



# Hudební databáze

**Projekt do předmětu Databázové systémy II**

16. dubna 2023

Martin Minařík

<b>Historie dokumentu</b>	<b>3</b>
<b>1. Specifikace zadání</b>	<b>4</b>
Vize	4
Role	4
Vstupy	4
Výstupy	5
Funkce	6
<b>2. Datová analýza</b>	<b>6</b>
Konceptuální datový model	6
Relační datový model	7
Datový slovník	8
<b>3. Funkční analýza</b>	<b>14</b>
Seznam funkcí	14
Detailní popis funkcí	18
Funkce 7.1 - Vytvoření nové skladby	18
Funkce 7.2 - Seznam skladeb	19
Funkce 7.3 - Seznam skladeb k danému interpretu	21
Funkce 7.4 - Seznam skladeb k danému žánru	21
Funkce 7.5 - Detail skladby	21
Funkce 7.6 - Aktualizace skladby	22
Funkce 7.7 - Seznam historie aktualizací	23
Funkce 7.8 - Vygenerování řetězce v normalizovaném tvaru	23
<b>4. Návrh uživatelského rozhraní</b>	<b>24</b>
Seznam formulářů	24
Detailní popis formulářů	26
Formulář 4.1 - Přehled skladeb	26
Formulář 4.5 - Advanced search	27
Formulář 4.6 - Sekce komentářů	27
Formulář 4.7 - Vytvoření skladby	28
Formulář 4.9 - Vytvoření playlistu	28
Formulář 4.11 - Detail skladby	29
Formulář 4.13 - Detail playlistu	29
Formulář 4.14 - Skladby playlistu	30
Formulář 4.16 - Editace skladby	30
Formulář 4.19 - Přihlášení	31
Formulář 4.20 - Registrace	31
Formulář 4.21 - Detail uživatele	32
Formulář 4.22 - Přehled vlastních komentářů	32
Formulář 4.23 - Přehled vlastních playlistů	33

## Historie dokumentu

<b>Specifikace zadání</b>	15/10/2022
<b>Úprava specifikace zadání</b> <ul style="list-style-type: none"> <li>- Vstupy</li> <li>- Výstupy</li> <li>- funkce</li> </ul>	28/10/2022
<b>Úprava specifikace zadání</b> <ul style="list-style-type: none"> <li>- Vstupy, zjednodušení</li> </ul>	29/10/2022
<b>Datové modely</b>	12/11/2022
<b>Datové modely</b> - oprava	26/11/2022
Přepřarování <b>datových modelů</b> do Oracle Data Modeler	19/03/2023
<b>Funkční analýza</b>	19/03/2023
Návrh <b>uživatelského rozhraní</b>	19/03/2023
Předělání <b>funkční analýzy</b>	02/04/2023
Předělání <b>uživatelského rozhraní</b>	02/04/2023
<b>Zjednodušení</b> formuláře <b>advanced search spojeného s funkcí 7.2</b> a tedy i zjednodušení funkce 7.2 <ul style="list-style-type: none"> <li>- Jedná se o zjednodušení, aby to nebylo zbytečně moc komplikované</li> <li>- <b>Místo řetězce jmen interpretů</b> oddělených středníkem bude řetězec s který obsahuje <b>jedno jméno interpreta</b> <ul style="list-style-type: none"> <li>- Bylo třeba rozdělit daný řetězec, spočítat kolik obsahuje interpretů, dynamicky sestavit podmínku a dát to se zbytkem dohromady, případně použít nějaké pokročilejší věci z PL/SQL. To ale v PL/SQL nevypadalo dobře, o výkonu ani nemluvě. Spíš by se hodilo si do-pomoct pomoci nějakého redundantního atributu.</li> </ul> </li> <li>- <b>Místo řetězce názvů žánrů</b> oddělených středníkem bude řetězec s který obsahuje <b>jeden název žánru</b>. <ul style="list-style-type: none"> <li>- Stejná situace jako u jmen interpretů</li> </ul> </li> </ul>	16/04/2023
<b>Detailní popisy funkcí</b>	16/04/2023

# 1. Specifikace zadání

## Vize

Cílem je vytvořit informační systém pro informace okolo hudebních skladeb. Systém, kde uživatelé budou moci objevovat novou hudbu, hledat podobnou, doporučovat, sdílet, vytvářet playlisty, komentovat (uživatel se například může podělit o své pocity, názory), hodnotit a vyhledávat informace o skladbách a autorech.

