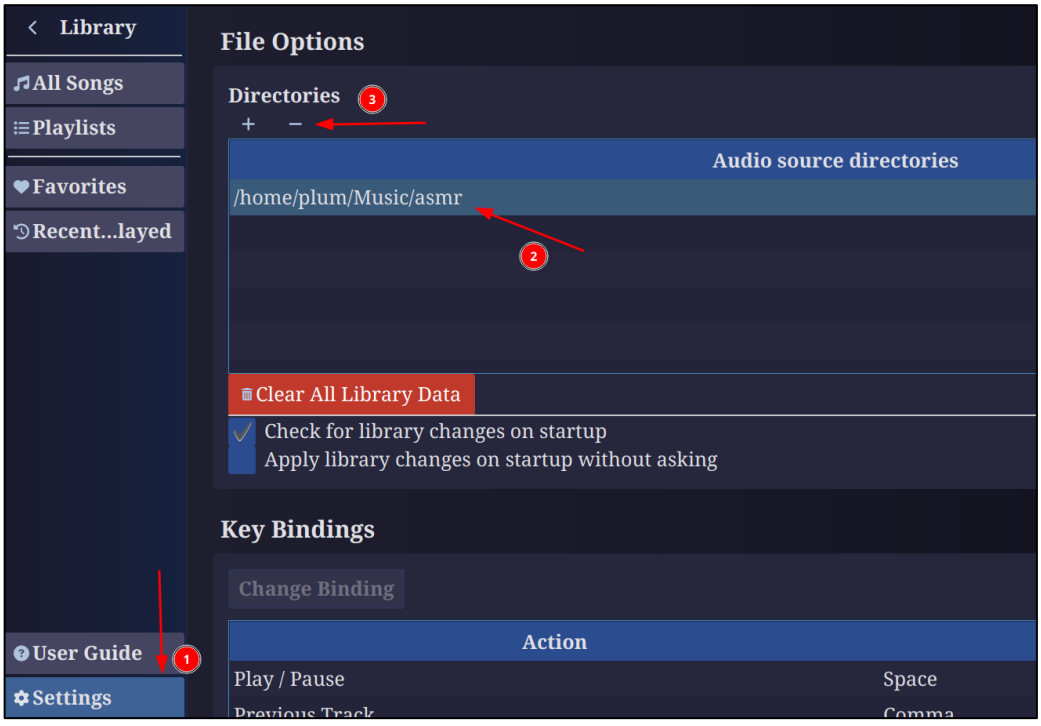
Ja mūziku pievienojat vai dzēšat ārpus programmas, izmantojiet *Reload Library*:

1. labās peles pogas klikšķis gandrīz jebkur programmā atver konteksta izvēlni - *Reload Library.*
2. tastatūras saīsni (noklusējums: R), ko var mainīt *Settings* → *Key Bindings*.

Pirms skenēšanas parādās apstiprinājuma dialogs. Skenēšanas laikā loga augšdaļā redzams progresa pārklājums ar iespēju atcelt. Pēc pabeigšanas tiek atjaunināti izpildītāji, albumi, mapes un atskaņošanas saraksti.

5.3.3. Avota mapes noņemšana

Settings → *File Options* - atlasiet ceļu tabulā un nospiediet *Remove* (−) (skatīt 30. attēls).



30. attēls **Avota mapes noņemšana [Autora veidots attēls]**

No bibliotēkas datubāzes tiek noņemti tikai ieraksti no šīs mapes; faili diskā netiek dzēsti. Ja noņemtie ieraksti bija rindā vai atskaņojās, atskaņotājs pielāgo stāvokli.

5.3.4. Bibliotēkas pārbaude startā

Settings → File Options:

- *Check for library changes on startup* (noklusējums: ieslēgts) - pēc palaišanas notiek fona skenēšana;
- *Apply library changes on startup without asking* - automātiska izmaiņu piemērošana (pieejama, ja ieslēgta augšējā pārbaude).

Ja otrā opcija ir izslēgta, izmaiņas tiek parādītas apstiprinājuma dialogā.

5.3.5. Pazuduši vai pārvietoti faili

Ja fails ir dzēsts vai pārvietots, ieraksta rinda tiek vizuāli atšķirīga. Mēģinot atskaņot, parādās brīdinājums ar divām darbībām:

- *Reload library now* - pārskenē bibliotēku un noņem nepieejamos ierakstus;
- *Go to missing track* - pārslēdz uz *All Songs* un ritina līdz attiecīgajam ierakstam.

5.4. Saskarnes pārskats un navigācija

Programmas logs sastāv no trim pastāvīgām zonām (skatīt 3.5.2 sadaļu):

1. Navigācijas panelis (pa kreisi) - pārslēgšanās starp galvenajām lapām.
2. Satura apgabals (centrā) - bibliotēkas skati, iestatījumi, pamācība.
3. Atskaņošanas vadības panelis (apakšā) - progress, transporta pogas, skaļums, rinda.

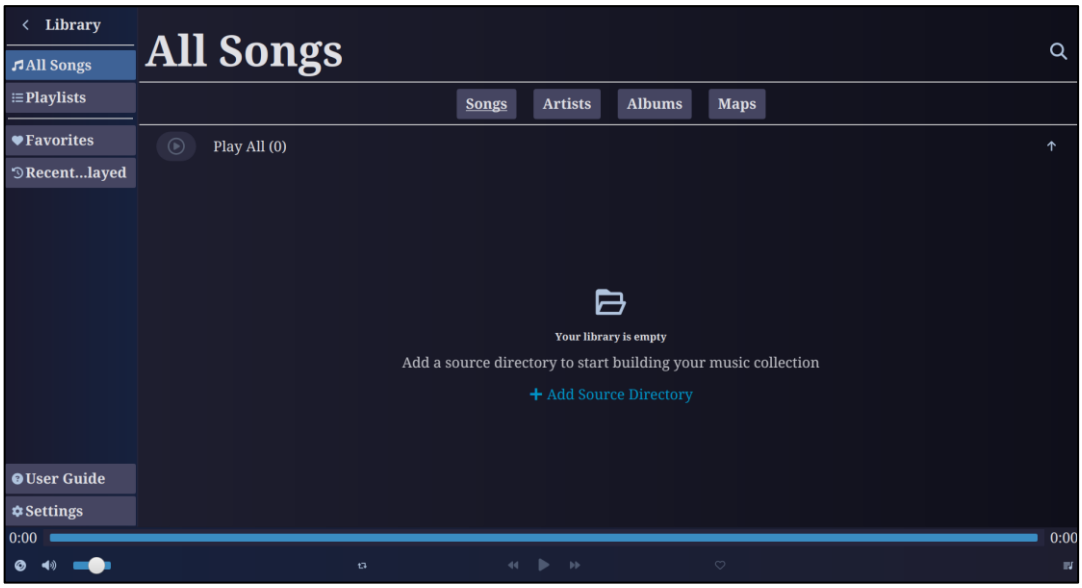
5.4.1. Navigācijas panelis

Paneli spēj:

- mainīt platumu, velkot labo malu (platums nevar pārsniegt aptuveni pusi no loga);
- samazināt līdz ikonu režīmam ar bultiņu augšējā kreisajā stūrī; ikonu režīmā pogas rāda tikai ikonas, ne tekstu.

Galvenās lapas:

- *All Songs* - galvenā bibliotēka (dziesmas, izpildītāji, albumi, mapes) (skatīt 31. attēls);

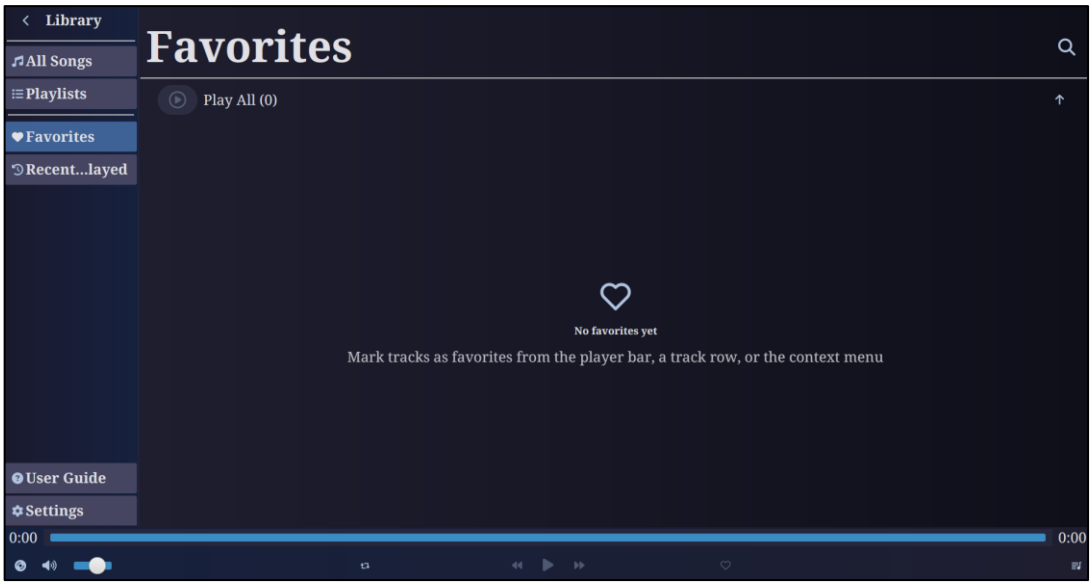


31. attēls ***All Songs*** **lapa [Autora veidota ekrānkopija]**

- *Playlists* - lietotāja atskaņošanas saraksti (skatīt 32. attēls);

32. attēls ***Playlist*** **lapa [Autora veidota ekrānkopija]**

- *Favorites* - favorītu ieraksti (skatīt 33. attēls);

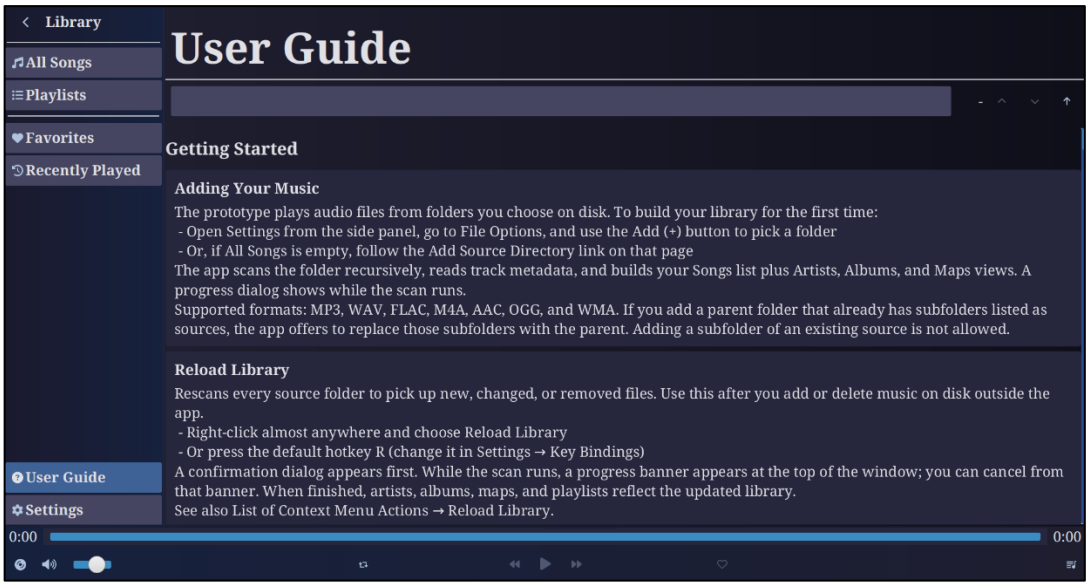


33. attēls ***Favorites*** **lapa [Autora veidota ekrānkopija]**

- *Recently Played* - nesen atskaņotie (lapas virsraksts: *History*) (skatīt 34. attēls);

34. attēls ***History*** **lapa [Autora veidota ekrānkopija]**

- *User Guide* - iebūvētā pamācība (skatīt 35. attēls);



35. attēls ***User Guide*** **lapa [Autora veidota ekrānkopija]**

- *Settings* - Iestatījumi un bibliotēkas uzturēšana (skatīt 36. attēls);

36. attēls ***Settings*** **lapa [Autora veidota ekrānkopija]**

Vienlaikus aktīva ir tikai viena navigācijas poga.

5.4.2. Konteksta izvēlne

Labās peles pogas nospiešana (izņemot teksta laukus) atver konteksta izvēlni ar darbībām atkarībā no tā, uz ko noklikšķināts. Pilns darbību saraksts atrodams 5.8. sadaļā.

5.4.3. Karstie taustiņi

Daudzas darbības pieejamas ar tastatūru, kamēr lietotnes logs ir aktīvs (ne teksta ievades laukā). Noklusējuma saīsinājumi:

19. tabula **Noklusējuma saīsinājumi [Autora veidota tabula]**

Taustiņš	Darbība
Space	Atskaņot / Pauzēt
, (Comma)	Iepriekšējais ieraksts
. (Period)	Nākamais ieraksts
B	Atvērt / aizvērt lielo atskaņošanas skatu
N	Atvērt / aizvērt rindas paneli
F	Favorīts pašreiz atskaņotajam ierakstam
C	Mainīt atskaņošanas režīmu
R	Pārlādēt bibliotēku

Visas saistības var mainīt *Settings → Key Bindings*.

5.5. Atskaņošanas vadība

Apakšējais vadības panelis ir redzams visu laiku un ļauj kontrolēt atskaņošanu neatkarīgi no atvērtās lapas (skatīt 3.5.4 sadaļu un 37. attēls).

37. attēls **Atskaņošanas vadības panelis [Autora veidots attēls]**

5.5.1. Pamatvadība

Play / Pause - centrālā poga; ikona mainās atbilstoši stāvoklim.

Previous / Next - pāreja uz iepriekšējo vai nākamo ierakstu rindā (atkarībā no atskaņošanas režīma).

Progress bar - laika līnija starp taimeriem; velkot vai noklikšķinot var mainīt atskaņošanas pozīciju. Bez ielādēta ieraksta rāda --:--.

5.5.2. Atskaņošanas režīmi

Poga *Play Mode* cikliski pārslēdz:

1. *Play in Order* - secīgi no rindas augšas uz leju.
2. *Shuffle* - nejauša secība rindas ietvaros.
3. *Repeat List* - pēc pēdējā ieraksta sāk no jauna.
4. *Repeat Current Song* - atkārto pašreizējo ierakstu.

Režīmu var mainīt arī rindas paneļa galvenē.

5.5.3. Skaļums

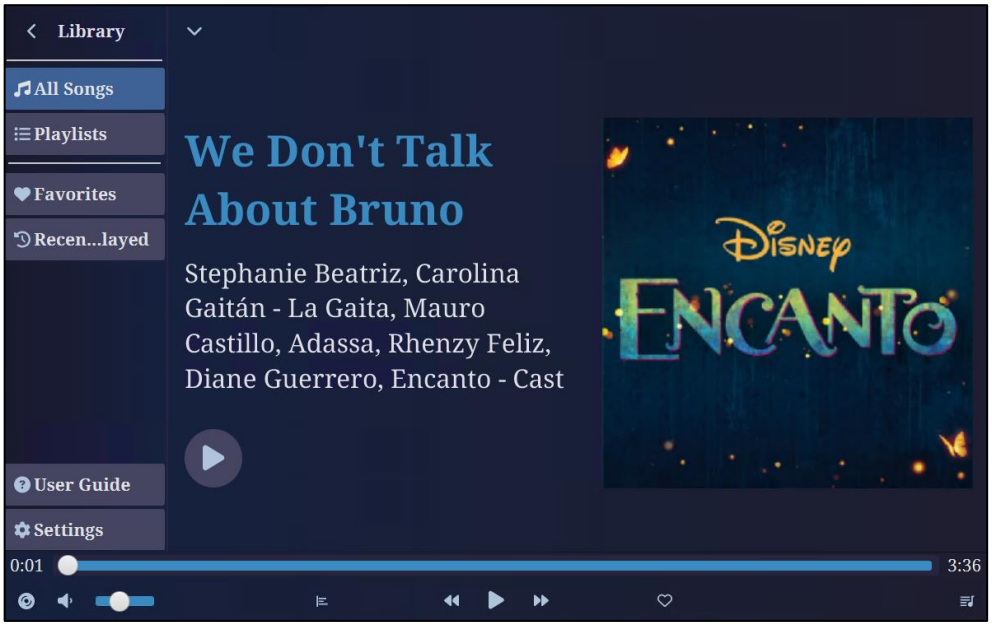
Skaļuma poga izslēdz/ieslēdz skaņu (atjaunojot iepriekšējo līmeni). Slīdnis: 0%–125%; virs 100% vērtība tiek vizuāli izcelta. Virs slīdņa var ritināt peles riteni. Skaļums tiek saglabāts starp sesijām.

5.5.4. Favorīts

Sirds ikona apakšējā panelī atzīmē vai noņem favorītu pašreiz atskaņotajam ierakstam (noklusējums: F). Tas sinhronizējas ar *Favorites* lapu un konteksta izvēlnes darbību *Toggle Favorite*.

5.5.5. Lielais atskaņošanas skats (*Big Player*)

Pārklājums ar albuma vāku, nosaukumu, izpildītāju, albumu un lielu *play/pause* pogu (skatīt 38. attēls).



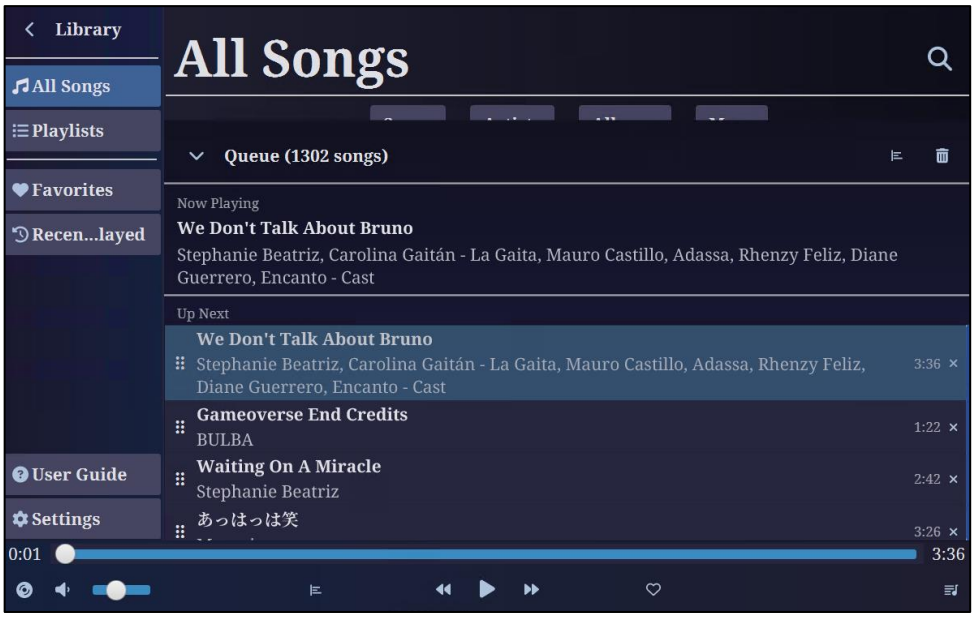
38. attēls ***Big Player* skats [Autora veidots attēls]**

Atveras ar diska pogu apakšējā panelī (kreisajā pusē), bultiņu pārklājuma augšējā kreisajā stūrī vai taustiņu B. Atskaņošanas laikā diska poga var rotēt.

5.5.6. Atskaņošanas rindas panelis (*Queue*)

Atveras ar rindas pogu apakšējā panelī (labajā pusē) vai taustiņu N.

Panelī atrodas (skatīt 39. attēls):



39. attēls ***Queue* skats [Autora veidots attēls]**

Now Playing - pašreiz atskaņotais ieraksts;

Up Next - nākamie ieraksti.

Var vilkt rindas, lai mainītu secību; noņemt atsevišķus ierakstus; notīrīt visu rindu (atkritumu tvertne galvenē); mainīt režīmu. Augšējā "vilkšanas" josla ļauj mainīt paneļa augstumu; ja tas kļūst pārāk mazs, panelis aizveras.

5.6. Bibliotēkas skati un saraksti

5.6.1. *All Songs*

Galvenā bibliotēkas lapa ar cilnēm:

Songs - visas dziesmas;

Artists - grupēts pēc izpildītāja;

Albums - grupēts pēc albuma;

Maps - grupēts pēc avota mapes.

Galvenē: meklēšanas poga, *Play All* poga (rindā pievieno visus redzamos ierakstus) un poga ritināšanai uz augšu.

5.6.2. Grupu rindas un pārklājumi

Izpildītāji, albumi, mapes un atskaņošanas saraksti parādās kā grupu rindas ar vāku un ierakstu skaitu. Noklikšķinot uz rindas vai izmantojot *Open Group* konteksta izvēlnē, atveras pārklājums ar attiecīgā grupas dziesmām. Pārklājumā pieejams *Play All* poga un *Back* poga atgriešanai.

5.6.3. Dziesmu rindas

Katrs ieraksts rādās ar vāku, nosaukumu, izpildītāju un ilgumu. Rindas elementi: *play* poga, favorīta sirds, izcelšana, ja ieraksts atskaņojas. Tās pašas darbības pieejamas apakšējā panelī un konteksta izvēlnē.

5.6.4. *Playlists*

Playlists lapā - *Create New* poga izveido jaunam sarakstam. Dziesmas pievieno ar *Add* to *Playlist* konteksta izvēlnē. Sarakstu var dzēst, labo klikšķi uz saraksta rindas (*Delete Playlist*). Dziesmu noņemšana no saraksta - atvērtā saraksta pārklājumā (*Remove from Playlist*).

5.6.5. *Favorites* un *Recently Played*

Favorites – visas dziesmas kas atzīmētas par favorītu; *Play All* poga un meklēšana galvenē.

Recently Played (virsraksts *History*) - nesen klausītie ieraksti (viens ieraksts uz dziesmu); atjaunināts atskaņošanas laikā.

5.7. Meklēšana

Meklēšanas pārklājumu atver ar meklēšanas ikonu lapu galvenēs (*All Songs, Favorites, History, Playlists*).

- Ievadiet tekstu laukā ar aizpildītāju *Search songs, artists, albums…*
- Rezultāti atjaunināts īsi pēc rakstīšanas; katrā kategorijā līdz 30 rezultātiem.
- Dziesmu rindas darbojas kā bibliotēkā; izpildītāja, albuma un atskaņošanas saraksta rindas atver grupas pārklājumu.
- *Back* poga aizver pārklājumu; x notīra meklēšanas lauku.

5.8. Konteksta izvēlnes darbības

Konteksta izvēlne atkarībā no konteksta piedāvā šādas darbības.

5.8.1. Darbības uz dziesmu rindām

Pieejamas ar labo klikšķi atverot konteksta izvēlni uz dziesmas rindas (*All Songs, Favorites, History*, meklēšanas rezultāti, grupu pārklājumi):

20. tabula **Konteksta izvēlnes darbības uz dziesmu rindām [Autora veidota tabula]**

Darbība	Apraksts
Play Track	Uzsāk atskaņošanu (kā rindas *play* poga)
Pause Track	Pauzē, ja šī rinda ir pašreiz atskaņotais ieraksts (aizstāj *Play Track*)
Play Next	Ievieto dziesmu tieši aiz pašreiz atskaņotā
Add to Queue	Pievieno rindas beigās, nemainot pašreizējo atskaņošanu
Toggle Favorite	Pievieno vai noņem no favorītiem
Add to Playlist	Apakšizvēlne ar sarakstiem; jau iekļautie - neaktīvi
Go to Artist	Apakšizvēlne; atver izpildītāja dziesmu sarakstu
Go to Album	Apakšizvēlne; atver albuma dziesmu sarakstu
Go to Source	Atver mapes (*Maps*) skatu avota direktorijai
Remove from Playlist	Tikai atvērtā atskaņošanas saraksta pārklājumā - noņem no saraksta, bet ne no bibliotēkas

5.8.2. Darbības uz grupu rindām

Darbība	Konteksts
Open Group	Izpildītājs, albums, mape vai atskaņošanas saraksts
Delete Playlist	Atskaņošanas saraksta rinda *Playlists* lapā vai meklēšanā

5.8.3. Darbības rindas panelī

Darbība	Konteksts
Scroll to Now Playing	Labo klikšķi uz rindas ieraksta *Up Next* sarakstā — ritina līdz pašreiz atskaņotajam
Go to Now Playing	Labo klikšķi uz rindas paneļa fona (ne uz ieraksta) — pārslēdz uz *All Songs* un ritina līdz atskaņotajam ierakstam

5.8.4. Vispārīga darbība

Darbība	Apraksts
Reload Library	Gandrīz visur apakšā izvēlnē (pirms tās var būt atdalītājs); pārskenē avotu mapes

5.9. Personalizācija un iestatījumi

Settings lapa (ritināma vertikāli) satur piecas sadaļas.

5.9.1. *Appearance* (izskats)

Theme izvēlne ļauj izvēlēties krāsu motīvus: *Dark-Abyss, Deep-Ember, Evening-Tea, Light-Dawn, Violet-Dusk*. Izmaiņas tiek piemērotas uzreiz un saglabātas. *Default* atgriež *Dark-Abyss.*

5.9.2. *Accessibility* (pieejamība)

Font izvēlne rāda fonta ģimenes ar priekšskatījumu; *Apply Font* saglabā, *Reset to Default* uzliek fontu uz *Noto Serif CJK*.

Font Size sadaļa, ļauj mainīt teksta izmēru no 50% - 250% (noklusējums 100%); *Apply Size / Reset* pogas.

Izmaiņas stājas spēkā pēc attiecīgās *Apply* pogas nospiešanas.

5.9.3. *File Options*

Directories - avota mapju pievienošana (+) un noņemšana (−).

Clear All Library Data poga - dzēš bibliotēkas datus datubāzē (faili diskā netiek skarti).

Starta bibliotēkas pārbaude (skatīt 5.3.4.).

5.9.4. *Key Bindings*

Tabula ar darbībām un taustiņiem. *Change Binding* poga atļauj jauna taustiņa ierakstīšanu; *Default* atiestata pārmaiņas. *Escape* atceļ modificējumu.

5.9.5. *About*

Īss prototipa apraksts, autora informācija un saite uz projekta repozitoriju.

5.10. Iebūvētā lietotāja pamācība (*User Guide*)

Programmā ir iebūvēta palīdzības lapa *User Guide*, kas atspoguļo šīs nodaļas saturu strukturētā un meklējamā formā. Tēmas grupētas sadaļās:

- *Getting Started*
- *Navigation*
- *Player Controls*
- *Pages*
- *Search*
- *Lists and Rows*

- *Settings*
- *List of Context Menu Actions*

Meklēšana notiek augšējā meklēšanas laukā, kas filtrē tēmas - atbilstošais teksts tiek izcelts. Blakus rezultātu skaitītājam atrodas pogas, kas pārvieto starp rezultātiem, *Previous / Next* vai ar taustiņiem *Up / Down* (pēc noklusējuma). *Scroll* to top poga atgriež saraksta sākumā.

Ja meklēšana neatrod neko, tiek rādīts tukšā stāvokļa ziņojums.

SECINĀJUMI UN PRIEKŠLIKUMI

Secinājumi

1. Darba gaitā tika izprasti produkta būtiskākie darbības principi un funkcionalitāte, kas nodrošināja izpratni par pilna prototipa izstrādes procesu, darbu sadalījumu un tehnoloģiju izvēles pamatojumu.
2. Konsultējoties ar klientu, tika izplānotas un izveidotas produkta struktūrskices un lietojumgadījumu diagrammas, kas paātrināja reālo skatu izstrādi. Šī plānošanas fāze aizņēma ievērojamu laiku, jo prasīja vienlaikus dizaina un funkcionālu pieeju, tomēr samazināja neskaidrības implementācijas posmā.
3. Tika veiksmīgi ieviesta trīsslāņu arhitektūra (prezentācijas, darbības un datu slānis), kas nodrošināja skaidru atbildību nošķiršanu un atviegloja koda uzturēšanu. Atsevišķu komponentu pilnveidošana notika bez būtiskas ietekmes uz pārējām sistēmas daļām.
4. Personalizācijas iespēju integrācija - pieci krāsu motīvi, CJK fontu atbalsts, pielāgojami karstie taustiņi - apstiprināja *JavaFX* piemērotību vizuāli elastīgu darbvirsmas lietotņu izstrādei. Īpaši veiksmīgs bija *ScriptFontResolver* un *ScriptFontUtils* risinājums dažādu rakstzīmju korektai attēlošanai.
5. Projekts deva praktisku pieredzi pilnā programmatūras izstrādes ciklā un papildināja autoru portfolio ar klienta apstiprinātu, daudzplatformu prototipu.

Priekšlikumi

1. Ar klienta atļauju plānots turpināt prototipa attīstību, koncentrējoties uz metadatu rediģēšanu, veiktspējas optimizāciju, paplašinātu testēšanu un izplatīšanas uzticamības uzlabojumiem (koda parakstīšana, plašāks OS atbalsts).
2. Ieteicams saglabāt esošo modulāro arhitektūru un iteratīvi papildināt funkcionalitāti, neveicot sistēmas pamatu pārstrukturēšanu.

ATSAUCES

1. ArchLinux, About Arch Linux. [Tiešsaiste]. Pieejams: https://archlinux.org/about/ [Skatīts: 30.10.2025]
2. Computer Hope, Windows. 2025. [Tiešsaiste]. Pieejams: https://www.computerhope.com/jargon/w/windows.htm [Skatīts: 30.10.2025]
3. David Nield, 9 Useful Spotify Features You Might Not Have Started Using Yet. 2019. [Tiešsaiste]. Pieejams: https://gizmodo.com/9-useful-spotify-features-you-might-not-have-started-us-1838061029 [Skatīts: 25.10.2025]
4. Joe Hindy, Google Play Store: A definitive guide for beginners. 2025. [Tiešsaiste] Pieejams: https://www.androidauthority.com/google-play-store-1093442/ [Skatīts: 30.10.2025]
5. Keychorn, What Is a Hotkey? Learn Quick Keyboard Shortcuts. 2025. [Tiešsaiste] Pieejams: https://www.keychron.com/blogs/news/what-is-a-hotkey [Skatīts: 30.10.2025]
6. MasterClass, What Is a Podcast? 5 Types of Podcasts Explained. 2021. [Tiešsaiste] Pieejams: https://www.masterclass.com/articles/what-is-a-podcast [Skatīts: 30.10.2025]
7. Monica Pawlan, What Is JavaFX?. 2013. [Tiešsaiste] Pieejams: https://docs.oracle.com/javafx/2/overview/jfxpub-overview.htm [Skatīts: 30.10.2025]
8. Sebastian Vidal, What is HUAWEI AppGallery and how it works. 2023. [Tiešsaiste] Pieejams: https://tecnobits.com/en/what-is-it-and-how-does-it-work-huawei-appgallery/ [Skatīts: 30.10.2025]
9. Wim Hoogenraad, Kas ir prototips. 2022. [Tiešsaiste]. Pieejams: https://lv.itpedia.nl/2022/03/02/wat-is-een-prototype/ [Skatīts: 30.10.2025]
10. XDA Staff, Huawei Music is offering 3-months of free trial for new subscribers in Europe. 2020. [Tiešsaiste] Pieejams: https://www.xda-developers.com/huawei-music-offering-3-months-free-trial-new-subscribers-europe/ [Skatīts: 25.10.2025]
11. Projekta *GitLab* repozitorijs. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto [Skatīts: 04.06.2026]
12. Projekta *GitLab* repozitorijas `docs` pakotne. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/tree/4482095e95c61c2f3435fc39932b98368efd3d96/docs [Skatīts: 04.06.2026]

13. Projekta *GitLab* repozitorijas `java` pakotne. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/tree/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java
14. Projekta *GitLab* repozitorijas `controllers` pakotne. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/tree/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/controllers [Skatīts: 04.06.2026]
15. Projekta *GitLab* repozitorijas `managers` pakotne. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/tree/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/managers [Skatīts: 04.06.2026]
16. Projekta *GitLab* repozitorijas `services` pakotne. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/tree/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/services [Skatīts: 04.06.2026]
17. Projekta *GitLab* repozitorijas `playback` pakotne. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/tree/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/services/playback [Skatīts: 04.06.2026]
18. Projekta *GitLab* repozitorijas `queue` pakotne. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/tree/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/services/queue [Skatīts: 04.06.2026]
19. Projekta *GitLab* repozitorijas `database` pakotne. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/tree/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/database [Skatīts: 04.06.2026]
20. Projekta *GitLab* repozitorijas `config` pakotne. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/tree/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/config [Skatīts: 04.06.2026]
21. Projekta *GitLab* repozitorijas `util` pakotne. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-

/tree/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/util [Skatīts: 04.06.2026]

22. Projekta *GitLab* repozitorijas `session` pakotne. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/tree/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/session [Skatīts: 04.06.2026]
23. Projekta *GitLab* repozitorijas `resources` pakotne. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/tree/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/resources/manicsalsa/soufone_proto [Skatīts: 04.06.2026]
24. *SongsPage.fxml* *GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/resources/manicsalsa/soufone_proto/SongsPage.fxml [Skatīts: 04.06.2026]
25. *ArtistsPage.fxml* *GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/resources/manicsalsa/soufone_proto/ArtistsPage.fxml [Skatīts: 04.06.2026]
26. *AlbumsPage.fxml* *GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/resources/manicsalsa/soufone_proto/AlbumsPage.fxml [Skatīts: 04.06.2026]
27. *MapsPage.fxml* *GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/resources/manicsalsa/soufone_proto/MapsPage.fxml [Skatīts: 04.06.2026]
28. *FavoritesPage.fxml* *GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/resources/manicsalsa/soufone_proto/FavoritesPage.fxml [Skatīts: 04.06.2026]
29. *HistoryPage.fxml* *GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/resources/manicsalsa/soufone_proto/HistoryPage.fxml [Skatīts: 04.06.2026]

30. *PlaylistPage.fxml GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/resources/manicsalsa/soufone_proto/PlaylistPage.fxml [Skatīts: 04.06.2026]
31. *BigPlayerPage.fxml GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/resources/manicsalsa/soufone_proto/BigPlayerPage.fxml [Skatīts: 04.06.2026]
32. *SearchPage.fxml GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/resources/manicsalsa/soufone_proto/SearchPage.fxml [Skatīts: 04.06.2026]
33. *GroupTrackListPage.fxml GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/resources/manicsalsa/soufone_proto/GroupTrackListPage.fxml [Skatīts: 04.06.2026]
34. *LibraryReloadOverlay.fxml GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/resources/manicsalsa/soufone_proto/LibraryReloadOverlay.fxml [Skatīts: 04.06.2026]
35. *AppearanceSectionPane.fxml GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/resources/manicsalsa/soufone_proto/AppearanceSectionPane.fxml [Skatīts: 04.06.2026]
36. *AccessibilitySectionPane.fxml GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/resources/manicsalsa/soufone_proto/AccessibilitySectionPane.fxml [Skatīts: 04.06.2026]
37. *FileOptionsSectionPane.fxml GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/resources/manicsalsa/soufone_proto/FileOptionsSectionPane.fxml [Skatīts: 04.06.2026]
38. *KeyBindingsSectionPane.fxml GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-

/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/resources/manicsalsa/soufone_proto/KeyBindingsSectionPane.fxml [Skatīts: 04.06.2026]

39. *AboutSectionPane.fxml* *GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/resources/manicsalsa/soufone_proto/AboutSectionPane.fxml [Skatīts: 04.06.2026]
40. *GroupItem.fxml* *GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/resources/manicsalsa/soufone_proto/GroupItem.fxml [Skatīts: 04.06.2026]
41. *QueueItem.fxml* *GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/resources/manicsalsa/soufone_proto/QueueItem.fxml [Skatīts: 04.06.2026]
42. Projekta *GitLab* repozitorijas `themes/colors` pakotne. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/tree/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/resources/manicsalsa/soufone_proto/themes/colors [Skatīts: 04.06.2026]
43. *img/default_cover.png* *GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/resources/manicsalsa/soufone_proto/img/default_cover.png [Skatīts: 04.06.2026]
44. Projekta *GitLab* repozitorijas `config/templates/` pakotne. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/tree/4482095e95c61c2f3435fc39932b98368efd3d96/config/templates [Skatīts: 04.06.2026]
45. *SongController.java* *GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/controllers/SongController.java [Skatīts: 04.06.2026]
46. *PlaylistController.java* *GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/controllers/PlaylistController.java [Skatīts: 04.06.2026]

47. *QueueController.java GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/controllers/QueueController.java [Skatīts: 04.06.2026]
48. *SearchController.java GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/controllers/SearchController.java [Skatīts: 04.06.2026]
49. *SettingsController.java GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/controllers/SettingsController.java [Skatīts: 04.06.2026]
50. *GuideController.java GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/controllers/GuideController.java [Skatīts: 04.06.2026]
51. *NavigationManager.java GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/managers/NavigationManager.java [Skatīts: 04.06.2026]
52. *PlayerControlsManager.java GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/managers/PlayerControlsManager.java [Skatīts: 04.06.2026]
53. *PageRegistry.java GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/managers/PageRegistry.java [Skatīts: 04.06.2026]
54. *ContentOverlayStack.java GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/managers/ContentOverlayStack.java [Skatīts: 04.06.2026]
55. *LibraryReloadManager.java GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-

/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/managers/LibraryReloadManager.java [Skatīts: 04.06.2026]

56. *HotKeyManager.java* *GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/managers/HotKeyManager.java [Skatīts: 04.06.2026]
57. *SearchOverlayOpener.java* *GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/managers/SearchOverlayOpener.java [Skatīts: 04.06.2026]
58. *ContextMenuHost.java* *GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/managers/ContextMenuHost.java [Skatīts: 04.06.2026]
59. Projekta *GitLab* repozitorijas `dao` pakotne. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/tree/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/database/dao [Skatīts: 04.06.2026]
60. *QueueService.java* *GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/services/QueueService.java [Skatīts: 04.06.2026]
61. *DirectoryService.java* *GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/services/DirectoryService.java [Skatīts: 04.06.2026]
62. *MetadataService.java* *GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/services/MetadataService.java [Skatīts: 04.06.2026]
63. *AudioLoaderService.java* *GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/services/AudioLoaderService.java [Skatīts: 04.06.2026]

64. *LibraryChangeService.java GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/services/LibraryChangeService.java [Skatīts: 04.06.2026]
65. *LibraryBootstrap.java GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/services/LibraryBootstrap.java [Skatīts: 04.06.2026]
66. *PlayHistoryService.java GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/services/PlayHistoryService.java [Skatīts: 04.06.2026]
67. *FavoriteService.java GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/services/FavoriteService.java [Skatīts: 04.06.2026]
68. *CoverImageCache.java GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/services/CoverImageCache.java [Skatīts: 04.06.2026]
69. *DatabaseSetupService.java GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/services/DatabaseSetupService.java [Skatīts: 04.06.2026]
70. *PlayerActionFeedbackService.java GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/services/PlayerActionFeedbackService.java [Skatīts: 04.06.2026]
71. *MissingFileRegistry.java GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/services/MissingFileRegistry.java [Skatīts: 04.06.2026]
72. *DatabaseHelper.java GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-

/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/database/DatabaseHelper.java [Skatīts: 04.06.2026]

73. *DatabaseSchema.java* *GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/database/DatabaseSchema.java [Skatīts: 04.06.2026]
74. *DatabaseManager.java* *GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/database/DatabaseManager.java [Skatīts: 04.06.2026]
75. Projekta *GitLab* repozitorijas `models` pakotne. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/tree/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/database/models [Skatīts: 04.06.2026]
76. *ThemeConfig.java* *GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/config/ThemeConfig.java [Skatīts: 04.06.2026]
77. *AppPreferences.java* *GitLab* repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/4482095e95c61c2f3435fc39932b98368efd3d96/src/main/java/manicsalsa/soufone_proto/config/AppPreferences.java [Skatīts: 04.06.2026]
78. Projekta *GitLab* repozitorijas `test.java` pakotne. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/tree/4482095e95c61c2f3435fc39932b98368efd3d96/src/test/java/manicsalsa/soufone_proto [Skatīts: 04.06.2026]
79. Projekta *GitLab* repozitorijas izlaidumu versijas lapa. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/releases/v1.0.0 [Skatīts: 04.06.2026]
80. *install-linux.md* GitLab repozitorijā. 2026. [Tiešsaiste] Pieejams: https://gitlab.com/SalsaManic/soufone_proto/-/blob/23e21cf9ec6bcedbb7c6d63b468eeb980010e1ee/docs/install-linux.md [Skatīts: 04.06.2026]

PIELIKUMI

1. pielikums. **pom.xml**

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>ManicSalsa</groupId>
    <artifactId>Soufone_proto</artifactId>
    <version>1.0-SNAPSHOT</version>
    <name>Soufone_proto</name>

    <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <junit.version>5.12.1</junit.version>
        <fonts.pack.url>https://gitlab.com/SalsaManic/soufone_proto/-
/releases/v1.0.0/downloads/soufone-fonts-v2.zip</fonts.pack.url>
        <!-- true for normal dev builds; installer-* profiles set this to
false -->
        <skipFontsDownload>true</skipFontsDownload>
        <fonts.zip.path>${project.build.directory}/fonts-
pack.zip</fonts.zip.path>
        <fonts.generated.dir>${project.build.directory}/generated-
fonts</fonts.generated.dir>
        <jpackage.input.dir>${project.build.directory}/jpackage-
input</jpackage.input.dir>
        <jpackage.app.name>Audio Player Prototype</jpackage.app.name>
        <jpackage.app.version>1.0.0</jpackage.app.version>
        <jpackage.vendor>ManicSalsa</jpackage.vendor>

<appimage.output.dir>${project.build.directory}/appimage</appimage.output.d
ir>
        <appimage.file.name>Audio-Player-Prototype-${jpackage.app.version}-
x86_64.AppImage</appimage.file.name>
    </properties>

    <dependencies>
        <dependency>
            <groupId>org.openjfx</groupId>
            <artifactId>javafx-controls</artifactId>
            <version>21.0.6</version>
        </dependency>
        <dependency>
            <groupId>org.openjfx</groupId>
            <artifactId>javafx-fxml</artifactId>
            <version>21.0.6</version>
        </dependency>
        <dependency>
            <groupId>org.openjfx</groupId>
            <artifactId>javafx-media</artifactId>
            <version>21.0.6</version>
        </dependency>
        <dependency>
            <groupId>com.googlecode.soundlibs</groupId>
            <artifactId>mp3spi</artifactId>
            <version>1.9.5.4</version>
        </dependency>
        <dependency>
            <groupId>net.synedra</groupId>
```

```
        <artifactId>validatorfx</artifactId>
        <version>0.6.1</version>
        <exclusions>
            <exclusion>
                <groupId>org.openjfx</groupId>
                <artifactId>*</artifactId>
            </exclusion>
        </exclusions>
    </dependency>
    <dependency>
        <groupId>org.kordamp.ikonli</groupId>
        <artifactId>ikonli-javafx</artifactId>
        <version>12.3.1</version>
    </dependency>
    <dependency>
        <groupId>org.kordamp.ikonli</groupId>
        <artifactId>ikonli-fontawesome6-pack</artifactId>
        <version>12.4.0</version>
    </dependency>
    <dependency>
        <groupId>org.kordamp.ikonli</groupId>
        <artifactId>ikonli-boxicons-pack</artifactId>
        <version>12.4.0</version>
    </dependency>
    <dependency>
        <groupId>org.kordamp.ikonli</groupId>
        <artifactId>ikonli-bootstrapicons-pack</artifactId>
        <version>12.4.0</version>
    </dependency>
    <dependency>
        <groupId>org.junit.jupiter</groupId>
        <artifactId>junit-jupiter-api</artifactId>
        <version>${junit.version}</version>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>org.junit.jupiter</groupId>
        <artifactId>junit-jupiter-engine</artifactId>
        <version>${junit.version}</version>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>org.mockito</groupId>
        <artifactId>mockito-core</artifactId>
        <version>5.6.0</version>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>com.twelvemonkeys.imageio</groupId>
        <artifactId>imageio-core</artifactId>
        <version>3.13.0</version>
    </dependency>
    <dependency>
        <groupId>com.twelvemonkeys.imageio</groupId>
        <artifactId>imageio-webp</artifactId>
        <version>3.13.0</version>
    </dependency>
    <dependency>
        <groupId>org.xerial</groupId>
        <artifactId>sqlite-jdbc</artifactId>
        <version>3.51.1.0</version>
```

```
</dependency>
<dependency>
    <groupId>com.j256.ormlite</groupId>
    <artifactId>ormlite-jdbc</artifactId>
    <version>6.1</version>
</dependency>
<dependency>
    <groupId>com.zaxxer</groupId>
    <artifactId>HikariCP</artifactId>
    <version>5.0.1</version>
</dependency>
<dependency>
    <groupId>com.fasterxml.jackson.core</groupId>
    <artifactId>jackson-databind</artifactId>
    <version>2.17.2</version>
</dependency>
<dependency>
    <groupId>com.fasterxml.jackson.core</groupId>
    <artifactId>jackson-core</artifactId>
    <version>2.17.2</version>
</dependency>
<dependency>
    <groupId>org.slf4j</groupId>
    <artifactId>slf4j-api</artifactId>
    <version>2.0.16</version>
    <scope>test</scope>
</dependency>
<dependency>
    <groupId>org.slf4j</groupId>
    <artifactId>slf4j-simple</artifactId>
    <version>2.0.16</version>
    <scope>test</scope>
</dependency>
<dependency>
    <groupId>io.github.classgraph</groupId>
    <artifactId>classgraph</artifactId>
    <version>4.8.165</version>
</dependency>
<dependency>
    <groupId>net.jthink</groupId>
    <artifactId>jaudiotagger</artifactId>
    <version>3.0.1</version>
    <scope>compile</scope>
</dependency>
<dependency>
    <groupId>com.google.code.gson</groupId>
    <artifactId>gson</artifactId>
    <version>2.13.2</version>
</dependency>
<dependency>
    <groupId>org.jflac</groupId>
    <artifactId>jflac-codec</artifactId>
    <version>1.5.2</version>
</dependency>
<dependency>
    <groupId>com.googlecode.soundlibs</groupId>
    <artifactId>vorbisspi</artifactId>
    <version>1.0.3.3</version>
</dependency>
<dependency>
    <groupId>com.googlecode.soundlibs</groupId>
```

```
            <artifactId>tritonus-share</artifactId>
            <version>0.3.7.4</version>
        </dependency>
    </dependencies>

    <build>
        <pluginManagement>
            <plugins>
                <plugin>
                    <groupId>org.codehaus.mojo</groupId>
                    <artifactId>build-helper-maven-plugin</artifactId>
                    <version>3.6.0</version>
                </plugin>
                <plugin>
                    <groupId>org.apache.maven.plugins</groupId>
                    <artifactId>maven-antrun-plugin</artifactId>
                    <version>3.1.0</version>
                </plugin>
                <plugin>
                    <groupId>org.codehaus.mojo</groupId>
                    <artifactId>exec-maven-plugin</artifactId>
                    <version>3.5.0</version>
                </plugin>
                <plugin>
                    <groupId>org.apache.maven.plugins</groupId>
                    <artifactId>maven-resources-plugin</artifactId>
                    <version>3.3.1</version>
                </plugin>
            </plugins>
        </pluginManagement>
        <plugins>
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-compiler-plugin</artifactId>
                <version>3.13.0</version>
                <configuration>
                    <source>25</source>
                    <target>25</target>
                </configuration>
            </plugin>
            <plugin>
                <groupId>org.openjfx</groupId>
                <artifactId>javafx-maven-plugin</artifactId>
                <version>0.0.8</version>
                <executions>
                    <execution>
                        <id>default-cli</id>
                        <configuration>

<mainClass>manicsalsa.soufone_proto/manicsalsa.soufone_proto.Launcher</main
Class>
                            <launcher>app</launcher>
                            <jlinkZipName>app</jlinkZipName>
                            <jlinkImageName>app</jlinkImageName>
                            <noManPages>true</noManPages>
                            <stripDebug>true</stripDebug>
                            <noHeaderFiles>true</noHeaderFiles>
                        </configuration>
                    </execution>
                </executions>
            </plugin>
```

```
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-shade-plugin</artifactId>
                <version>3.5.0</version>
                <executions>
                    <execution>
                        <phase>package</phase>
                        <goals>
                            <goal>shade</goal>
                        </goals>
                        <configuration>
                            <transformers>
                                <!-- Merge META-INF/services files -->
                                <transformer
implementation="org.apache.maven.plugins.shade.resource.ServicesResourceTra
nsformer"/>
                                <!-- Add main class to manifest -->
                                <transformer
implementation="org.apache.maven.plugins.shade.resource.ManifestResourceTra
nsformer">

<mainClass>manicsalsa.soufone_proto.Launcher</mainClass>
                                </transformer>
                            </transformers>
                            <filters>
                                <filter>
                                    <artifact>*:*</artifact>
                                    <excludes>
                                        <exclude>module-
info.class</exclude>
                                    </excludes>
                                </filter>
                            </filters>
                        </configuration>
                    </execution>
                </executions>
            </plugin>

            <plugin>
                <groupId>org.codehaus.mojo</groupId>
                <artifactId>build-helper-maven-plugin</artifactId>
            </plugin>

        </plugins>
    </build>
    <profiles>
        <!-- Windows self-contained runtime (no WiX required) -->
        <profile>
            <id>installer-win</id>
            <properties>
                <skipFontsDownload>false</skipFontsDownload>
            </properties>
            <build>
                <finalName>${project.artifactId}-${project.version}-
windows-installer-x64</finalName>
                <plugins>
                    <plugin>
                        <groupId>org.apache.maven.plugins</groupId>
                        <artifactId>maven-antrun-plugin</artifactId>
                        <executions>
```

```
                            <execution>
                                <id>download-and-unpack-fonts</id>
                                <phase>generate-resources</phase>
                                <goals>
                                    <goal>run</goal>
                                </goals>
                                <configuration>
                                    <skip>${skipFontsDownload}</skip>
                                    <target>
                                        <delete
dir="${fonts.generated.dir}/manicsalsa/soufone_proto/themes/fonts"/>
                                        <mkdir
dir="${fonts.generated.dir}/manicsalsa/soufone_proto/themes/fonts"/>
                                        <echo message="Downloading font
pack from ${fonts.pack.url}"/>
                                        <get src="${fonts.pack.url}"
dest="${fonts.zip.path}" usetimestamp="true" skipexisting="true"/>
                                        <unzip src="${fonts.zip.path}"
dest="${fonts.generated.dir}/manicsalsa/soufone_proto/themes/fonts"
overwrite="true"/>
                                    </target>
                                </configuration>
                            </execution>
                            <execution>
                                <id>stage-jpackage-input-win-app-image</id>
                                <phase>package</phase>
                                <goals>
                                    <goal>run</goal>
                                </goals>
                                <configuration>
                                    <target>
                                        <delete
dir="${jpackage.input.dir}"/>
                                        <mkdir
dir="${jpackage.input.dir}"/>
                                        <copy
file="${project.build.directory}/${project.build.finalName}.jar"
todir="${jpackage.input.dir}"/>
                                    </target>
                                </configuration>
                            </execution>
                        </executions>
                    </plugin>
                    <plugin>
                        <groupId>org.codehaus.mojo</groupId>
                        <artifactId>build-helper-maven-plugin</artifactId>
                        <executions>
                            <execution>
                                <id>add-downloaded-font-resources</id>
                                <phase>generate-resources</phase>
                                <goals>
                                    <goal>add-resource</goal>
                                </goals>
                                <configuration>
                                    <resources>
                                        <resource>

<directory>${fonts.generated.dir}</directory>
                                        </resource>
                                    </resources>
                                </configuration>
```

```
                            </execution>
                        </executions>
                    </plugin>
                    <plugin>
                        <groupId>org.codehaus.mojo</groupId>
                        <artifactId>exec-maven-plugin</artifactId>
                        <executions>
                            <execution>
                                <id>jpackage-win-app-image</id>
                                <phase>package</phase>
                                <goals>
                                    <goal>exec</goal>
                                </goals>
                                <configuration>
                                    <executable>jpackage</executable>
                                    <arguments>
                                        <argument>--type</argument>
                                        <argument>app-image</argument>
                                        <argument>--dest</argument>

<argument>${project.build.directory}/installer-win</argument>
                                        <argument>--name</argument>

<argument>${jpackage.app.name}</argument>
                                        <argument>--app-version</argument>

<argument>${jpackage.app.version}</argument>
                                        <argument>--vendor</argument>

<argument>${jpackage.vendor}</argument>
                                        <argument>--input</argument>

<argument>${jpackage.input.dir}</argument>
                                        <argument>--main-jar</argument>

<argument>${project.build.finalName}.jar</argument>
                                        <argument>--main-class</argument>

<argument>manicsalsa.soufone_proto.Launcher</argument>
                                        <argument>--icon</argument>

<argument>${project.basedir}/music_icon.ico</argument>
                                        <argument>--description</argument>

<argument>${jpackage.app.name}</argument>
                                    </arguments>
                                </configuration>
                            </execution>
                        </executions>
                    </plugin>
                </plugins>
            </build>
        </profile>

        <!-- Windows MSI wizard (requires WiX 3+ on PATH) -->
        <profile>
            <id>installer-win-msi</id>
            <properties>
                <skipFontsDownload>false</skipFontsDownload>
            </properties>
            <build>
```

```
                <finalName>${project.artifactId}-${project.version}-
windows-installer-x64</finalName>
                <plugins>
                    <plugin>
                        <groupId>org.apache.maven.plugins</groupId>
                        <artifactId>maven-antrun-plugin</artifactId>
                        <executions>
                            <execution>
                                <id>download-and-unpack-fonts</id>
                                <phase>generate-resources</phase>
                                <goals>
                                    <goal>run</goal>
                                </goals>
                                <configuration>
                                    <skip>${skipFontsDownload}</skip>
                                    <target>
                                        <delete
dir="${fonts.generated.dir}/manicsalsa/soufone_proto/themes/fonts"/>
                                        <mkdir
dir="${fonts.generated.dir}/manicsalsa/soufone_proto/themes/fonts"/>
                                        <echo message="Downloading font
pack from ${fonts.pack.url}"/>
                                        <get src="${fonts.pack.url}"
dest="${fonts.zip.path}" usetimestamp="true" skipexisting="true"/>
                                        <unzip src="${fonts.zip.path}"
dest="${fonts.generated.dir}/manicsalsa/soufone_proto/themes/fonts"
overwrite="true"/>
                                    </target>
                                </configuration>
                            </execution>
                            <execution>
                                <id>stage-jpackage-input-win-msi</id>
                                <phase>package</phase>
                                <goals>
                                    <goal>run</goal>
                                </goals>
                                <configuration>
                                    <target>
                                        <delete
dir="${jpackage.input.dir}"/>
                                        <mkdir
dir="${jpackage.input.dir}"/>
                                        <copy
file="${project.build.directory}/${project.build.finalName}.jar"
todir="${jpackage.input.dir}"/>
                                    </target>
                                </configuration>
                            </execution>
                        </executions>
                    </plugin>
                    <plugin>
                        <groupId>org.codehaus.mojo</groupId>
                        <artifactId>build-helper-maven-plugin</artifactId>
                        <executions>
                            <execution>
                                <id>add-downloaded-font-resources</id>
                                <phase>generate-resources</phase>
                                <goals>
                                    <goal>add-resource</goal>
                                </goals>
                                <configuration>
```

```
                                    <resources>
                                        <resource>

<directory>${fonts.generated.dir}</directory>
                                        </resource>
                                    </resources>
                                </configuration>
                            </execution>
                        </executions>
                    </plugin>
                    <plugin>
                        <groupId>org.codehaus.mojo</groupId>
                        <artifactId>exec-maven-plugin</artifactId>
                        <executions>
                            <execution>
                                <id>jpackage-win-msi</id>
                                <phase>package</phase>
                                <goals>
                                    <goal>exec</goal>
                                </goals>
                                <configuration>
                                    <executable>jpackage</executable>
                                    <arguments>
                                        <argument>--type</argument>
                                        <argument>msi</argument>
                                        <argument>--dest</argument>

<argument>${project.build.directory}/installer-win-msi</argument>
                                        <argument>--name</argument>

<argument>${jpackage.app.name}</argument>
                                        <argument>--app-version</argument>

<argument>${jpackage.app.version}</argument>
                                        <argument>--vendor</argument>

<argument>${jpackage.vendor}</argument>
                                        <argument>--input</argument>

<argument>${jpackage.input.dir}</argument>
                                        <argument>--main-jar</argument>

<argument>${project.build.finalName}.jar</argument>
                                        <argument>--main-class</argument>

<argument>manicsalsa.soufone_proto.Launcher</argument>
                                        <argument>--icon</argument>

<argument>${project.basedir}/music_icon.ico</argument>
                                        <argument>--win-menu</argument>
                                        <argument>--win-shortcut</argument>
                                        <argument>--win-dir-
chooser</argument>
                                        <argument>--description</argument>

<argument>${jpackage.app.name}</argument>
                                    </arguments>
                                </configuration>
                            </execution>
                        </executions>
                    </plugin>
```

```
                </plugins>
            </build>
        </profile>

        <!-- Linux self-contained app-image (jpackage; build on Linux x64)
-->
        <profile>
            <id>installer-linux</id>
            <properties>
                <skipFontsDownload>false</skipFontsDownload>
            </properties>
            <build>
                <finalName>${project.artifactId}-${project.version}-linux-
installer-x64</finalName>
                <plugins>
                    <plugin>
                        <groupId>org.apache.maven.plugins</groupId>
                        <artifactId>maven-antrun-plugin</artifactId>
                        <executions>
                            <execution>
                                <id>download-and-unpack-fonts</id>
                                <phase>generate-resources</phase>
                                <goals>
                                    <goal>run</goal>
                                </goals>
                                <configuration>
                                    <skip>${skipFontsDownload}</skip>
                                    <target>
                                        <delete
dir="${fonts.generated.dir}/manicsalsa/soufone_proto/themes/fonts"/>
                                        <mkdir
dir="${fonts.generated.dir}/manicsalsa/soufone_proto/themes/fonts"/>
                                        <echo message="Downloading font
pack from ${fonts.pack.url}"/>
                                        <get src="${fonts.pack.url}"
dest="${fonts.zip.path}" usetimestamp="true" skipexisting="true"/>
                                        <unzip src="${fonts.zip.path}"
dest="${fonts.generated.dir}/manicsalsa/soufone_proto/themes/fonts"
overwrite="true"/>
                                    </target>
                                </configuration>
                            </execution>
                            <execution>
                                <id>stage-jpackage-input-linux-app-
image</id>
                                <phase>package</phase>
                                <goals>
                                    <goal>run</goal>
                                </goals>
                                <configuration>
                                    <target>
                                        <delete
dir="${jpackage.input.dir}"/>
                                        <mkdir
dir="${jpackage.input.dir}"/>
                                        <copy
file="${project.build.directory}/${project.build.finalName}.jar"
todir="${jpackage.input.dir}"/>
                                    </target>
                                </configuration>
                            </execution>
```

```
                        </executions>
                    </plugin>
                    <plugin>
                        <groupId>org.codehaus.mojo</groupId>
                        <artifactId>build-helper-maven-plugin</artifactId>
                        <executions>
                            <execution>
                                <id>add-downloaded-font-resources</id>
                                <phase>generate-resources</phase>
                                <goals>
                                    <goal>add-resource</goal>
                                </goals>
                                <configuration>
                                    <resources>
                                        <resource>

<directory>${fonts.generated.dir}</directory>
                                        </resource>
                                    </resources>
                                </configuration>
                            </execution>
                        </executions>
                    </plugin>
                    <plugin>
                        <groupId>org.codehaus.mojo</groupId>
                        <artifactId>exec-maven-plugin</artifactId>
                        <executions>
                            <execution>
                                <id>jpackage-linux-app-image</id>
                                <phase>package</phase>
                                <goals>
                                    <goal>exec</goal>
                                </goals>
                                <configuration>
                                    <executable>jpackage</executable>
                                    <arguments>
                                        <argument>--type</argument>
                                        <argument>app-image</argument>
                                        <argument>--dest</argument>

<argument>${project.build.directory}/installer-linux</argument>
                                        <argument>--name</argument>

<argument>${jpackage.app.name}</argument>
                                        <argument>--app-version</argument>

<argument>${jpackage.app.version}</argument>
                                        <argument>--vendor</argument>

<argument>${jpackage.vendor}</argument>
                                        <argument>--input</argument>

<argument>${jpackage.input.dir}</argument>
                                        <argument>--main-jar</argument>

<argument>${project.build.finalName}.jar</argument>
                                        <argument>--main-class</argument>

<argument>manicsalsa.soufone_proto.Launcher</argument>
                                        <argument>--icon</argument>
```

```
<argument>${project.basedir}/music_icon.png</argument>
                                        <argument>--description</argument>

<argument>${jpackage.app.name}</argument>
                                    </arguments>
                                </configuration>
                            </execution>
                        </executions>
                    </plugin>
                </plugins>
            </build>
        </profile>

        <!-- Linux AppImage (requires installer-linux output + appimagetool
on PATH) -->
        <profile>
            <id>installer-linux-appimage</id>
            <properties>
                <skipFontsDownload>false</skipFontsDownload>
            </properties>
            <build>
                <finalName>${project.artifactId}-${project.version}-linux-
installer-x64</finalName>
                <plugins>
                    <plugin>
                        <groupId>org.apache.maven.plugins</groupId>
                        <artifactId>maven-antrun-plugin</artifactId>
                        <executions>
                            <execution>
                                <id>download-and-unpack-fonts</id>
                                <phase>generate-resources</phase>
                                <goals>
                                    <goal>run</goal>
                                </goals>
                                <configuration>
                                    <skip>${skipFontsDownload}</skip>
                                    <target>
                                        <delete
dir="${fonts.generated.dir}/manicsalsa/soufone_proto/themes/fonts"/>
                                        <mkdir
dir="${fonts.generated.dir}/manicsalsa/soufone_proto/themes/fonts"/>
                                        <echo message="Downloading font
pack from ${fonts.pack.url}"/>
                                        <get src="${fonts.pack.url}"
dest="${fonts.zip.path}" usetimestamp="true" skipexisting="true"/>
                                        <unzip src="${fonts.zip.path}"
dest="${fonts.generated.dir}/manicsalsa/soufone_proto/themes/fonts"
overwrite="true"/>
                                    </target>
                                </configuration>
                            </execution>
                            <execution>
                                <id>stage-jpackage-input-linux-
appimage</id>
                                <phase>package</phase>
                                <goals>
                                    <goal>run</goal>
                                </goals>
                                <configuration>
                                    <target>
```

```
                                        <delete
dir="${jpackage.input.dir}"/>
                                        <mkdir
dir="${jpackage.input.dir}"/>
                                        <copy
file="${project.build.directory}/${project.build.finalName}.jar"
todir="${jpackage.input.dir}"/>
                                    </target>
                                </configuration>
                            </execution>
                        </executions>
                    </plugin>
                    <plugin>
                        <groupId>org.codehaus.mojo</groupId>
                        <artifactId>build-helper-maven-plugin</artifactId>
                        <executions>
                            <execution>
                                <id>add-downloaded-font-resources</id>
                                <phase>generate-resources</phase>
                                <goals>
                                    <goal>add-resource</goal>
                                </goals>
                                <configuration>
                                    <resources>
                                        <resource>

<directory>${fonts.generated.dir}</directory>
                                        </resource>
                                    </resources>
                                </configuration>
                            </execution>
                        </executions>
                    </plugin>
                    <plugin>
                        <groupId>org.codehaus.mojo</groupId>
                        <artifactId>exec-maven-plugin</artifactId>
                        <executions>
                            <execution>
                                <id>jpackage-linux-appimage-base</id>
                                <phase>package</phase>
                                <goals>
                                    <goal>exec</goal>
                                </goals>
                                <configuration>
                                    <executable>jpackage</executable>
                                    <arguments>
                                        <argument>--type</argument>
                                        <argument>app-image</argument>
                                        <argument>--dest</argument>

<argument>${project.build.directory}/installer-linux</argument>
                                        <argument>--name</argument>

<argument>${jpackage.app.name}</argument>
                                        <argument>--app-version</argument>

<argument>${jpackage.app.version}</argument>
                                        <argument>--vendor</argument>

<argument>${jpackage.vendor}</argument>
                                        <argument>--input</argument>
```

```
<argument>${jpackage.input.dir}</argument>
                                        <argument>--main-jar</argument>

<argument>${project.build.finalName}.jar</argument>
                                        <argument>--main-class</argument>

<argument>manicsalsa.soufone_proto.Launcher</argument>
                                        <argument>--icon</argument>

<argument>${project.basedir}/music_icon.png</argument>
                                        <argument>--description</argument>

<argument>${jpackage.app.name}</argument>
                                    </arguments>
                                </configuration>
                            </execution>
                            <execution>
                                <id>build-appimage</id>
                                <phase>package</phase>
                                <goals>
                                    <goal>exec</goal>
                                </goals>
                                <configuration>

<executable>${project.basedir}/scripts/build-appimage.sh</executable>
                                    <arguments>

<argument>${project.build.directory}/installer-linux</argument>

<argument>${jpackage.app.name}</argument>

<argument>${jpackage.app.version}</argument>

<argument>${appimage.output.dir}</argument>

<argument>${appimage.file.name}</argument>

<argument>${project.basedir}/music_icon.png</argument>
                                    </arguments>
                                </configuration>
                            </execution>
                        </executions>
                    </plugin>
                </plugins>
            </build>
        </profile>
    </profiles>
</project>
```

2. pielikums. **Launcher.java**

```
package manicsalsa.soufone_proto;

import javafx.application.Application;

public class Launcher {
    static void main(String[] args) {
        Application.launch(SoufoneApplication.class, args);
    }
}
```

3. pielikums. **SoufoneApplication.java**

```
package manicsalsa.soufone_proto;

import javafx.application.Application;
import javafx.application.Platform;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;
import manicsalsa.soufone_proto.config.ThemeConfig;
import manicsalsa.soufone_proto.controllers.MainLayoutController;
import manicsalsa.soufone_proto.database.DatabaseHelper;
import manicsalsa.soufone_proto.database.dao.AudioDAO;
import manicsalsa.soufone_proto.database.dao.PlayerStateDAO;
import manicsalsa.soufone_proto.database.dao.QueueItemDAO;
import manicsalsa.soufone_proto.services.AudioPlayerService;
import manicsalsa.soufone_proto.services.CoverImageCache;
import manicsalsa.soufone_proto.services.DatabaseSetupService;
import manicsalsa.soufone_proto.services.LibraryBootstrap;
import manicsalsa.soufone_proto.services.QueueService;

import java.io.IOException;
import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;

/**
 * JavaFX {@link Application}: open DB, restore {@link QueueService}, show
 * {@code MainLayout.fxml}, then load the songs page.
 */
public class SoufoneApplication extends Application {

    private static final Logger LOGGER =
Logger.getLogger(SoufoneApplication.class.getName());

    private DatabaseSetupService databaseSetupService;

    /** Prepares {@link DatabaseSetupService} and quiets noisy third-party
loggers. */
    @Override
    public void init() {
        databaseSetupService = DatabaseSetupService.getInstance();
        LOGGER.info("Application initialization started");
        suppressNoisyLoggers();
    }

    /** Opens the DB, restores queue state, and shows the main window. */
    @Override
    public void start(Stage primaryStage) throws IOException, SQLException
{
        LOGGER.info("Starting application...");

        if (!handleDatabaseInitialization()) {
            Platform.exit();
            return;
        }

        DatabaseHelper dbHelper = DatabaseHelper.getInstance();
        AudioDAO audioDAO = dbHelper.getAudioDAO();
        QueueItemDAO queueItemDAO = dbHelper.getQueueItemDAO();
        PlayerStateDAO playerStateDAO = dbHelper.getPlayerStateDAO();
```

```
        QueueService queueService = QueueService.getInstance();
        queueService.init(queueItemDAO, playerStateDAO, audioDAO);
        queueService.restore();

        LibraryBootstrap.getInstance().start(dbHelper);

        FXMLLoader fxmlLoader = new FXMLLoader(

SoufoneApplication.class.getResource("/manicsalsa/soufone_proto/MainLayout.
fxml"));
        installControllerFactory(fxmlLoader, audioDAO);

        Parent root = fxmlLoader.load();
        Scene scene = new Scene(root, 1200, 800);

        ThemeConfig.getInstance().applyTheme(scene);

        primaryStage.setTitle("Soufone Prototype");
        primaryStage.setScene(scene);
        primaryStage.show();

        LOGGER.info("Main UI loaded successfully");
        // Song page init runs from FXML; avoid duplicating heavy setup
here.
    }

    /** Shuts down caches and background resources. */
    @Override
    public void stop() {
        LOGGER.info("Application shutting down");
        LibraryBootstrap.getInstance().shutdown();
        CoverImageCache.getInstance().shutdown();
        QueueService.getInstance().shutdown();
        AudioPlayerService.getInstance().shutdown();
    }

    /** Lowers log level for tag parsing and CSS helper noise during
startup. */
    private void suppressNoisyLoggers() {
        Logger.getLogger("org.jaudiotagger").setLevel(Level.SEVERE);

Logger.getLogger("javafx.scene.CssStyleHelper").setLevel(Level.SEVERE);
    }

    /** Creates schema on first run; false when init throws. */
    private boolean handleDatabaseInitialization() {
        try {
            if (!databaseSetupService.isDatabaseInitialized()) {
                LOGGER.info("Database not initialized, initializing
now...");
                databaseSetupService.initializeDatabase();
            } else {
                LOGGER.info("Database already initialized");
            }
            return true;
        } catch (SQLException e) {
            LOGGER.log(Level.SEVERE, "Failed to initialize database", e);
            return false;
        }
    }
```

```
    /**
     * Injects {@link MainLayoutController} with {@link AudioDAO}; other
FXML types use
     * no-arg constructors.
     */
    private void installControllerFactory(FXMLLoader fxmlLoader, AudioDAO
audioDAO) {
        fxmlLoader.setControllerFactory(param -> {
            if (param == MainLayoutController.class) {
                return new MainLayoutController(audioDAO);
            }
            try {
                return param.getDeclaredConstructor().newInstance();
            } catch (Exception e) {
                throw new RuntimeException(e);
            }
        });
    }
}
```

4. pielikums. **MainLayoutController.java**

```
package manicsalsa.soufone_proto.controllers;

import javafx.application.Platform;
import javafx.beans.binding.Bindings;
import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.control.*;
import javafx.scene.layout.*;
import manicsalsa.soufone_proto.database.dao.AudioDAO;
import manicsalsa.soufone_proto.managers.ContextMenuHost;
import manicsalsa.soufone_proto.managers.LibraryReloadManager;
import manicsalsa.soufone_proto.managers.NavPanelView;
import manicsalsa.soufone_proto.managers.NavigationManager;
import manicsalsa.soufone_proto.managers.PageRegistry;
import manicsalsa.soufone_proto.session.PageLifecycle;
import manicsalsa.soufone_proto.session.PageSessionAware;
import javafx.scene.layout.Region;
import manicsalsa.soufone_proto.managers.PlayerControlsBarView;
import manicsalsa.soufone_proto.managers.PlayerControlsManager;
import manicsalsa.soufone_proto.services.AudioPlayerService;
import org.kordamp.ikonli.javafx.FontIcon;

import java.io.IOException;
import java.net.URL;
import java.sql.SQLException;
import java.util.ResourceBundle;
import java.util.function.Consumer;
import java.util.logging.Level;
import java.util.logging.Logger;

/**
 * Application shell ({@code MainLayout.fxml}): content {@link #loadPage},
global context menu,
 * and library reload. Injects nav/player FXML into {@link
NavigationManager} and
 * {@link PlayerControlsManager} ({@link PlayerControlsBarView}) once at
startup.
 */
```

```
public class MainLayoutController implements Initializable {

    private static final Logger LOGGER =
Logger.getLogger(MainLayoutController.class.getName());

    /** Minimum content area when the shell shows horizontal/vertical
scroll bars. */
    public static final double CONTENT_MIN_WIDTH = 400;
    public static final double CONTENT_MIN_HEIGHT = 400;

    private static final String VIEWPORT_CONSTRAINED_KEY =
"soufone.viewportConstrained";
    private static final String SCROLL_MODE_KEY = "soufone.scrollMode";

    /** How a page participates in the main content scroll pane. */
    private enum PageScrollMode {
        /** Fills viewport; lists scroll inside (All Songs, Guide, etc.).
*/
        FILL_VIEWPORT,
        /** Natural content height; main pane scrolls vertically
(Settings). */
        INTRINSIC_HEIGHT
    }

    public  BorderPane navBar;
    public  VBox       bottomButtonsVBox;
    public  VBox       topButtonsVBox;
    public  VBox       navVBox;
    public  ScrollPane navScrollPane;

    @FXML private SplitPane    mainSplitPane;
    @FXML private Label        libraryLabel;
    @FXML private Button       navToggleButton;
    @FXML private ToggleButton AllSongsButton;
    @FXML private ToggleButton PlaylistsButton;
    @FXML private ToggleButton FavoritesButton;
    @FXML private ToggleButton HistoryButton;
    @FXML private ToggleButton GuideButton;
    @FXML private ToggleButton SettingsButton;
    @FXML private StackPane    contentPane;
    @FXML private StackPane    contentOverlayPane;
    @FXML private BorderPane   mainContainer;

    @FXML private ToggleButton bottomOpenBigPlayerButton;
    @FXML private Button       bottomPrevButton;
    @FXML private Button       bottomPlayPauseButton;
    @FXML private Button       bottomNextButton;
    @FXML private Button       bottomQueueButton;
    @FXML private Slider  bottomProgressBar;
    @FXML private Label        bottomCurrentTimeLabel;
    @FXML private Label        bottomTotalTimeLabel;
    @FXML private Button       bottomVolumeButton;
    @FXML private HBox         bottomVolumeContainer;
    @FXML private Label        bottomVolumePercentLabel;
    @FXML private Slider       bottomVolumeSlider;
    @FXML private FontIcon     bottomVolumeIcon;
    @FXML private FontIcon     bottomPlayPauseIcon;
    public         Button      playModeButton;
    public         Button      bottomFavoriteButton;

    private final AudioDAO           audioDAO;
```

```
    private final AudioPlayerService playerService =
AudioPlayerService.getInstance();

    private PlayerControlsManager playerControlsManager;
    private NavigationManager navigationManager;
    private LibraryReloadManager libraryReloadManager;
    private PageRegistry pageRegistry;

    private Object currentPageController;
    private String currentFxmlFile;
    private Consumer<SongController> pendingLibraryReveal;

    public MainLayoutController(AudioDAO audioDAO) {
        this.audioDAO = audioDAO;
    }

    @Override
    public void initialize(URL location, ResourceBundle resources) {
        navigationManager = new NavigationManager(
                new NavPanelView(
                        mainSplitPane, mainContainer, contentPane,
                        navVBox, navBar, navScrollPane, topButtonsVBox,
bottomButtonsVBox,
                        navToggleButton, libraryLabel,
                        AllSongsButton, PlaylistsButton, FavoritesButton,
HistoryButton,
                        GuideButton, SettingsButton),
                this);
        navigationManager.init();

        playerControlsManager = new PlayerControlsManager(
                new PlayerControlsBarView(
                        bottomOpenBigPlayerButton,
                        bottomPrevButton,
                        bottomPlayPauseButton,
                        bottomNextButton,
                        bottomQueueButton,
                        bottomProgressBar,
                        bottomCurrentTimeLabel,
                        bottomTotalTimeLabel,
                        bottomPlayPauseIcon,
                        playModeButton,
                        bottomFavoriteButton,
                        bottomVolumeButton,
                        bottomVolumeContainer,
                        bottomVolumePercentLabel,
                        bottomVolumeSlider,
                        bottomVolumeIcon,
                        contentOverlayPane),
                playerService,
                audioDAO,
                this);
        playerControlsManager.init();

        libraryReloadManager = new LibraryReloadManager(contentOverlayPane,
mainContainer);
        libraryReloadManager.init();
        libraryReloadManager.checkOnStartupIfEnabled();

        installGlobalContextMenu();
```

```
        playerService.installShutdownHook();

        pageRegistry = new PageRegistry(getClass());
        loadPage("AllSongsPage.fxml");
    }

    /** Swaps the center content page; captures session state for the
outgoing page. */
    public void loadPage(String fxmlFile) {
        try {
            if (playerControlsManager != null &&
playerControlsManager.isBigPlayerVisible())
                playerControlsManager.hideBigPlayer();

            detachCurrentPage();

            if (!"AllSongsPage.fxml".equals(fxmlFile)) {
                pendingLibraryReveal = null;
            }

            if (!fxmlFile.equals("BigPlayerPage.fxml")) {
                PageRegistry.CachedPage cached =
pageRegistry.getOrLoad(fxmlFile);
                Region page = cached.root();
                currentPageController = cached.controller();
                currentFxmlFile = fxmlFile;

                PageScrollMode scrollMode = scrollModeForFxml(fxmlFile);
                configureContentScrollPaneForMode(scrollMode);
                applyContentViewportConstraints(page, scrollMode);

                StackPane.setAlignment(page, javafx.geometry.Pos.TOP_LEFT);
                contentPane.getChildren().add(page);

                if (currentPageController instanceof PageLifecycle
lifecycle) {
                    lifecycle.onPageShown();
                }
                if (currentPageController instanceof PageSessionAware
aware) {
                    aware.restoreSessionState();
                }

                scheduleFillPageLayoutPass(scrollMode, page);
                runPendingLibraryRevealIfReady();
            }
        } catch (IOException e) {
            LOGGER.log(Level.SEVERE, "Failed to load page: " + fxmlFile,
e);
        }
    }

    private void runPendingLibraryRevealIfReady() {
        if (pendingLibraryReveal == null ||
!"AllSongsPage.fxml".equals(currentFxmlFile)) {
            return;
        }
        if (!(currentPageController instanceof SongController
songController)) {
            return;
        }
```

```
        Consumer<SongController> reveal = pendingLibraryReveal;
        pendingLibraryReveal = null;
        Platform.runLater(() -> reveal.accept(songController));
    }

    private void detachCurrentPage() {
        captureLeavingPageState();
        if (currentPageController instanceof PageLifecycle lifecycle) {
            lifecycle.shutdown();
        }
        contentPane.getChildren().clear();
    }

    private static PageScrollMode scrollModeForFxml(String fxmlFile) {
        if ("SettingsPage.fxml".equals(fxmlFile)) {
            return PageScrollMode.INTRINSIC_HEIGHT;
        }
        return PageScrollMode.FILL_VIEWPORT;
    }

    /**
     * List pages: fit viewport height, list scrolls inside, no main
vertical bar.
     * Settings: natural height with main-pane vertical scroll.
     */
    private void configureContentScrollPaneForMode(PageScrollMode mode) {
        ScrollPane contentScrollPane = getScrollPane();
        contentScrollPane.setFitToWidth(false);
        if (mode == PageScrollMode.FILL_VIEWPORT) {
            contentScrollPane.setFitToHeight(true);

contentScrollPane.setVbarPolicy(ScrollPane.ScrollBarPolicy.NEVER);
        } else {
            contentScrollPane.setFitToHeight(false);

contentScrollPane.setVbarPolicy(ScrollPane.ScrollBarPolicy.AS_NEEDED);
        }

contentScrollPane.setHbarPolicy(ScrollPane.ScrollBarPolicy.AS_NEEDED);
    }

    /**
     * Applies unified min width ({@value #CONTENT_MIN_WIDTH}) and page-
type height behavior.
     * Fill pages match the content host height; Settings uses intrinsic
height for main scroll.
     */
    private void applyContentViewportConstraints(Region page,
PageScrollMode mode) {
        PageScrollMode existing = (PageScrollMode)
page.getProperties().get(SCROLL_MODE_KEY);
        if
(Boolean.TRUE.equals(page.getProperties().get(VIEWPORT_CONSTRAINED_KEY))
                && mode == existing) {
            return;
        }

        page.prefWidthProperty().unbind();
        page.prefHeightProperty().unbind();
        page.maxHeightProperty().unbind();
```

```
        page.setMinWidth(CONTENT_MIN_WIDTH);

        ScrollPane contentScrollPane = getScrollPane();
        page.prefWidthProperty().bind(
                Bindings.createDoubleBinding(
                        () -> Math.max(page.getMinWidth(),
contentScrollPane.getViewportBounds().getWidth()),
                        contentScrollPane.viewportBoundsProperty(),
page.minWidthProperty()
                )
        );

        if (mode == PageScrollMode.FILL_VIEWPORT) {
            page.setMinHeight(0);
            page.prefHeightProperty().bind(contentPane.heightProperty());
            page.maxHeightProperty().bind(contentPane.heightProperty());
        } else {
            page.setMinHeight(CONTENT_MIN_HEIGHT);
            page.setPrefHeight(Region.USE_COMPUTED_SIZE);
        }

        page.getProperties().put(VIEWPORT_CONSTRAINED_KEY, Boolean.TRUE);
        page.getProperties().put(SCROLL_MODE_KEY, mode);
    }

    /** Re-layout fill pages after the scene/viewport has real dimensions
(fixes first-open list clip). */
    private void scheduleFillPageLayoutPass(PageScrollMode mode, Region
page) {
        if (mode != PageScrollMode.FILL_VIEWPORT) {
            return;
        }
        Runnable layoutPass = () -> {
            if (!contentPane.getChildren().contains(page)) {
                return;
            }
            contentPane.requestLayout();
            page.requestLayout();
        };
        ScrollPane scrollPane = getScrollPane();
        if (scrollPane.getViewportBounds().getHeight() > 0) {
            Platform.runLater(layoutPass);
        } else {
            scrollPane.viewportBoundsProperty().addListener(new
javafx.beans.value.ChangeListener<>() {
                @Override
                public void changed(
                        javafx.beans.value.ObservableValue<? extends
javafx.geometry.Bounds> obs,
                        javafx.geometry.Bounds oldBounds,
                        javafx.geometry.Bounds bounds) {
                    if (bounds.getHeight() > 0) {

scrollPane.viewportBoundsProperty().removeListener(this);
                        Platform.runLater(layoutPass);
                    }
                }
            });
        }
    }
```

```
    /** Captures outgoing page session state without forcing a full layout
pass. */
    private void captureLeavingPageState() {
        if (currentPageController instanceof PageSessionAware aware) {
            aware.captureSessionState();
        }
    }

    private ScrollPane getScrollPane() {
        return (ScrollPane) ((StackPane)
mainSplitPane.getItems().get(1)).getChildren().getFirst();
    }

    public void hideBigPlayer() {
        playerControlsManager.hideBigPlayer();
    }

    public void hideQueuePanel() {
        playerControlsManager.hideQueuePanel();
    }

    public void playPauseWithFeedback() {
        if (playerControlsManager != null) {
            playerControlsManager.playPauseWithFeedback();
        }
    }

    public void closeSearchOverlays() {
        if (playerControlsManager != null)
            playerControlsManager.closeSearchOverlays();
    }

    /** Closes group track overlays and shuts down their loader threads. */
    public void closeGroupTrackOverlay() {
        contentOverlayPane.getChildren().removeIf(node -> {
            if (node.getUserData() instanceof GroupTrackListController
ctrl) {
                ctrl.shutdown();
                return true;
            }
            return false;
        });
    }

    /** Library reload with confirmation (R / context menu). */
    public void reloadLibrary() {
        if (libraryReloadManager != null) {
            libraryReloadManager.requestReload();
        }
    }

    /** Library reload without confirmation (e.g. startup apply). */
    public void reloadLibraryNow() {
        if (libraryReloadManager != null) {
            libraryReloadManager.reloadLibraryNow();
        }
    }

    /** Navigate to All Songs and reveal a track (context menu Go to
Source, etc.). */
    public void jumpToAudioInLibrary(Integer audioId) {
```

```
        if (audioId == null) {
            return;
        }
        jumpToLibrary(songController ->
songController.revealSongById(audioId));
    }

    /** Navigate to All Songs and reveal the now-playing track. */
    public void jumpToNowPlayingInLibrary() {
        jumpToLibrary(SongController::revealCurrentSong);
    }

    private void jumpToLibrary(Consumer<SongController> revealAction) {
        hideBigPlayer();
        closeSearchOverlays();
        closeGroupTrackOverlay();
        hideQueuePanel();
        if (!"AllSongsPage.fxml".equals(currentFxmlFile)) {
            pendingLibraryReveal = revealAction;
            navigationManager.goToAllSongsPage();
            return;
        }
        runLibraryReveal(revealAction);
    }

    private void runLibraryReveal(Consumer<SongController> revealAction) {
        Platform.runLater(() -> {
            if (currentPageController instanceof SongController
songController) {
                revealAction.accept(songController);
            }
        });
    }

    private boolean canGoToNowPlaying() {
        Integer id = playerService.getCurrentAudioId();
        if (id == null) {
            return false;
        }
        if (currentPageController instanceof SongController songController
                && "AllSongsPage.fxml".equals(currentFxmlFile)) {
            return songController.isCurrentSongInAllSongsList(id);
        }
        try {
            return audioDAO.findById(id, false) != null;
        } catch (SQLException e) {
            LOGGER.log(Level.WARNING, "Could not verify current track in
library", e);
            return true;
        }
    }

    private void installGlobalContextMenu() {
        try {
            GlobalContextMenuController menuController =
GlobalContextMenuController.load();
            menuController.setOnReloadLibrary(this::reloadLibrary);

menuController.setOnGoToNowPlaying(this::jumpToNowPlayingInLibrary);
            menuController.setCanGoToNowPlaying(this::canGoToNowPlaying);
```

```
            ContextMenuHost contextMenuHost = new
ContextMenuHost(mainContainer);
            contextMenuHost.setMenu(menuController.getMenu());

contextMenuHost.setOnBeforeShow(menuController::prepareMenuForTarget);
            contextMenuHost.install();
        } catch (IOException e) {
            LOGGER.log(Level.SEVERE, "Failed to load global context menu",
e);
        }
    }
}
```

5. pielikums. **module-info.java**

```
module manicsalsa.soufone_proto {
    requires javafx.controls;
    requires javafx.fxml;
    requires org.kordamp.ikonli.javafx;
    requires java.logging;
    requires java.sql;
    requires org.xerial.sqlitejdbc;
    requires jaudiotagger;
    requires com.google.gson;
    requires javafx.media;
    requires java.desktop;
    requires io.github.classgraph;

    opens manicsalsa.soufone_proto to javafx.fxml;
    exports manicsalsa.soufone_proto;
    exports manicsalsa.soufone_proto.controllers;
    opens manicsalsa.soufone_proto.controllers to javafx.fxml;

    exports manicsalsa.soufone_proto.database;
    exports manicsalsa.soufone_proto.database.models;
    exports manicsalsa.soufone_proto.database.dao;
    exports manicsalsa.soufone_proto.services;
    exports manicsalsa.soufone_proto.services.playback;
}
```

6. pielikums. **MainLayout.fxml**

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- Shell: nav SplitPane, content overlay stack, bottom transport bar. -->
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>
<?import org.kordamp.ikonli.javafx.*?>
<?import javafx.geometry.Insets?>
<BorderPane fx:id="mainContainer" maxHeight="-Infinity" maxWidth="-
Infinity"
            prefHeight="600.0" prefWidth="1200.0"
xmlns="http://javafx.com/javafx/17.0.12"
xmlns:fx="http://javafx.com/fxml/1"

fx:controller="manicsalsa.soufone_proto.controllers.MainLayoutController">
    <center>
        <SplitPane fx:id="mainSplitPane"
dividerPositions="0.2579298831385643">
            <BorderPane fx:id="navBar" minWidth="0.0" styleClass="nav-bar"
>
```

```
            <center>
                <ScrollPane fx:id="navScrollPane" fitToWidth="true"
styleClass="nav-scroll-pane" >
                    <VBox fx:id="navVBox" styleClass="nav-pane"
spacing="5">
                        <VBox fx:id="topButtonsVBox" spacing="5">
                            <ToggleButton fx:id="AllSongsButton"
alignment="BASELINE_LEFT" maxWidth="1.7976931348623157E308" minWidth="0"
                                          mnemonicParsing="false"
prefHeight="50.0" styleClass="nav-button" text="All Songs">
                                <graphic><FontIcon iconLiteral="fas-
music" iconSize="20" /></graphic>
                            </ToggleButton>
                            <ToggleButton fx:id="PlaylistsButton"
alignment="BASELINE_LEFT" maxWidth="1.7976931348623157E308" minWidth="0"
                                          mnemonicParsing="false"
prefHeight="50.0" styleClass="nav-button" text="Playlists">
                                <graphic><FontIcon iconLiteral="fas-
list" iconSize="20" /></graphic>
                            </ToggleButton>
                            <Separator styleClass="nav-separator" />
                            <ToggleButton fx:id="FavoritesButton"
alignment="BASELINE_LEFT" maxWidth="1.7976931348623157E308" minWidth="0"
                                          mnemonicParsing="false"
prefHeight="50.0" styleClass="nav-button" text="Favorites">
                                <graphic><FontIcon iconLiteral="fas-
heart" iconSize="20" /></graphic>
                            </ToggleButton>
                            <ToggleButton fx:id="HistoryButton"
alignment="BASELINE_LEFT" maxWidth="1.7976931348623157E308" minWidth="0"
                                          mnemonicParsing="false"
prefHeight="50.0" styleClass="nav-button" text="Recently Played">
                                <graphic><FontIcon iconLiteral="fas-
history" iconSize="20" /></graphic>
                            </ToggleButton>
                        </VBox>

                        <Region VBox.vgrow="ALWAYS" />

                        <VBox fx:id="bottomButtonsVBox"
alignment="CENTER" spacing="5">
                            <ToggleButton fx:id="GuideButton"
alignment="BASELINE_LEFT" maxWidth="1.7976931348623157E308" minWidth="0"
                                          mnemonicParsing="false"
prefHeight="50.0" styleClass="nav-button" text="User Guide">
                                <graphic><FontIcon iconLiteral="fas-
question-circle" iconSize="20" /></graphic>
                            </ToggleButton>
                            <ToggleButton fx:id="SettingsButton"
alignment="BASELINE_LEFT" maxWidth="1.7976931348623157E308" minWidth="0"
                                          mnemonicParsing="false"
prefHeight="50.0" styleClass="nav-button" text="Settings">
                                <graphic><FontIcon iconLiteral="fas-
cog" iconSize="20" /></graphic>
                            </ToggleButton>
                        </VBox>
                    </VBox>
                </ScrollPane>
            </center>
            <top>
                <VBox>
```

```
                        <HBox alignment="CENTER_LEFT"
maxWidth="1.7976931348623157E308"
                              prefHeight="60.0" spacing="10"
styleClass="nav-header">
                            <Button fx:id="navToggleButton"
mnemonicParsing="false" styleClass="nav-toggle-button">
                                <graphic>
                                    <FontIcon iconLiteral="fas-chevron-
left" iconSize="20" />
                                </graphic>
                                <tooltip>
                                    <Tooltip text="Collapse"/>
                                </tooltip>
                            </Button>
                            <Label fx:id="libraryLabel"
styleClass="library-label" text="Library" />
                        </HBox>
                        <Separator styleClass="nav-separator" />
                    </VBox>
                </top>
            </BorderPane>

            <StackPane fx:id="contentOverlayPane">
                <ScrollPane fitToHeight="true" vbarPolicy="NEVER"
                            styleClass="content-scroll-pane">
                    <StackPane fx:id="contentPane"/>
                </ScrollPane>
            </StackPane>
        </SplitPane>
    </center>
    <bottom>
        <VBox spacing="5" styleClass="bottom-container">
            <padding>
                <Insets left="10" right="10" bottom="5"/>
            </padding>
            <HBox alignment="CENTER" spacing="10" maxWidth="Infinity">
                <Label fx:id="bottomCurrentTimeLabel" text="--:--" />
                <Slider fx:id="bottomProgressBar" HBox.hgrow="ALWAYS"
maxWidth="Infinity" prefHeight="8" styleClass="bottom-progress-bar"
blockIncrement="1" />
                <Label fx:id="bottomTotalTimeLabel" text="--:--" />
            </HBox>

            <GridPane hgap="15" alignment="CENTER" VBox.vgrow="NEVER">
                <columnConstraints>
                    <ColumnConstraints percentWidth="35" halignment="LEFT"
/>
                    <ColumnConstraints percentWidth="30"
halignment="CENTER" />
                    <ColumnConstraints percentWidth="35" halignment="RIGHT"
/>
                </columnConstraints>

                <HBox alignment="CENTER_LEFT" spacing="10"
GridPane.columnIndex="0">
                    <ToggleButton fx:id="bottomOpenBigPlayerButton"
styleClass="bottom-control-button">
                        <graphic><FontIcon iconLiteral="bi-disc-fill"
iconSize="20" /></graphic>
                        <tooltip><Tooltip text="Big Player
view"/></tooltip>
```

```
                    </ToggleButton>
                    <Button fx:id="bottomVolumeButton" styleClass="bottom-
control-button">
                        <graphic><FontIcon fx:id="bottomVolumeIcon"
iconLiteral="fas-volume-up" iconSize="20" /></graphic>
                        <tooltip><Tooltip text="Mute/Unmute"/></tooltip>
                    </Button>
                    <HBox fx:id="bottomVolumeContainer"
alignment="CENTER_LEFT" spacing="6">
                        <Slider fx:id="bottomVolumeSlider" max="125"
value="100" prefWidth="90" maxWidth="120"
                                blockIncrement="5" styleClass="bottom-
volume-slider" />
                        <Label fx:id="bottomVolumePercentLabel"
visible="false" managed="false"
                               styleClass="bottom-volume-label" />
                    </HBox>
                    <Region HBox.hgrow="ALWAYS"/>
                    <Button fx:id="playModeButton" styleClass="icon-
button">
                        <graphic>
                            <FontIcon iconLiteral="bx-repost"
iconSize="20"/>
                        </graphic>
                        <tooltip><Tooltip text="Shuffle" /></tooltip>
                    </Button>
                </HBox>
                <HBox alignment="CENTER" spacing="20"
GridPane.columnIndex="1">
                    <Button fx:id="bottomPrevButton" styleClass="bottom-
control-button">
                        <graphic><FontIcon iconLiteral="fas-backward"
iconSize="18" /></graphic>
                        <tooltip>
                            <Tooltip text="Previous track"/>
                        </tooltip>
                    </Button>
                    <Button fx:id="bottomPlayPauseButton"
styleClass="bottom-control-button">
                        <graphic><FontIcon fx:id="bottomPlayPauseIcon"
iconLiteral="fas-play" iconSize="25" /></graphic>
                        <tooltip>
                            <Tooltip text="Play/Pause track"/>
                        </tooltip>
                    </Button>
                    <Button fx:id="bottomNextButton" styleClass="bottom-
control-button">
                        <graphic><FontIcon iconLiteral="fas-forward"
iconSize="18" /></graphic>
                        <tooltip>
                            <Tooltip text="Next track"/>
                        </tooltip>
                    </Button>
                </HBox>

                <HBox alignment="CENTER_RIGHT" spacing="10"
GridPane.columnIndex="2">
                    <Button fx:id="bottomFavoriteButton"
styleClass="bottom-control-button">
                        <graphic><FontIcon iconLiteral="far-heart"
iconSize="18" /></graphic>
```

```
                        <tooltip><Tooltip text="Add to
favourites"/></tooltip>
                    </Button>
                    <Region HBox.hgrow="ALWAYS"/>
                    <Button fx:id="bottomQueueButton" styleClass="bottom-
control-button">
                        <graphic><FontIcon iconLiteral="bxs-playlist"
iconSize="20" /></graphic>
                        <tooltip>
                            <Tooltip text="Open queue"/>
                        </tooltip>
                    </Button>
                </HBox>
            </GridPane>
        </VBox>
    </bottom>
</BorderPane>
```

7. pielikums. **AllSongsPage.fxml**

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>
<?import org.kordamp.ikonli.javafx.*?>

<VBox xmlns="http://javafx.com/javafx/17.0.12"
      xmlns:fx="http://javafx.com/fxml/1"
      fx:controller="manicsalsa.soufone_proto.controllers.SongController"
      fx:id="rootVBox" maxHeight="Infinity" VBox.vgrow="ALWAYS">

    <VBox fx:id="headerSection" VBox.vgrow="NEVER">
        <HBox fx:id="pageHeader" alignment="CENTER" style="-fx-padding: 20
10 0 10;">
            <Label text="All Songs" styleClass="text-header" />
            <Region HBox.hgrow="ALWAYS" />
            <Button fx:id="SearchAllSongsButton" styleClass="icon-button"
mnemonicParsing="false">
                <graphic>
                    <FontIcon iconLiteral="fas-search" iconSize="30" />
                </graphic>
            </Button>
        </HBox>
        <Separator />
        <HBox fx:id="subPageTabs" alignment="CENTER" spacing="30" style="-
fx-padding: 10 0 10 0;">
            <Button fx:id="SubAllSongsButton" text="Songs" styleClass="tab-
button" onAction="#showAllSongsPage"/>
            <Button fx:id="SubArtistsButton" text="Artists"
styleClass="tab-button" onAction="#showArtistsPage"/>
            <Button fx:id="SubAlbumsButton" text="Albums" styleClass="tab-
button" onAction="#showAlbumsPage" />
            <Button fx:id="SubMapsButton" text="Maps" styleClass="tab-
button" onAction="#showMapsPage" />
        </HBox>
        <Separator />
    </VBox>

    <StackPane fx:id="contentStack" VBox.vgrow="ALWAYS" />
</VBox>
```

8. pielikums. **QueuePanel.fxml**

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- Bottom-sheet queue overlay (resize handle, now playing, reorderable
list). -->
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>
<?import javafx.scene.text.TextFlow?>
<?import org.kordamp.ikonli.javafx.FontIcon?>
<AnchorPane fx:id="rootPane" xmlns="http://javafx.com/javafx/17.0.12"
            xmlns:fx="http://javafx.com/fxml/1"

fx:controller="manicsalsa.soufone_proto.controllers.QueueController"
            styleClass="queue-panel">

    <VBox AnchorPane.topAnchor="0"    AnchorPane.bottomAnchor="0"
          AnchorPane.leftAnchor="0"   AnchorPane.rightAnchor="0">
        <HBox fx:id="dragHandle" alignment="CENTER" prefHeight="22"
              style="-fx-cursor: n-resize; -fx-padding: 6 0 4 0;">
            <Region prefWidth="36" prefHeight="4"/>
        </HBox>

        <HBox alignment="CENTER_LEFT" spacing="10"
              style="-fx-padding: 12 16 12 16; ">

            <Button fx:id="closeQueueButton" styleClass="icon-button"
mnemonicParsing="false">
                <graphic><FontIcon iconLiteral="fas-chevron-down"
iconSize="20"/></graphic>
            </Button>

            <Label fx:id="queueCountLabel" text="Queue" styleClass="label-
bold"/>

            <Region HBox.hgrow="ALWAYS"/>

            <Button fx:id="playModeButton" styleClass="icon-button"
mnemonicParsing="false">
                <graphic>
                    <FontIcon fx:id="playModeIcon" iconLiteral="fas-random"
iconSize="20"/>
                </graphic>
            </Button>

            <Button fx:id="clearQueueButton" styleClass="icon-button"
mnemonicParsing="false">
                <graphic><FontIcon iconLiteral="fas-trash-alt"
iconSize="18"/></graphic>
            </Button>
        </HBox>

        <Separator/>

        <VBox style="-fx-padding: 10 16 6 16; -fx-spacing: 4;">
            <Label text="Now Playing" styleClass="label-sub"/>
            <TextFlow fx:id="nowPlayingTitleFlow" styleClass="audio-title"
                      maxWidth="1.7976931348623157E308"/>
            <TextFlow fx:id="nowPlayingArtistFlow" styleClass="audio-
artists"
                      maxWidth="1.7976931348623157E308"/>
        </VBox>
```

```
        <Separator/>

        <HBox style="-fx-padding: 8 16 2 16;">
            <Label text="Up Next"
                   styleClass="label-sub"/>
        </HBox>

        <ListView fx:id="queueListView" VBox.vgrow="ALWAYS"
                  styleClass="queue-list-view">
            <placeholder>
                <Label text="Queue is empty" styleClass="label-sub"/>
            </placeholder>
        </ListView>

    </VBox>
</AnchorPane>
```

9. pielikums. **SettingsPage.fxml**

```
<?xml version="1.0" encoding="UTF-8"?>
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>

<VBox xmlns="http://javafx.com/javafx/17.0.12"
      xmlns:fx="http://javafx.com/fxml/1"

fx:controller="manicsalsa.soufone_proto.controllers.SettingsController"
      minWidth="400" minHeight="400">

    <HBox alignment="CENTER" style="-fx-padding: 20 10 0 10;">
        <Label text="Settings" styleClass="text-header" />
        <Region HBox.hgrow="ALWAYS" />
    </HBox>
    <Separator />
    <VBox styleClass="settings-content" minWidth="400" spacing="10">
        <fx:include source="AppearanceSectionPane.fxml"/>
        <fx:include source="AccessibilitySectionPane.fxml"/>
        <fx:include source="PlaybackSectionPane.fxml"/>
        <fx:include source="FileOptionsSectionPane.fxml"/>
        <fx:include source="KeyBindingsSectionPane.fxml"/>
        <fx:include source="AboutSectionPane.fxml"/>
    </VBox>
</VBox>
```

10. pielikums. **GuidePage.fxml**

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>
<?import org.kordamp.ikonli.javafx.FontIcon?>
<VBox fx:id="pageLayout" xmlns="http://javafx.com/javafx/17.0.12"
xmlns:fx="http://javafx.com/fxml/1"
      fx:controller="manicsalsa.soufone_proto.controllers.GuideController"
      maxHeight="Infinity" VBox.vgrow="ALWAYS">
    <HBox fx:id="pageHeader" alignment="CENTER" style="-fx-padding: 20 10 0
10;">
        <Label text="User Guide" styleClass="text-header"/>
        <Region HBox.hgrow="ALWAYS" />
    </HBox>
```

```
    <Separator />
    <VBox spacing="10.0" VBox.vgrow="ALWAYS" styleClass="guide-content"
minWidth="400">
        <HBox alignment="CENTER_LEFT" spacing="8" styleClass="guide-search-
row">
            <TextField fx:id="searchField" promptText="Search guide
(hotkeys, context menu, pages, settings...)" styleClass="guide-search-
field" HBox.hgrow="ALWAYS" />
            <Label fx:id="matchCounterLabel" text="—" styleClass="guide-
match-counter" minWidth="48" />
            <Button fx:id="prevMatchButton" mnemonicParsing="false"
styleClass="icon-button" disable="true">
                <graphic><FontIcon iconLiteral="fas-chevron-up"
iconSize="16"/></graphic>
                <tooltip><Tooltip text="Previous match (Up)"/></tooltip>
            </Button>
            <Button fx:id="nextMatchButton" mnemonicParsing="false"
styleClass="icon-button" disable="true">
                <graphic><FontIcon iconLiteral="fas-chevron-down"
iconSize="16"/></graphic>
                <tooltip><Tooltip text="Next match (Down)"/></tooltip>
            </Button>
            <Button fx:id="scrollToTopButton" mnemonicParsing="false"
styleClass="icon-button">
                <graphic><FontIcon iconLiteral="fas-arrow-up"
iconSize="16"/></graphic>
                <tooltip><Tooltip text="Scroll to top"/></tooltip>
            </Button>
        </HBox>
        <ScrollPane fx:id="guideScrollPane" styleClass="guide-scroll-pane"
fitToWidth="true" hbarPolicy="NEVER" VBox.vgrow="ALWAYS">
            <VBox fx:id="guideContentBox" styleClass="guide-list-view"
spacing="8"/>
        </ScrollPane>
        <Label fx:id="emptyStateLabel" managed="false" text="No guide
entries found for this search." visible="false" wrapText="true"
styleClass="label-sub guide-empty-state" />
    </VBox>
</VBox>
```

11. pielikums. **AudioItem.fxml**

```
<?xml version="1.0" encoding="UTF-8"?>
<?import javafx.scene.control.*?>
<?import javafx.scene.text.TextFlow?>
<?import javafx.scene.layout.*?>
<?import javafx.scene.image.*?>
<?import javafx.geometry.*?>
<?import org.kordamp.ikonli.javafx.FontIcon?>

<HBox xmlns:fx="http://javafx.com/fxml" fx:id="rootContainer"
      spacing="10" alignment="CENTER_LEFT"
      styleClass="audio-item"

fx:controller="manicsalsa.soufone_proto.controllers.AudioItemController">

    <padding>
        <Insets top="5" right="10" bottom="5" left="10"/>
    </padding>

    <ImageView fx:id="coverImage" fitHeight="40" fitWidth="40"
```

```
                preserveRatio="true">
            <HBox.margin>
                <Insets right="10"/>
            </HBox.margin>
        </ImageView>

        <VBox spacing="2" HBox.hgrow="ALWAYS" prefWidth="0">
            <TextFlow fx:id="titleFlow" styleClass="audio-title"
                      minWidth="250" VBox.vgrow="NEVER"/>
            <HBox spacing="8" alignment="CENTER_LEFT">
                <TextFlow fx:id="artistFlow" styleClass="audio-artists"
                          HBox.hgrow="ALWAYS" minWidth="250"/>
                <Region HBox.hgrow="ALWAYS"/>
                <Label fx:id="durationLabel" text="00:00" HBox.hgrow="NEVER"
minWidth="45" alignment="CENTER_RIGHT"/>
            </HBox>
        </VBox>

        <Button fx:id="favoriteButton" styleClass="action-button, favorite-
button"
                minWidth="30" minHeight="30">
            <tooltip>
                <Tooltip text="Toggle Favorite"/>
            </tooltip>
            <graphic>
                <FontIcon iconLiteral="far-heart" iconSize="20" />
            </graphic>
        </Button>

        <Button fx:id="playButton" styleClass="action-button, play-button"
                minWidth="40" minHeight="40">
            <tooltip>
                <Tooltip text="Play"/>
            </tooltip>
            <graphic>
                <FontIcon iconLiteral="fas-play" iconSize="20"/>
            </graphic>
        </Button>
</HBox>
```

12. pielikums. **GlobalContextMenu.fxml**

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.*?>
<?import org.kordamp.ikonli.javafx.*?>

<ContextMenu fx:id="rootMenu" xmlns="http://javafx.com/javafx/17.0.12"
             xmlns:fx="http://javafx.com/fxml/1"

fx:controller="manicsalsa.soufone_proto.controllers.GlobalContextMenuContro
ller">
    <items>
        <MenuItem fx:id="reloadLibraryMenuItem" text="Reload Library">
            <graphic>
                <FontIcon iconLiteral="fas-sync-alt" iconSize="16"/>
            </graphic>
        </MenuItem>
    </items>
</ContextMenu>
```

```
/* Base structural styles - layout and interactions */

.root { -fx-font-size: 16px; }

.text-header { -fx-font-size: 3em; -fx-text-fill: -fx-base-text; -fx-font-
weight: bold;} /* Use by almost all page headers */

.flow-header { -fx-font-size: 3em; -fx-fill: -fx-base-text; -fx-font-
weight: bold;}

/* ========== Navigation Elements ========== */
.nav-button {
    -fx-cursor: hand;
    -fx-background-color: -fx-nav-button-bg;
    -fx-text-fill: -fx-base-text;
    -fx-font-weight: bold;
    -fx-padding: 8px 12px;
    -fx-alignment: baseline-left;
    -fx-content-display: left;
    -fx-text-overrun: center-ellipsis;
    -fx-max-width: Infinity;
    -fx-min-width: 40px;
}

.nav-button:hover {
    -fx-background-color: -fx-nav-button-bg-hover;
}

.nav-button:selected {
    -fx-background-color: -fx-nav-button-bg-selected;
}

/* Icon-only variant */
.nav-button-minimized {
    -fx-padding: 0;
    -fx-alignment: center;
    -fx-content-display: graphic-only;
    -fx-min-width: 0;
    -fx-max-width: Infinity;
    -fx-pref-width: -1;
}

.nav-button-minimized:selected {
    -fx-background-color: -fx-nav-button-bg-selected;
}

.nav-button-minimized:hover {
    -fx-background-color: -fx-nav-button-bg-hover;
}

/* FontIcon colors inside nav buttons */
.nav-button .ikonli-font-icon {
    -fx-icon-color: -fx-icon-text-color;
}

.nav-button:hover .ikonli-font-icon,
.nav-button:selected .ikonli-font-icon {
    -fx-icon-color: -fx-base-text;
}
```

```
/* ========== Navigation Layout ========== */

.nav-bar,
.nav-pane {
    -fx-background-color: -fx-nav-bg;
    -fx-padding: 0;
}

.nav-header {
    -fx-padding: 0 16 0 16;
    -fx-background-color: -fx-nav-bg;
}

.nav-scroll-pane {
    -fx-background-color: transparent;
    -fx-border-color: transparent;
}

.nav-separator {
    -fx-background-color: transparent;
    -fx-padding: 5;
}

.nav-toggle-button {
    -fx-background-color: transparent;
    -fx-icon-color: -fx-icon-text-color;
    -fx-cursor: hand;
}

.nav-toggle-button:hover {
    -fx-icon-color: -fx-base-text;
}

/* ========== Context Menu ========== */

.context-menu {
    -fx-background-color: -fx-nav-bg;
    -fx-border-color: -fx-nav-button-bg-hover;
    -fx-border-width: 1;
    -fx-background-insets: 0;
    -fx-padding: 4;
}

.context-menu .menu-item {
    -fx-background-color: -fx-nav-button-bg;
    -fx-padding: 6 12;
}

.context-menu .menu-item .label {
    -fx-text-fill: -fx-base-text;
}

.context-menu .context-menu-label {
    -fx-fill: -fx-base-text;
}

.context-menu .custom-menu-item {
    -fx-padding: 6 12;
}

.context-menu .menu-item:hover,
```

```
.context-menu .menu-item:focused {
    -fx-background-color: -fx-nav-button-bg-hover;
}

.context-menu .menu-item:armed {
    -fx-background-color: -fx-nav-button-bg-selected;
}

.context-menu .menu-item .ikonli-font-icon {
    -fx-icon-color: -fx-icon-text-color;
}

.context-menu .menu-item:hover .ikonli-font-icon,
.context-menu .menu-item:focused .ikonli-font-icon,
.context-menu .menu-item:armed .ikonli-font-icon {
    -fx-icon-color: -fx-base-text;
}

.library-label {
    -fx-text-fill: -fx-base-text;
    -fx-font-weight: bold;
}

/* ========== Main Layout ========== */

.split-pane {
    -fx-background-color: -fx-page-bg;
}

.split-pane-divider {
    -fx-background-color: -fx-list-element-bg;
    -fx-padding: 0 1 0 1;
}

.content-scroll-pane {
    -fx-background-color: -fx-page-bg;
    -fx-border-color: transparent;
}

.content-scroll-pane .viewport {
    -fx-background-color: -fx-page-bg;
}

/* ========== Modern Thin Scroll Bars (No Arrows) ========== */

/* Hide the increment/decrement buttons (arrows) entirely */
.scroll-bar .increment-button,
.scroll-bar .decrement-button {
    -fx-padding: 0;
    -fx-opacity: 0;
    -fx-max-height: 0;
    -fx-max-width: 0;
    -fx-pref-height: 0;
    -fx-pref-width: 0;
    -fx-min-height: 0;
    -fx-min-width: 0;
    -fx-background-color: transparent;
}

/* Make the scroll bar track transparent or very subtle */
.scroll-bar .track {
```

```
    -fx-background-color: -fx-solid;
    -fx-background-radius: 10;
    -fx-border-radius: 10;
}

/* Thumb (the draggable part) - thin, rounded, semi-transparent */
.scroll-bar .thumb {
    -fx-background-color: -fx-highlight;
    -fx-background-radius: 10;
    -fx-border-radius: 10;
    -fx-pref-width: 6px;      /* vertical scroll bar thickness */
    -fx-pref-height: 6px;     /* horizontal scroll bar thickness */
}

/* Make thumb slightly darker on hover */
.scroll-bar .thumb:hover {
    -fx-background-color: -fx-base-text;
}

/* Adjust vertical and horizontal bar sizes */
.scroll-bar:vertical {
    -fx-pref-width: 8px;
    -fx-min-width: 8px;
}
.scroll-bar:horizontal {
    -fx-pref-height: 8px;
    -fx-min-height: 8px;
}

/*
 * Scroll policy: horizontal scrolling for narrow tiles lives on .content-
scroll-pane
 * (see MainLayoutController.bindPageWidthOnce). Shell-bound lists use
HBAR_NEVER in code.
 */
.scroll-pane > .corner,
.list-view > .corner {
    -fx-background-color: -fx-page-bg;
}

/* ========== Generic List ========== */

.list-view {
    -fx-background-color: transparent;
    -fx-border-color: transparent;
    -fx-padding: 0;
}

/* Fixed row height for library lists — reduces VirtualFlow measurement
work */
.list-view:not(.search-list-view):not(.guide-list-view) {
    -fx-fixed-cell-size: 58;
}

.guide-scroll-pane {
    -fx-background-color: transparent;
    -fx-background-insets: 0;
    -fx-padding: 0;
}

.guide-scroll-pane > .viewport {
```

```
    -fx-background-color: transparent;
}

.guide-scroll-pane > .corner {
    -fx-background-color: -fx-page-bg;
}

/* Cell reset — audio-item / queue-cell HBoxes own row background and hover
*/
.list-cell {
    -fx-background-color: transparent;
    -fx-padding: 0;
    -fx-text-fill: -fx-base-text;
}

.list-cell:selected {
    -fx-background-color: transparent;
}

/* ========== Audio Item (Song Row) ========== */

/* Items use -fx-list-element-bg so they are visually distinct from the
   transparent list container, which shows the page background behind it.
*/
.audio-item {
    -fx-background-color: -fx-list-element-bg;
    -fx-cursor: default;
}

.audio-item:hover {
    -fx-background-color: -fx-list-element-bg-hover;
}

.audio-item:selected {
    -fx-background-color: -fx-list-element-bg-selected;
}

/* Text inside TextFlows - used by queue, audio, group items */
.audio-title {
    -fx-font-weight: bold;
    -fx-fill: -fx-base-text;
}

.audio-item:playing .audio-title {
    -fx-fill: -fx-highlight;
}

.audio-item:missing-file {
    -fx-opacity: 0.42;
}

.audio-item:missing-file:hover {
    -fx-opacity: 0.55;
}

.audio-artists {
    -fx-opacity: 0.9;
    -fx-fill: -fx-base-text;
}

/* ========== Action Buttons (Play, Favorite, Options) ========== */
```

```
.action-button {
    -fx-background-color: transparent;
    -fx-cursor: hand;
    -fx-padding: 5;
}

.action-button:hover {
    -fx-background-color: -fx-icon-button-bg-hover;
}

/* List rows: hover handled at ListCell level — skip nested button repaints
*/
.audio-item .action-button:hover,
.audio-item .play-button:hover,
.audio-item .icon-button:hover,
.queue-cell .action-button:hover,
.queue-cell .play-button:hover,
.queue-cell .icon-button:hover {
    -fx-background-color: transparent;
}

.audio-item .favorite-button:hover,
.queue-cell .favorite-button:hover {
    -fx-icon-color: -fx-icon-text-color;
}

.audio-item .play-button:hover,
.queue-cell .play-button:hover {
    -fx-background-color: -fx-nav-button-bg;
}

.action-draggable {
    -fx-font-size: 1.1em;
    -fx-text-fill: -fx-icon-text-color;
    -fx-background-color: transparent;
    -fx-padding: 5;
}

.action-draggable:hover {
    -fx-cursor: move;
}

/* Favorite (heart icon) */
.favorite-button {
    -fx-icon-color: -fx-icon-text-color;
}

.favorite-button:hover {
    -fx-icon-color: -fx-highlight;
}

.favorite-button:selected,
.favorite-button-active {
    -fx-icon-color: #e53935;
}

/* Circular play button */
.play-button {
    -fx-background-color: -fx-nav-button-bg;
    -fx-background-radius: 50%;
```

```
    -fx-min-width: 36px;
    -fx-min-height: 36px;
}

.play-button:hover {
    -fx-background-color: -fx-nav-button-bg-hover;
}

.play-button .ikonli-font-icon {
    -fx-icon-color: -fx-icon-text-color;
    -fx-icon-size: 16px;
}

/* Three-dot options button */
.options-button {
    -fx-icon-color: -fx-icon-text-color;
}

.options-button:hover {
    -fx-icon-color: -fx-base-text;
}


/* ========== Songs Page Toolbar ========== */

.play-all-button {
    -fx-background-color: -fx-nav-button-bg;
    -fx-background-radius: 50px;
    -fx-padding: 8 20;
    -fx-text-fill: -fx-base-text;
    -fx-font-weight: bold;
    -fx-cursor: hand;
}

.play-all-button:hover {
    -fx-background-color: -fx-nav-button-bg-hover;
}

.count-label {
    -fx-text-fill: -fx-base-text;
}

.toolbar-button {
    -fx-background-color: transparent;
    -fx-icon-color: -fx-icon-text-color;
}

.toolbar-button:hover {
    -fx-background-color: -fx-icon-button-bg-hover;
}

.toolbar-button:selected {
    -fx-background-color: -fx-nav-button-bg-selected;
}

/* ========== Settings Page ========== */

.danger-button {
    -fx-background-color: #c0392b;
    -fx-text-fill: white;
    -fx-cursor: hand;
```

```
}

.danger-button:hover {
    -fx-background-color: #e74c3c;
}

/* ========== Settings Page ========== */

/* ========== Guide Page ========== */
.guide-content {
    -fx-padding: 0;
    -fx-min-width: 400px;
    -fx-background-color: -fx-page-bg;
}

.guide-search-row {
    -fx-padding: 10 10 10 10;
}

.guide-search-field {
    -fx-background-color: -fx-nav-button-bg;
    -fx-text-fill: -fx-base-text;
}

.guide-list {
    -fx-background-color: transparent;
}

/* Section groups entries; each entry is its own panel below the section
title */
.guide-content .settings-section {
    -fx-background-color: transparent;
    -fx-padding: 0 0 4 0;
}

.guide-content .settings-section-title {
    -fx-padding: 16 0 8 0;
}

.guide-entry {
    -fx-padding: 14 16 14 16;
    -fx-spacing: 6;
    -fx-background-color: -fx-list-element-bg;
    -fx-background-radius: 5;
}

.guide-entry-section {
    -fx-text-fill: -fx-highlight;
    -fx-fill: -fx-highlight;
    -fx-font-weight: bold;
    -fx-font-size: 0.85em;
}

.guide-entry-title {
    -fx-text-fill: -fx-base-text;
    -fx-fill: -fx-base-text;
    -fx-font-weight: bold;
    -fx-font-size: 1.05em;
}

.guide-entry-content {
```

```
    -fx-text-fill: -fx-base-text;
    -fx-fill: -fx-base-text;
}

.guide-list .list-cell,
.guide-list-view .list-cell {
    -fx-padding: 0 0 8 0;
    -fx-background-color: transparent;
}

.guide-list-view .list-cell:hover {
    -fx-background-color: transparent;
}

.guide-empty-state {
    -fx-padding: 0 0 10 0;
}

/* Main content container */
.settings-content {
    -fx-padding: 10 30 30 30;
    -fx-min-width: 400px;
    -fx-background-color: -fx-page-bg;
}

/* Section */
.settings-section {
    -fx-padding: 20 20 20 20;
    -fx-background-color: -fx-list-element-bg;
    -fx-background-radius: 5;
}

.settings-section-title {
    -fx-padding: 20 10 10 10;
    -fx-font-weight: bold;
    -fx-text-fill: -fx-base-text;
    -fx-font-size: 1.2em;
}

/* Standard label */
.label {
    -fx-text-fill: -fx-base-text;
}

.label-bold {
    -fx-text-fill: -fx-base-text;
    -fx-font-weight: bold;
}

.label-sub {
    -fx-opacity: 0.7;
    -fx-font-size: 0.8em;
    -fx-font-style: normal;
}

/* Default button (neutral action) */
.default-button {
    -fx-background-color: -fx-nav-button-bg;
    -fx-text-fill: -fx-base-text;
    -fx-font-weight: bold;
    -fx-padding: 6 12;
```

```
    -fx-cursor: hand;
    -fx-background-radius: 4;
    -fx-border-color: -fx-solid;
    -fx-border-radius: 2;
}

.default-button:hover {
    -fx-background-color: -fx-nav-button-bg-hover;
}

.default-button:pressed {
    -fx-background-color: -fx-nav-button-bg-selected;
}

/* Combobox in settings */
.settings-combobox {
    -fx-background-color: -fx-nav-button-bg;
    -fx-padding: 5;
    -fx-border-color: -fx-solid;
    -fx-border-radius: 2;
}

.settings-combobox .list-cell {
    -fx-text-fill: -fx-base-text;
    -fx-background-color: -fx-nav-button-bg;
}

.settings-combobox .list-cell:hover {
    -fx-background-color: -fx-nav-button-bg-hover;
}

/* Checkbox */
.settings-checkbox {
    -fx-text-fill: -fx-base-text;
}

.settings-checkbox .box {
    -fx-background-color: -fx-solid;
}

/* TableView for directories and key bindings */
.settings-table {
    -fx-background-color: -fx-list-element-bg;
    -fx-border-color: -fx-highlight;
    -fx-table-cell-border-color: transparent;
}

.settings-table .column-header-background {
    -fx-background-color: -fx-solid;
}

.settings-table .column-header,
.settings-table .filler {
    -fx-background-color: -fx-solid;
    -fx-border-color: transparent;
}

.settings-table .column-header .label {
    -fx-text-fill: -fx-base-text;
    -fx-font-weight: bold;
}
```

```
.settings-table .table-row-cell {
    -fx-background-color: -fx-list-element-bg;
    -fx-text-fill: -fx-base-text;
}

.settings-table .table-row-cell:odd {
    -fx-background-color: derive(-fx-list-element-bg, -10%);
}

.settings-table .table-row-cell:selected {
    -fx-background-color: -fx-list-element-bg-selected;
}

.settings-table .table-cell {
    -fx-text-fill: -fx-base-text;
    -fx-border-color: transparent;
}

/* Toolbar in settings (for directory buttons) */
.settings-toolbar {
    -fx-background-color: transparent;
    -fx-padding: 0;
}

/* Font preview text flow */
.font-preview-flow {
     -fx-padding: 4;
     -fx-fill: -fx-base-text;
}

/* Font size value label */
.font-size-value-label {
    -fx-font-weight: bold;
    -fx-text-fill: -fx-base-text;
    -fx-padding: 0 0 8 0;
}

/* Slider in settings */
.settings-slider {
    -fx-control-inner-background: -fx-list-element-bg;
}

.settings-slider .track {
    -fx-background-color: -fx-nav-button-bg;
}

.settings-slider .thumb {
    -fx-background-color: -fx-highlight;
}

.settings-slider .axis {
    -fx-tick-label-fill: -fx-base-text;
}

/* ========== Bottom Player Bar ========== */

.bottom-container {
    -fx-background-color: -fx-player-bg;
}
```

```
/* Library reload banner (bottom of content area) */
.library-reload-banner {
    -fx-background-color: -fx-player-bg;
    -fx-background-radius: 8;
    -fx-border-color: -fx-highlight;
    -fx-border-width: 1;
    -fx-border-radius: 8;
    -fx-effect: dropshadow(gaussian, rgba(0, 0, 0, 0.2), 10, 0, 0, 2);
}

.library-reload-status {
    -fx-text-fill: -fx-base-text;
    -fx-font-size: 13px;
}

.library-reload-icon {
    -fx-icon-color: -fx-highlight;
}

.library-reload-cancel-button {
    -fx-min-width: 72px;
}

/* Library list loading / empty placeholders */
.library-loading-placeholder {
    -fx-padding: 40;
    -fx-alignment: center;
}

.library-loading-label {
    -fx-text-fill: -fx-base-text;
    -fx-font-size: 14px;
    -fx-opacity: 0.75;
}

.library-empty-placeholder {
    -fx-padding: 40;
    -fx-alignment: center;
}

.library-empty-title {
    -fx-font-size: 18px;
    -fx-font-weight: bold;
    -fx-text-fill: -fx-base-text;
}

.library-empty-subtitle {
    -fx-text-fill: -fx-base-text;
    -fx-opacity: 0.85;
}

.bottom-time-label,
.bottom-volume-label {
    -fx-text-fill: -fx-base-text;
    -fx-font-family: monospace;
    -fx-font-size: 11px;
}

.bottom-volume-label {
    -fx-min-width: 32px;
    -fx-alignment: center-left;
```

```
}

.bottom-volume-label.volume-boosted {
    -fx-text-fill: #ffb347;
    -fx-font-weight: 700;
}

.bottom-progress-bar {
    -fx-accent: -fx-highlight;
    -fx-background-color: -fx-list-element-bg;
    -fx-pref-height: 6;
    -fx-max-height: 6;
}

.bottom-progress-bar .track {
    -fx-background-color: -fx-highlight;
    -fx-background-radius: 3;
}

.bottom-progress-bar .bar {
    -fx-background-color: -fx-list-element-bg;
    -fx-background-radius: 3;
}

.bottom-volume-slider {
    -fx-accent: -fx-highlight;
    -fx-pref-height: 14;
    -fx-min-height: 14;
    -fx-max-height: 14;
}

.bottom-volume-slider .track {
    -fx-background-color: -fx-highlight;
    -fx-background-radius: 3;
}

.bottom-volume-slider .bar {
    -fx-background-color: -fx-list-element-bg;
    -fx-background-radius: 3;
}

/* Stronger, VLC-like boosted volume state (>100%) */
.bottom-volume-slider.volume-boosted .track {
    -fx-background-color: linear-gradient(to right, -fx-highlight 0%,
#ff9f43 65%, #ff6b35 100%);
    -fx-background-radius: 4;
    -fx-border-color: rgba(255, 179, 71, 0.85);
    -fx-border-radius: 4;
    -fx-border-width: 1;
}

.bottom-volume-slider.volume-boosted .thumb {
    -fx-background-color: #ffb347;
    -fx-background-insets: 0;
    -fx-background-radius: 9;
    -fx-border-color: #ff6b35;
    -fx-border-width: 2;
    -fx-border-radius: 9;
    -fx-padding: 6;
}
```

```
.bottom-volume-slider.volume-boosted .bar {
    -fx-background-color: rgba(255, 107, 53, 0.35);
    -fx-background-radius: 4;
}

.bottom-control-button.volume-boosted {
    -fx-background-color: rgba(255, 107, 53, 0.18);
    -fx-background-radius: 8;
    -fx-border-color: rgba(255, 179, 71, 0.9);
    -fx-border-width: 1;
    -fx-border-radius: 8;
    -fx-icon-color: #ffb347;
}

.bottom-control-button.volume-boosted:hover {
    -fx-background-color: rgba(255, 107, 53, 0.28);
}

.bottom-control-button {
    -fx-background-color: transparent;
    -fx-icon-color: -fx-icon-text-color;
    -fx-cursor: hand;
    -fx-padding: 8;
}

.bottom-control-button:hover {
    -fx-background-color: -fx-icon-button-bg-hover;
    -fx-icon-color: -fx-base-text;
}

.bottom-control-button:disabled {
    -fx-opacity: 0.35;
    -fx-cursor: default;
}

.bottom-control-button:disabled:hover {
    -fx-background-color: transparent;
}

/* ========== Generic Icon Button ========== */

.icon-button {
    -fx-background-color: transparent;
    -fx-icon-color: -fx-icon-text-color;
    -fx-cursor: hand;
}

.icon-button:hover {
    -fx-background-color: -fx-icon-button-bg-hover;
    -fx-icon-color: -fx-base-text;
}

/* ========== All FontIcons (global fallback) ========== */

.ikonli-font-icon {
    -fx-icon-color: -fx-icon-text-color;
}

/* ========== Big Player ========== */

.big-player-icon {
```

```
    -fx-icon-color: -fx-icon-text-color;
}

.big-player-container {
    -fx-background-color: -fx-player-bg;
}

.big-player-play-button {
    -fx-background-color: -fx-nav-button-bg;
    -fx-background-radius: 50%;
    -fx-min-width: 70px;
    -fx-min-height: 70px;
    -fx-padding: 14;
    -fx-cursor: hand;
}

.big-player-play-button:hover {
    -fx-background-color: -fx-nav-button-bg-hover;
}

.big-player-title {
    -fx-font-weight: bold;
    -fx-fill: -fx-highlight;
    -fx-font-size: 2.5em;
}

.big-player-sub-text {
    -fx-fill: -fx-base-text;
    -fx-font-size: 1.5em;
}

.big-player-header,
.big-player-footer {
    -fx-padding: 8;
}

.big-player-content {
    -fx-padding: 16 20;
}

/* ========== Queue Panel ========== */

.queue-panel {
    -fx-background-color: -fx-queue-bg;
}

.queue-list-view {
    -fx-background-color: -fx-queue-bg;
    -fx-border-color: transparent;
    -fx-padding: 0;
}

.queue-list-view .list-cell {
    -fx-background-color: transparent;
    -fx-padding: 0;
    -fx-text-fill: -fx-base-text;
}

.queue-list-view .list-cell:selected,
.queue-list-view .list-cell:focused:selected {
    -fx-background-color: transparent;
```

```
}

.queue-cell {
    -fx-background-color: -fx-list-element-bg;
}

.queue-cell:hover {
    -fx-background-color: -fx-list-element-bg-hover;
}

.queue-cell:playing {
    -fx-background-color: -fx-list-element-bg-selected;
    -fx-opacity: 0.85;
}

.queue-cell:playing:hover {
    -fx-background-color: -fx-list-element-bg-hover;
    -fx-opacity: 0.95;
}

/* Queue reorder — only the row that will be replaced at the drop position
*/
.queue-cell:reorder-drop-target {
    -fx-background-color: -fx-nav-button-bg-selected;
    -fx-opacity: 0.3;
    -fx-border-color: -fx-highlight;
    -fx-border-width: 0 0 0 5;
    -fx-background-insets: 0;
    -fx-border-insets: 0;
}

.queue-cell:playing:reorder-drop-target {
    -fx-background-color: -fx-nav-button-bg;
    -fx-opacity: 0.3;
}

/* ========== Sub-page Tabs ========== */
.tab-button {
    -fx-cursor: hand;
    -fx-background-color: -fx-nav-button-bg;
    -fx-text-fill: -fx-base-text;
    -fx-font-weight: bold;
    -fx-padding: 8 16;
    -fx-alignment: center;
    -fx-background-radius: 4;
    -fx-border-radius: 8px 8px 0 0;
    -fx-overflow: hidden;
}

.tab-button:hover {
    -fx-background-color: -fx-nav-button-bg-hover;
}

.tab-button:selected {
    -fx-background-color: -fx-nav-button-bg-selected;
}

/* Search page */
.search-highlight {
    -fx-fill: -fx-highlight;
    -fx-font-weight: bold;
```

```
}

.search-highlight-active {
    -fx-fill: -fx-hover-highlight;
    -fx-font-weight: bold;
    -fx-underline: true;
}

.guide-match-counter {
    -fx-text-fill: -fx-base-text;
    -fx-opacity: 0.85;
    -fx-alignment: center-right;
}

.search-field {
    -fx-text-fill: -fx-solid;
}

.search-field .text {
    -fx-fill: -fx-solid;
}

/* prompt text sits in a separate node */
.search-field:focused .prompt-text,
.search-field .prompt-text {
    -fx-fill: -fx-solid;
}
```

14. pielikums. **DatabaseConfig.java**

```
package manicsalsa.soufone_proto.config;

import java.io.File;
import java.io.IOException;
import java.util.logging.Logger;

/**
 * Location of the SQLite music library and the user app data directory
 * ({@code user.home}/.soufone). Property configs use the same folder via
 * {@link #getAppDataDir()}.

 * The app data directory is created when this class is first loaded.
 * {@link #ensureDatabaseDirectoryExists()} can recreate it later if
missing.
 */
public class DatabaseConfig {
    private static final String APP_DATA_DIR =
System.getProperty("user.home") + "/.soufone";
    private static final String DB_FILE = "music_library.db";
    private static final Logger LOGGER =
Logger.getLogger(DatabaseConfig.class.getName());

    static {
        // Create app directory if it doesn't exist
        File appDir = new File(APP_DATA_DIR);
        if (!appDir.exists()) {
            boolean created = appDir.mkdirs();
            if (!created) {
                LOGGER.severe("Failed to create application directory: " +
APP_DATA_DIR);
```

```
                throw new RuntimeException("Failed to create application
directory: " + APP_DATA_DIR);
            }
            LOGGER.config("Created application directory: " +
APP_DATA_DIR);
        }
    }

    public static String getDatabasePath() {
        return APP_DATA_DIR + File.separator + DB_FILE;
    }

    public static String getConnectionString() {
        return "jdbc:sqlite:" + getDatabasePath();
    }

    public static File getDatabaseFile() {
        return new File(getDatabasePath());
    }

    public static boolean databaseFileExists() {
        return getDatabaseFile().exists();
    }

    public static String getAppDataDir() {
        return APP_DATA_DIR;
    }

    public static void ensureDatabaseDirectoryExists() throws IOException {
        File dbFile = getDatabaseFile();
        File parentDir = dbFile.getParentFile();

        if (parentDir != null && !parentDir.exists()) {
            boolean created = parentDir.mkdirs();
            if (!created) {
                throw new IOException("Failed to create database directory:
" + parentDir.getAbsolutePath());
            }
            LOGGER.config("Created database directory: " +
parentDir.getAbsolutePath());
        }
    }
}
```

15. pielikums. **AppConfigPaths.java**

```
package manicsalsa.soufone_proto.config;

import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.StandardCopyOption;
import java.util.logging.Logger;

/**
 * Resolves paths for {@link #APP_PREFERENCES_FILE}, {@link
#THEME_CONFIG_FILE},
 * and {@link #HOT_KEY_CONFIG_FILE} under {@link
DatabaseConfig#getAppDataDir()}
 * (same directory as the SQLite library database).
```

```
 * On first resolve per file, if the destination is missing but a same-
named file
 * exists in the process working directory, it is copied into app data; the
legacy
 * file is left in place. Migration failures are logged; the app-data path
is
 * returned either way, including before the file has been created.
 */
public final class AppConfigPaths {

    private static final Logger LOGGER =
Logger.getLogger(AppConfigPaths.class.getName());

    public static final String APP_PREFERENCES_FILE =
"app_preferences.properties";
    public static final String THEME_CONFIG_FILE =
"theme_config.properties";
    public static final String HOT_KEY_CONFIG_FILE =
"hot_key_config.properties";

    private AppConfigPaths() {}

    public static Path appPreferences() {
        return resolve(APP_PREFERENCES_FILE);
    }

    public static Path themeConfig() {
        return resolve(THEME_CONFIG_FILE);
    }

    public static Path hotKeyConfig() {
        return resolve(HOT_KEY_CONFIG_FILE);
    }

    private static Path resolve(String fileName) {
        ensureConfigDirExists();
        Path target =
Path.of(DatabaseConfig.getAppDataDir()).resolve(fileName);
        if (Files.exists(target)) {
            return target;
        }
        migrateLegacyConfigIfPresent(fileName, target);
        return target;
    }

    static void ensureConfigDirExists() {
        try {
            DatabaseConfig.ensureDatabaseDirectoryExists();
        } catch (IOException e) {
            throw new RuntimeException("Failed to create config directory",
e);
        }
    }

    private static void migrateLegacyConfigIfPresent(String fileName, Path
target) {
        Path legacy = Path.of(fileName).toAbsolutePath().normalize();
        if (!Files.exists(legacy) ||
legacy.equals(target.toAbsolutePath().normalize())) {
            return;
        }
```

```
        try {
            Files.copy(legacy, target,
StandardCopyOption.REPLACE_EXISTING);
            LOGGER.info("Migrated config " + fileName + " to " + target);
        } catch (IOException e) {
            LOGGER.warning("Could not migrate legacy config " + legacy + ":
" + e.getMessage());
        }
    }
}
```

16. pielikums. **LazyAudioList. java**

```
package manicsalsa.soufone_proto.services;

import manicsalsa.soufone_proto.database.dao.AudioDAO;
import manicsalsa.soufone_proto.database.models.Audio;

import java.sql.SQLException;
import java.util.AbstractList;
import java.util.LinkedHashMap;
import java.util.List;
import java.util.Map;
import java.util.Objects;
import java.util.Set;
import java.util.concurrent.ConcurrentHashMap;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;

/**
 * Paged {@link AbstractList} for {@code ListView}: full {@link #size()}
from the DB,
 * rows loaded in fixed-size pages into a bounded LRU cache.
 */
public final class LazyAudioList extends AbstractList<Audio> {

    /** Which {@link AudioDAO} query backs this list. */
    public enum Mode { ALL, FAVORITES }

    private static final int PAGE_SIZE = 100;
    private static final int MAX_CACHED = 400;

    private final AudioDAO dao;
    private final Mode mode;

    private int totalCount;

    private final Map<Integer, Audio> cache = new
LinkedHashMap<>(MAX_CACHED, 0.75f, true) {
        @Override
        protected boolean removeEldestEntry(Map.Entry<Integer, Audio>
eldest) {
            return size() > MAX_CACHED;
        }
    };

    private final Set<Integer> prefetchInFlight =
ConcurrentHashMap.newKeySet();
    private final ExecutorService prefetchExecutor =
Executors.newSingleThreadExecutor(r -> {
        Thread t = new Thread(r, "lazy-audio-prefetch");
```

```
        t.setDaemon(true);
        return t;
    });

    public LazyAudioList(AudioDAO dao, Mode mode) throws SQLException {
        this.dao = dao;
        this.mode = mode;
        refreshCount();
    }

    /** Drops cached pages without refreshing count. */
    public void clearCache() {
        synchronized (cache) {
            cache.clear();
        }
    }

    /** Last known row count (may shrink on sparse {@link #get(int)}). */
    public int getTotalCount() {
        return totalCount;
    }

    @Override
    public int size() {
        return totalCount;
    }

    /** Loads the page containing {@code index} so a list view can scroll
to that row. */
    public void warmIndex(int index) {
        Objects.requireNonNull(get(index));
    }

    /**
     * Loads the page containing {@code index}. If the row vanished
(concurrent delete),
     * shrinks {@link #totalCount} and throws {@link
IndexOutOfBoundsException}.
     */
    @Override
    public Audio get(int index) {
        if (index < 0 || index >= totalCount) {
            throw new IndexOutOfBoundsException("index=" + index + ",
size=" + totalCount);
        }
        Audio hit;
        synchronized (cache) {
            hit = cache.get(index);
        }
        if (hit != null) {
            schedulePrefetchAround(index);
            return hit;
        }

        int pageStart = (index / PAGE_SIZE) * PAGE_SIZE;
        loadPageSync(pageStart);

        synchronized (cache) {
            hit = cache.get(index);
        }
        if (hit != null) {
```

```
            schedulePrefetchAround(index);
            return hit;
        }

        totalCount = pageStart;
        throw new IndexOutOfBoundsException("index=" + index + ", size=" +
totalCount);
    }

    private void loadPageSync(int pageStart) {
        try {
            List<Audio> page = switch (mode) {
                case ALL -> dao.findAll(PAGE_SIZE, pageStart, false);
                case FAVORITES -> dao.findFavorites(PAGE_SIZE, pageStart,
false);
            };
            synchronized (cache) {
                for (int i = 0; i < page.size(); i++) {
                    cache.put(pageStart + i, page.get(i));
                }
            }
        } catch (SQLException e) {
            throw new RuntimeException("Failed to load audio page at offset
" + pageStart, e);
        }
    }

    private void schedulePrefetchAround(int index) {
        int pageStart = (index / PAGE_SIZE) * PAGE_SIZE;
        prefetchPage(pageStart + PAGE_SIZE);
        if (pageStart >= PAGE_SIZE) {
            prefetchPage(pageStart - PAGE_SIZE);
        }
    }

    private void prefetchPage(int pageStart) {
        if (pageStart < 0 || pageStart >= totalCount) {
            return;
        }
        if (!prefetchInFlight.add(pageStart)) {
            return;
        }
        prefetchExecutor.execute(() -> {
            try {
                synchronized (cache) {
                    if (cache.containsKey(pageStart)) {
                        return;
                    }
                }
                List<Audio> page = switch (mode) {
                    case ALL -> dao.findAll(PAGE_SIZE, pageStart, false);
                    case FAVORITES -> dao.findFavorites(PAGE_SIZE,
pageStart, false);
                };
                synchronized (cache) {
                    for (int i = 0; i < page.size(); i++) {
                        cache.putIfAbsent(pageStart + i, page.get(i));
                    }
                }
            } catch (SQLException e) {
```

```
                // Prefetch is best-effort; synchronous load will retry on
scroll.
            } finally {
                prefetchInFlight.remove(pageStart);
            }
        });
    }

    private void refreshCount() throws SQLException {
        totalCount = switch (mode) {
            case ALL -> dao.getTotalCount();
            case FAVORITES -> dao.getTotalFavoritesCount();
        };
    }

    /** Stops the prefetch executor when this list is replaced or the app
exits. */
    public void close() {
        prefetchInFlight.clear();
        prefetchExecutor.shutdown();
    }
}
```

17. pielikums. **LibraryIndexCache.java**

```
package manicsalsa.soufone_proto.services;

import javafx.application.Platform;
import manicsalsa.soufone_proto.database.dao.AudioDAO;

import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import java.util.concurrent.atomic.AtomicBoolean;
import java.util.logging.Level;
import java.util.logging.Logger;

/**
 * Cached full-library track id ordering for Play All and scroll-to-track.
 * Invalidated when the library changes; loaded off the FX thread.
 */
public final class LibraryIndexCache {

    private static final Logger LOGGER =
Logger.getLogger(LibraryIndexCache.class.getName());
    private static final LibraryIndexCache INSTANCE = new
LibraryIndexCache();

    private volatile List<Integer> cachedIds = List.of();
    private final AtomicBoolean loadInProgress = new AtomicBoolean(false);
    private final Object pendingLock = new Object();
    private final List<Runnable> pendingFxCallbacks = new ArrayList<>();

    private LibraryIndexCache() {}

    public static LibraryIndexCache getInstance() {
        return INSTANCE;
    }

    public List<Integer> getIdsSnapshot() {
        return cachedIds;
```

```
    }

    public void invalidate() {
        cachedIds = List.of();
    }

    /**
     * Loads ids if the cache is empty. {@code onReady} runs on the FX
thread when ids are available
     * (including immediately when already cached).
     */
    public void ensureLoaded(AudioDAO dao, Runnable onReady) {
        if (!cachedIds.isEmpty()) {
            if (onReady != null) {
                Platform.runLater(onReady);
            }
            return;
        }
        if (onReady != null) {
            enqueueFxCallback(onReady);
        }
        if (!loadInProgress.compareAndSet(false, true)) {
            return;
        }
        Thread t = new Thread(() -> {
            try {
                loadIds(dao);
            } finally {
                loadInProgress.set(false);
                runPendingFxCallbacks();
            }
        }, "library-index-cache");
        t.setDaemon(true);
        t.start();
    }

    private void loadIds(AudioDAO dao) {
        try {
            List<Integer> ids = dao.findAllIds();
            cachedIds = List.copyOf(ids);
        } catch (SQLException e) {
            LOGGER.log(Level.WARNING, "Failed to load library track ids",
e);
        }
    }

    private void enqueueFxCallback(Runnable onReady) {
        synchronized (pendingLock) {
            pendingFxCallbacks.add(() -> Platform.runLater(onReady));
        }
    }

    private void runPendingFxCallbacks() {
        List<Runnable> batch;
        synchronized (pendingLock) {
            if (pendingFxCallbacks.isEmpty()) {
                return;
            }
            batch = List.copyOf(pendingFxCallbacks);
            pendingFxCallbacks.clear();
        }
```

```
        for (Runnable callback : batch) {
            try {
                callback.run();
            } catch (RuntimeException e) {
                LOGGER.log(Level.WARNING, "Library index cache callback
failed", e);
            }
        }
    }

    /** Forces a reload and notifies on the FX thread when complete. */
    public void reload(AudioDAO dao, Runnable onReady) {
        invalidate();
        ensureLoaded(dao, onReady);
    }
}
```

18. pielikums. **AudioPlayerService.java**

```
package manicsalsa.soufone_proto.services;

import javafx.application.Platform;
import javafx.beans.property.DoubleProperty;
import javafx.beans.property.SimpleDoubleProperty;
import javafx.beans.property.SimpleStringProperty;
import javafx.beans.property.StringProperty;
import manicsalsa.soufone_proto.config.AppPreferences;
import manicsalsa.soufone_proto.database.DatabaseHelper;
import manicsalsa.soufone_proto.database.dao.AudioDAO;
import manicsalsa.soufone_proto.database.dao.PlayerStateDAO;
import manicsalsa.soufone_proto.database.models.Audio;
import manicsalsa.soufone_proto.database.models.PlayHistory;
import manicsalsa.soufone_proto.managers.PlayerControlsManager;
import manicsalsa.soufone_proto.services.playback.PlaybackPosition;
import manicsalsa.soufone_proto.services.playback.PlaybackTrackRef;
import manicsalsa.soufone_proto.services.playback.QueueRowIndex;
import manicsalsa.soufone_proto.services.queue.QueueAudioStubs;

import java.io.File;
import java.sql.SQLException;
import java.util.Collection;
import java.util.List;
import java.util.Objects;
import java.util.Optional;
import java.util.Set;
import java.util.concurrent.CopyOnWriteArrayList;
import java.util.function.BiConsumer;
import java.util.function.Supplier;
import java.util.logging.Logger;

/**
 * Transport only: play, pause, seek, volume. No queue policy — {@link
PlayerControlsManager}
 * advances on {@link #addEndOfMediaListener}. Status via {@link
#addStatusListener}.
 */
public class AudioPlayerService {
    private static final Logger LOGGER =
Logger.getLogger(AudioPlayerService.class.getName());

    private static AudioPlayerService instance;
```

```
    private PlayerStateDAO playerStateDAO;
    private static final Set<String> MPEG4_AUDIO_EXTENSIONS =
            Set.of("m4a", "aac", "mp4", "m4b");

    private Mp3Player currentPlayer;
    private Integer   currentAudioId;
    /** Index of the row in {@link QueueService} tied to this playback, or
{@code -1} if none. */
    private int       currentQueueIndex = -1;

    /** Seek applied on next mount; {@code -1} if none. */
    private double pendingSeek = -1;

    private AudioDAO getAudioDAO() {
        try {
            return DatabaseHelper.getInstance().getAudioDAO();
        } catch (SQLException e) {
            LOGGER.severe("Failed to get AudioDAO: " + e.getMessage());
            return null;
        }
    }

    private String currentAudioPath;
    private final DoubleProperty currentTime   = new
SimpleDoubleProperty(0);
    private final DoubleProperty totalDuration = new
SimpleDoubleProperty(0);
    private final DoubleProperty volume        = new
SimpleDoubleProperty(1.0);

    private final StringProperty currentAudioName   = new
SimpleStringProperty("");
    private final StringProperty currentAudioArtist = new
SimpleStringProperty("");
    private final StringProperty currentAlbumName   = new
SimpleStringProperty("");

    private final List<BiConsumer<Integer, Mp3Player.Status>>
statusListeners =
            new CopyOnWriteArrayList<>();

    private final List<BiConsumer<Integer, Integer>> endOfMediaListeners =
            new CopyOnWriteArrayList<>();

    private final List<BiConsumer<Integer, String>>
missingAudioFileListeners =
            new CopyOnWriteArrayList<>();

    /** Paths already reported missing (one alert per path until
forgotten). */
    private final Set<String> missingAudioFileNotifiedPaths =
            java.util.concurrent.ConcurrentHashMap.newKeySet();

    private final java.util.concurrent.ScheduledExecutorService
seekExecutor =

java.util.concurrent.Executors.newSingleThreadScheduledExecutor(r -> {
                Thread t = new Thread(r, "seek-debounce");
                t.setDaemon(true);
                return t;
            });
```

```
    private final java.util.concurrent.atomic.AtomicReference<
            java.util.concurrent.ScheduledFuture<?>> pendingSeekTask =
            new java.util.concurrent.atomic.AtomicReference<>();

    private volatile Double lastRequestedSeekSeconds = null;
    private static final long SEEK_DEBOUNCE_MS = 120;

    /** DB position flush interval while playing (also saved on
pause/stop/seek). */
    private static final long POSITION_SAVE_INTERVAL_SECONDS = 5L;

    private final java.util.concurrent.atomic.AtomicReference<
            java.util.concurrent.ScheduledFuture<?>> periodicPositionSave =
            new java.util.concurrent.atomic.AtomicReference<>();

    private AudioPlayerService() {
        loadSavedVolume();
    }

    public static synchronized AudioPlayerService getInstance() {
        if (instance == null) instance = new AudioPlayerService();
        return instance;
    }

    private void loadSavedVolume() {
        ensurePlayerStateDAO();
        if (playerStateDAO != null) {
            try {
                volume.set(Math.clamp(playerStateDAO.load().getVolume(),
0.0, MAX_VOLUME));
            } catch (SQLException e) {
                LOGGER.warning("Could not load saved volume, using 100%");
            }
        }
    }

    /** ({@code audioId}, status) on every playback status change. */
    public void addStatusListener(BiConsumer<Integer, Mp3Player.Status>
listener) {
        statusListeners.add(listener);
    }

    public void removeStatusListener(BiConsumer<Integer, Mp3Player.Status>
listener) {
        statusListeners.remove(listener);
    }

    /** ({@code audioId}, {@code queueIndex}) when the current track ends
naturally. */
    public void addEndOfMediaListener(BiConsumer<Integer, Integer>
listener) {
        endOfMediaListeners.add(listener);
    }

    /** ({@code audioId}, path) when play is requested for a missing file.
*/
    public void addMissingAudioFileListener(BiConsumer<Integer, String>
listener) {
        missingAudioFileListeners.add(listener);
    }
```

```
    /** Clears rate-limit state so the same path can alert again. */
    public void forgetMissingFileNotification(String path) {
        if (path != null) {
            missingAudioFileNotifiedPaths.remove(path);
        }
    }

    public Integer getCurrentAudioId()   { return currentAudioId; }
    public String  getCurrentAudioPath() { return currentAudioPath; }
    public int     getCurrentQueueIndex() { return currentQueueIndex; }

    public boolean isPlaying() {
        return isCurrentPlayerPlaying();
    }

    public DoubleProperty currentTimeProperty()      { return currentTime;
}
    public DoubleProperty totalDurationProperty()    { return
totalDuration; }
    public DoubleProperty volumeProperty()           { return volume; }
    public StringProperty currentAudioNameProperty() { return
currentAudioName; }
    public StringProperty currentAudioArtistProperty() { return
currentAudioArtist; }
    public StringProperty currentAlbumNameProperty() { return
currentAlbumName; }

    /** Play/pause toggle when the same queue slot is already loaded;
otherwise start playback. */
    public void playOrPause(Integer id, String path) {
        PlaybackTrackRef.parse(id, path).ifPresentOrElse(
                this::playOrPause,
                () -> LOGGER.warning("playOrPause called with null id or
path"));
    }

    public void playOrPause(PlaybackTrackRef track) {
        LOGGER.info(">>> playOrPause called with id=" + track.audioId()
                + ", current=" + currentAudioId);
        if (isSamePlaybackSlot(track.audioId())) {
            toggleLoadedTrackPlayback(track.audioId());
            return;
        }
        startPlayback(track, QueueRowIndex.fromCurrentQueue());
    }

    /** Plays queue row {@code row} from the start (queue advance /
repeat). */
    public void playQueueEntry(QueueRowIndex row) {
        resolveQueueEntry(row, this::playQueueEntry);
    }

    /** Loads queue row {@code row} paused at 0 when the current track was
removed. */
    public void loadQueueEntry(QueueRowIndex row) {
        resolveQueueEntry(row, (track, _) -> {
            loadTrack(track, PlaybackPosition.START);
            saveSelectedTrackToPlayerState();
        });
    }
```

```
    @FunctionalInterface
    private interface QueueEntryAction {
        void run(PlaybackTrackRef track, QueueRowIndex row);
    }

    private void resolveQueueEntry(QueueRowIndex row, QueueEntryAction
action) {
        QueueService queueService = QueueService.getInstance();
        if (!row.isValidForQueueSize(queueService.getQueue().size())) {
            return;
        }
        Audio atRow = queueService.getQueue().get(row.value());
        if (QueueAudioStubs.isIdOnly(atRow)) {
            queueService.hydrateQueueEntryAsync(row, hydrated ->
                    PlaybackTrackRef.from(hydrated).ifPresent(track -> {
                        queueService.setCurrentIndex(row);
                        currentQueueIndex = row.value();
                        action.run(track, row);
                    }));
            return;
        }
        PlaybackTrackRef.from(atRow).ifPresent(track -> {
            queueService.setCurrentIndex(row);
            currentQueueIndex = row.value();
            action.run(track, row);
        });
    }

    private void playQueueEntry(PlaybackTrackRef track, QueueRowIndex row)
{
        if (Objects.equals(track.audioId(), currentAudioId) &&
currentPlayer != null) {
            restartLoadedTrackFromBeginning(track.audioId(), row.value());
            return;
        }
        startPlayback(track, row);
    }

    /**
     * Session restore: mounts using already-loaded metadata (no second DB
fetch, no cover blob).
     */
    public void restoreTrack(Audio audio, double seekPosition) {
        PlaybackTrackRef.from(audio).ifPresent(track -> {
            File file = new File(track.filePath());
            if (!file.exists()) {
                return;
            }
            tryMountAtPosition(
                    new LoadedTrack(audio, file),
                    track,
                    QueueRowIndex.fromCurrentQueue(),
                    PlaybackPosition.ofSeconds(seekPosition),
                    "Failed to restore track");
        });
    }

    public void loadTrack(PlaybackTrackRef track, PlaybackPosition
position) {
        resolveLoadedTrack(track, false).ifPresent(loaded ->
```

```
                tryMountAtPosition(
                        loaded,
                        track,
                        QueueRowIndex.fromCurrentQueue(),
                        position,
                        "Failed to load track"));
    }

    /** Debounced seek; queues {@code pendingSeek} when no player is
mounted yet. */
    public void seekTo(double seconds) {
        seekTo(PlaybackPosition.ofSeconds(seconds));
    }

    public void seekTo(PlaybackPosition position) {
        if (currentPlayer == null) {
            pendingSeek = position.seconds();
            currentTime.set(position.seconds());
            return;
        }
        lastRequestedSeekSeconds = position.seconds();
        var old = pendingSeekTask.getAndSet(
                seekExecutor.schedule(() -> {
                    Double target = lastRequestedSeekSeconds;
                    if (target != null && currentPlayer != null) {
                        Platform.runLater(() -> {
                            if (currentPlayer != null) {
                                currentPlayer.seek(target);
                                saveCurrentPosition();
                            }
                        });
                    }
                }, SEEK_DEBOUNCE_MS,
java.util.concurrent.TimeUnit.MILLISECONDS)
        );
        if (old != null) old.cancel(false);
    }

    /** Seek to 0 and play (REPEAT_ONE without remounting). */
    public void replay() {
        if (currentPlayer == null) return;
        currentPlayer.seek(0);
        currentPlayer.play();
    }

    /** Maximum output level (125%). */
    public static final double MAX_VOLUME = 1.25;

    public void setVolume(double vol) {
        vol = Math.clamp(vol, 0.0, MAX_VOLUME);
        volume.set(vol);
        applyVolumeToCurrentPlayer();
        saveVolume(vol);
    }

    private void applyVolumeToCurrentPlayer() {
        if (currentPlayer != null) {
            currentPlayer.setVolume(volume.get());
        }
    }
```

```
    public void saveVolume(double vol) {
        ensurePlayerStateDAO();
        if (playerStateDAO != null) {
            try {
                playerStateDAO.saveVolume(vol);
            } catch (SQLException e) {
                LOGGER.warning("Failed to save volume: " + e.getMessage());
            }
        }
    }

    /** Disposes the player and clears observable state. */
    public void stop() { disposeCurrentPlayer(); }

    /** Stops playback when the current track id was deleted from the
library. */
    public void stopIfCurrentIdDeleted(Collection<Integer> deletedIds) {
        if (currentAudioId != null && deletedIds.contains(currentAudioId))
{
            disposeCurrentPlayer();
        }
    }

    /** Persists position and closes the player on JVM exit. */
    public void installShutdownHook() {
        Runtime.getRuntime().addShutdownHook(new Thread(() -> {
            if (currentPlayer != null) {
                saveCurrentPosition();
                currentPlayer.close();
            }
        }));
    }

    /** Updates {@link #currentQueueIndex} after queue reorder while the
same file keeps playing. */
    public void syncPlaybackQueueIndex() {
        if (currentAudioId == null) {
            return;
        }
        QueueRowIndex resolved = QueueService.getInstance()

.resolvePlayingQueueIndex(QueueRowIndex.of(currentQueueIndex),
currentAudioId);
        if (resolved.isAssigned()) {
            currentQueueIndex = resolved.value();
        }
    }

    private boolean isCurrentPlayerPlaying() {
        return currentPlayer != null
                && currentPlayer.statusProperty().get() ==
Mp3Player.Status.PLAYING;
    }

    private void toggleLoadedTrackPlayback(int audioId) {
        if (isCurrentPlayerPlaying()) {
            pauseLoadedTrack();
        } else {
            resumeLoadedTrack(audioId);
        }
    }
```

```
    private void pauseLoadedTrack() {
        currentPlayer.pause();
        saveCurrentPosition();
    }

    private void resumeLoadedTrack(int audioId) {
        attachEndOfMediaHandler(audioId, currentQueueIndex);
        currentPlayer.play();
    }

    private void restartLoadedTrackFromBeginning(int audioId, int
queueIndex) {
        attachEndOfMediaHandler(audioId, queueIndex);
        currentPlayer.seek(0);
        currentPlayer.play();
        recordPlayHistory(audioId);
    }

    private boolean isSamePlaybackSlot(int audioId) {
        if (!Objects.equals(audioId, currentAudioId) || currentPlayer ==
null) {
            return false;
        }
        if (matchesCurrentQueueRow(audioId)) {
            return true;
        }
        return matchesCachedQueueIndexFallback();
    }

    private boolean matchesCurrentQueueRow(int audioId) {
        QueueService qs = QueueService.getInstance();
        QueueRowIndex row = qs.getCurrentQueueRow();
        if (!row.isValidForQueueSize(qs.getQueue().size())) {
            return false;
        }
        Audio atRow = qs.getQueue().get(row.value());
        if (atRow != null && Objects.equals(atRow.getId(), audioId)) {
            currentQueueIndex = row.value();
            return true;
        }
        return false;
    }

    private boolean matchesCachedQueueIndexFallback() {
        QueueService qs = QueueService.getInstance();
        QueueRowIndex row = qs.getCurrentQueueRow();
        if (!row.isAssigned()) {
            return currentQueueIndex < 0;
        }
        return currentQueueIndex >= 0 && currentQueueIndex == row.value();
    }

    private void startPlayback(PlaybackTrackRef track, QueueRowIndex
queueIndex) {
        resolveLoadedTrack(track, true).ifPresent(loaded -> {
            tryMountAtPosition(
                    loaded,
                    track,
                    queueIndex,
                    PlaybackPosition.START,
```

```
                    "Failed to create Mp3Player for: " + track.filePath());
            if (currentPlayer == null) {
                return;
            }
            recordPlayHistory(track.audioId());
            currentPlayer.play();
        });
    }

    private record LoadedTrack(Audio metadata, File file) {}


    private void tryMountAtPosition(
            LoadedTrack loaded,
            PlaybackTrackRef track,
            QueueRowIndex queueIndex,
            PlaybackPosition position,
            String failureMessage
    ) {
        disposeCurrentPlayer();
        try {
            mountPlayer(loaded, track, queueIndex);
            applyInitialAndPendingSeek(position);
        } catch (Exception e) {
            LOGGER.severe(failureMessage + ": " + e.getMessage());
            disposeCurrentPlayer();
        }
    }

    private Optional<LoadedTrack> resolveLoadedTrack(PlaybackTrackRef
track, boolean includeCover) {
        return loadAudioMetadata(track.audioId(),
includeCover).flatMap(audio -> resolveExistingAudioFile(track)
                .map(file -> new LoadedTrack(audio, file)));
    }

    private Optional<Audio> loadAudioMetadata(int audioId, boolean
includeCover) {
        AudioDAO audioDAO = getAudioDAO();
        if (audioDAO == null) {
            LOGGER.severe("AudioDAO not available");
            return Optional.empty();
        }
        try {
            Audio audio = audioDAO.findById(audioId, includeCover);
            if (audio == null) {
                LOGGER.warning("No audio found with id " + audioId);
            }
            return Optional.ofNullable(audio);
        } catch (SQLException e) {
            LOGGER.severe("Failed to fetch audio for id " + audioId + ": "
+ e.getMessage());
            return Optional.empty();
        }
    }

    private Optional<File> resolveExistingAudioFile(PlaybackTrackRef track)
{
        File file = new File(track.filePath());
        if (!file.exists()) {
            LOGGER.warning("Audio file not found: " + track.filePath());
```

```
            notifyMissingAudioFile(track);
            return Optional.empty();
        }
        missingAudioFileNotifiedPaths.remove(track.filePath());
        return Optional.of(file);
    }

    private void mountPlayer(LoadedTrack loaded, PlaybackTrackRef track,
QueueRowIndex queueIndex)
            throws Exception {
        currentPlayer = createPlayerForFile(loaded.file());
        LOGGER.info("Active playback backend: "
                + currentPlayer.getClass().getSimpleName()
                + " for " + loaded.file().getName());
        currentAudioId = track.audioId();
        currentAudioPath = track.filePath();
        currentQueueIndex = queueIndex.value();
        applyMetadata(loaded.metadata());
        bindPlayerProperties();
        attachStatusListener(track.audioId());
        attachEndOfMediaHandler(track.audioId(), queueIndex.value());
        applyVolumeToCurrentPlayer();
    }

    private void applyInitialAndPendingSeek(PlaybackPosition
initialPosition) {
        if (currentPlayer == null) {
            return;
        }
        if (initialPosition.isAfterStart()) {
            applySeekWhenIdle(initialPosition.seconds());
        }
        if (pendingSeek >= 0) {
            applySeekWhenIdle(pendingSeek);
            pendingSeek = -1;
        }
    }

    /** Session restore and idle seeks avoid {@link Mp3Player#seek}
teardown before first play. */
    private void applySeekWhenIdle(double seconds) {
        if (currentPlayer == null) {
            return;
        }
        if (isCurrentPlayerPlaying()) {
            currentPlayer.seek(seconds);
        } else {
            currentPlayer.prepareStartPosition(seconds);
        }
    }

    private void recordPlayHistory(int audioId) {
        try {
            DatabaseHelper.getInstance().getPlayHistoryDAO().insert(new
PlayHistory(audioId));
            PlayHistoryService.getInstance().notifyPlayed(audioId);
        } catch (SQLException e) {
            LOGGER.warning("Failed to record play history for id " +
audioId + ": " + e.getMessage());
        }
    }
```

```
    private void applyMetadata(Audio audio) {
        currentAudioName.set(audio.getName() != null
                ? audio.getName() : "Unknown Title");
        currentAudioArtist.set(audio.getArtistNames() != null
                ? audio.getArtistNames() : "Unknown Artist");
        currentAlbumName.set(audio.getAlbum() != null
                ? audio.getAlbum().getName() : "");
    }

    /** Binds timeline properties; volume stays service-owned to avoid per-
player reset to 1.0. */
    private void bindPlayerProperties() {
        currentTime.bind(currentPlayer.currentTimeProperty());
        totalDuration.bind(currentPlayer.totalDurationProperty());
    }

    /** Persists position on pause/stop and on a timer while playing;
notifies status listeners. */
    private void attachStatusListener(Integer audioId) {
        currentPlayer.statusProperty().addListener((_, _, newStatus) -> {
            LOGGER.fine("Player status → " + newStatus + " audioId=" +
audioId);
            if (newStatus == Mp3Player.Status.PLAYING) {
                schedulePeriodicPositionSave();
            } else {
                cancelPeriodicPositionSave();
                if (shouldPersistPositionOnStatus(newStatus)) {
                    saveCurrentPosition();
                }
            }
            for (BiConsumer<Integer, Mp3Player.Status> l : statusListeners)
{
                Platform.runLater(() -> l.accept(audioId, newStatus));
            }
        });
    }

    /** Stops, saves position, and notifies end-of-media listeners (no
queue policy here). */
    private void attachEndOfMediaHandler(Integer audioId, int queueIndex) {
        final int capturedQueueIndex = queueIndex;
        currentPlayer.setOnEndOfMedia(player -> {
            player.stop();
            saveCurrentPosition();
            int finishedQueueIndex = QueueService.getInstance()

.resolvePlayingQueueIndex(QueueRowIndex.of(capturedQueueIndex), audioId)
                    .value();
            for (BiConsumer<Integer, Integer> l : endOfMediaListeners) {
                Platform.runLater(() -> l.accept(audioId,
finishedQueueIndex));
            }
        });
    }

    private static boolean shouldPersistPositionOnStatus(Mp3Player.Status
status) {
        return status == Mp3Player.Status.PAUSED || status ==
Mp3Player.Status.STOPPED;
    }
```

```
    private boolean canPersistCurrentPosition() {
        return playerStateDAO != null && currentAudioId != null &&
currentPlayer != null;
    }

    private void saveCurrentPosition() {
        if (!canPersistCurrentPosition()) {
            return;
        }
        savePositionToPlayerState(currentTime.get());
    }

    private void saveSelectedTrackToPlayerState() {
        savePositionToPlayerState(0.0);
    }

    private void savePositionToPlayerState(double positionSeconds) {
        ensurePlayerStateDAO();
        if (playerStateDAO == null || currentAudioId == null) {
            return;
        }
        try {
            playerStateDAO.savePosition(currentAudioId, positionSeconds);
        } catch (SQLException e) {
            LOGGER.warning("Failed to save position: " + e.getMessage());
        }
    }

    private void schedulePeriodicPositionSave() {
        cancelPeriodicPositionSave();
        java.util.concurrent.ScheduledFuture<?> fut =
seekExecutor.scheduleAtFixedRate(
                () -> Platform.runLater(() -> {
                    if (isCurrentPlayerPlaying()) {
                        saveCurrentPosition();
                    }
                }),
                POSITION_SAVE_INTERVAL_SECONDS,
                POSITION_SAVE_INTERVAL_SECONDS,
                java.util.concurrent.TimeUnit.SECONDS);
        periodicPositionSave.set(fut);
    }

    private void cancelPeriodicPositionSave() {
        java.util.concurrent.ScheduledFuture<?> f =
periodicPositionSave.getAndSet(null);
        if (f != null) f.cancel(false);
    }

    private void ensurePlayerStateDAO() {
        if (playerStateDAO == null) {
            playerStateDAO =
DatabaseHelper.getInstance().getPlayerStateDAO();
        }
    }

    /** Stops seek debounce and periodic position-save tasks (app
shutdown). */
    public void shutdown() {
        cancelPeriodicPositionSave();
```

```
        java.util.concurrent.ScheduledFuture<?> pending =
pendingSeekTask.getAndSet(null);
        if (pending != null) {
            pending.cancel(false);
        }
        seekExecutor.shutdown();
    }

    private void disposeCurrentPlayer() {
        pendingSeek = -1;
        cancelPeriodicPositionSave();
        if (currentPlayer != null) {
            Integer stoppedId = currentAudioId;
            if (stoppedId != null) {
                for (BiConsumer<Integer, Mp3Player.Status> l :
statusListeners) {
                    Platform.runLater(() -> l.accept(stoppedId,
Mp3Player.Status.STOPPED));
                }
            }
            currentPlayer.close();
            currentPlayer = null;
        }

        currentAudioId      = null;
        currentAudioPath    = null;
        currentQueueIndex   = -1;
        currentTime.unbind();
        currentTime.set(0);
        totalDuration.unbind();
        totalDuration.set(0);
        currentAudioName.set("");
        currentAudioArtist.set("");
        currentAlbumName.set("");
    }

    private void notifyMissingAudioFile(PlaybackTrackRef track) {
        String path = track.filePath();
        if (!missingAudioFileNotifiedPaths.add(path)) {
            return;
        }

        Platform.runLater(() -> {
            for (BiConsumer<Integer, String> l : missingAudioFileListeners)
{
                l.accept(track.audioId(), path);
            }
        });
    }

    private Mp3Player createPlayerForFile(File file) throws Exception {
        if (isMpeg4Audio(file)) {
            LOGGER.info("Using JavaFX media backend for MPEG-4 audio: " +
file.getName());
            JavaFxMediaPlayerAdapter player = new
JavaFxMediaPlayerAdapter();
            player.open(file);
            return player;
        }

        if (AppPreferences.getInstance().isPreferJavaFxPlayback()) {
```

```
            return openWithJavaFxFallbackToSpi(file);
        }
        return openWithSpiFallbackToJavaFx(file);
    }

    private Mp3Player openWithSpiFallbackToJavaFx(File file) throws
Exception {
        return openWithPrimaryBackend(
                file,
                Mp3Player::new,
                JavaFxMediaPlayerAdapter::new,
                "Java Sound SPI backend",
                "JavaFX media");
    }

    private Mp3Player openWithJavaFxFallbackToSpi(File file) throws
Exception {
        return openWithPrimaryBackend(
                file,
                JavaFxMediaPlayerAdapter::new,
                Mp3Player::new,
                "JavaFX media backend",
                "Java Sound SPI");
    }

    private Mp3Player openWithPrimaryBackend(
            File file,
            Supplier<Mp3Player> primarySupplier,
            Supplier<Mp3Player> fallbackSupplier,
            String primaryLabel,
            String fallbackLabel
    ) throws Exception {
        try {
            LOGGER.info("Using " + primaryLabel + " for: " +
file.getName());
            Mp3Player player = primarySupplier.get();
            player.open(file);
            return player;
        } catch (Exception primaryError) {
            LOGGER.info(primaryLabel + " failed for " + file.getName()
                    + ", falling back to " + fallbackLabel + ": " +
primaryError.getMessage());
            Mp3Player player = fallbackSupplier.get();
            player.open(file);
            return player;
        }
    }

    private boolean isMpeg4Audio(File file) {
        String ext = extensionOf(file.getName());
        return MPEG4_AUDIO_EXTENSIONS.contains(ext);
    }

    private String extensionOf(String name) {
        int dot = name.lastIndexOf('.');
        if (dot < 0 || dot == name.length() - 1) return "";
        return name.substring(dot + 1).toLowerCase();
    }
}
```

```
package manicsalsa.soufone_proto.database.dao;

import java.sql.*;
import java.util.logging.Logger;

/**
 * Shared JDBC helpers for SQLite DAOs: parameter binding and insert/update
execution.
 */
public abstract class BaseDAO {
    protected final Connection connection;
    protected static final Logger LOGGER =
Logger.getLogger(BaseDAO.class.getName());

    public BaseDAO(Connection connection) {
        this.connection = connection;
    }

    /** Binds {@code params} to {@code stmt} positionally (1-based). */
    protected void setParameters(PreparedStatement stmt, Object... params)
throws SQLException {
        for (int i = 0; i < params.length; i++) {
            stmt.setObject(i + 1, params[i]);
        }
    }

    /** INSERT with {@code RETURN_GENERATED_KEYS}; returns the new row id
or null. */
    protected Integer executeInsert(String sql, Object... params) throws
SQLException {
        try (PreparedStatement stmt = connection.prepareStatement(sql,
Statement.RETURN_GENERATED_KEYS)) {
            setParameters(stmt, params);
            stmt.executeUpdate();

            try (ResultSet rs = stmt.getGeneratedKeys()) {
                if (rs.next()) {
                    return rs.getInt(1);
                }
            }
        }
        return null;
    }

    /** Prepared UPDATE/DELETE; returns affected row count. */
    protected int executeUpdate(String sql, Object... params) throws
SQLException {
        try (PreparedStatement stmt = connection.prepareStatement(sql)) {
            setParameters(stmt, params);
            return stmt.executeUpdate();
        }
    }
}
```