## Role

V systému budou 4 základní role. **Návštěvník** (nepřihlášený uživatel, anonymní uživatel), který bude mít možnost vyhledávat a prohlížet obsah, ale nemá možnost nic přidávat. **Přihlášený uživatel** (registrovaný uživatel), který má navíc možnosti přidávat a editovat skladby a autory, vytvářet playlisty, komentovat a hodnotit. **Moderátor obsahu**, který bude kontrolovat a spravovat obsah a má právo zablokovat účet. **Administrátor** (správce), jenž přiděluje roli moderátora obsahu a řeší závažnější problémy, přidává žánry, ...

## Vstupy

	Povinně nás zajímá	Nepovinně nás zajímá
<b>Uživatelský účet</b>	Username, e-mail, heslo, časové razítko vytvoření, heslo v hash podobě, indikace zablokování a smazání	role, obrázek
<b>Skladba</b>	Název, jeden nebo více interpretů, časové razítko přidání	Popis, doba trvání, žádný nebo více žánrů, datum vydání, obrázek
<b>Interpret</b>	Jméno, časové razítko přidání	Popis, žádný nebo více žánrů, obrázek
<b>Playlist</b>	Název, zda je public nebo private, uživatelský účet (který ho vytvořil), časové razítko vytvoření	Popis, žádný nebo více žánrů, obrázek
<b>Playlist Položka</b>	Playlist do kterého patří, skladba, pořadí	_____
<b>Žánr</b>	Název	_____
<b>Hodnocení</b>	Autor, číselná hodnota, k čemu patří (skladba, interpret, playlist)	_____
<b>Komentář</b>	Autor, text, časové razítko, k čemu patří (skladba, interpret, playlist)	_____

Kterýkoliv návštěvník, může vytvořit **uživatelský účet**. Účet může mít roli administrátora nebo moderátora obsahu. Editovat účet bude mít právo pouze vlastník účtu. Smazat účet bude mít právo samotný uživatel nebo administrátor. Účet může být zablokován moderátorem obsahu.

Přidávat **skladby**, **interprety** a vytvářet **playlisty** může pouze přihlášený uživatel. Skladby a interprety může editovat, kterýkoliv přihlášený uživatel, nehledě na to kdo je přidal a odstranit je může pouze administrátor nebo moderátor obsahu. Playlisty může editovat a mazat pouze ten kdo je vytvořil, tedy jejich vlastník.

Skladba může mít více interpretů a interpret může mít více skladeb. Skladby, interpreti a playlisty mohou patřit do žádného nebo více žánrů a mohou mít mnoho **komentářů**, **hodnocení** od různých uživatelů.

Uživatel může hodnotit danou věc pouze jednou, ale komentovat vícekrát.

Jelikož skladby a interprety může editovat skoro kdokoliv, tak může docházet k vandalismu(editování obsahu se zlým úmyslem). Z tohoto důvodu bude u nich zavedena **historie změn**. Každá změna bude opatřena identifikátorem uživatelského účtu a časovým razítkem.

Vzhledem k pirátství a opatřením kolem, systém nebude evidovat soundtrack skladeb.

## Výstupy

Hlavním výstupem bude **zobrazení seznamu položek** s možností seřazení a filtrování. Primárně se bude jednat o skladby. Výstup bude možné seřadit podle hodnocení, počtu komentářů nebo od nejnovějších. Filtrovat bude možné na základě žánrů, délky skladby, interpretů, interpretů které uživatel sleduje a rozmezí dvou datumů.

Dalším výstupem bude **zobrazení obsahu k dané skladbě, autorovi, nebo playlistu**. To zahrnuje i komentáře a hodnocení. U obsahu skladby a interpretů bude dále možnost nahlédnout do historie změn.

## Funkce

System bude **evidovat historii změn** u nahrávek, autorů.

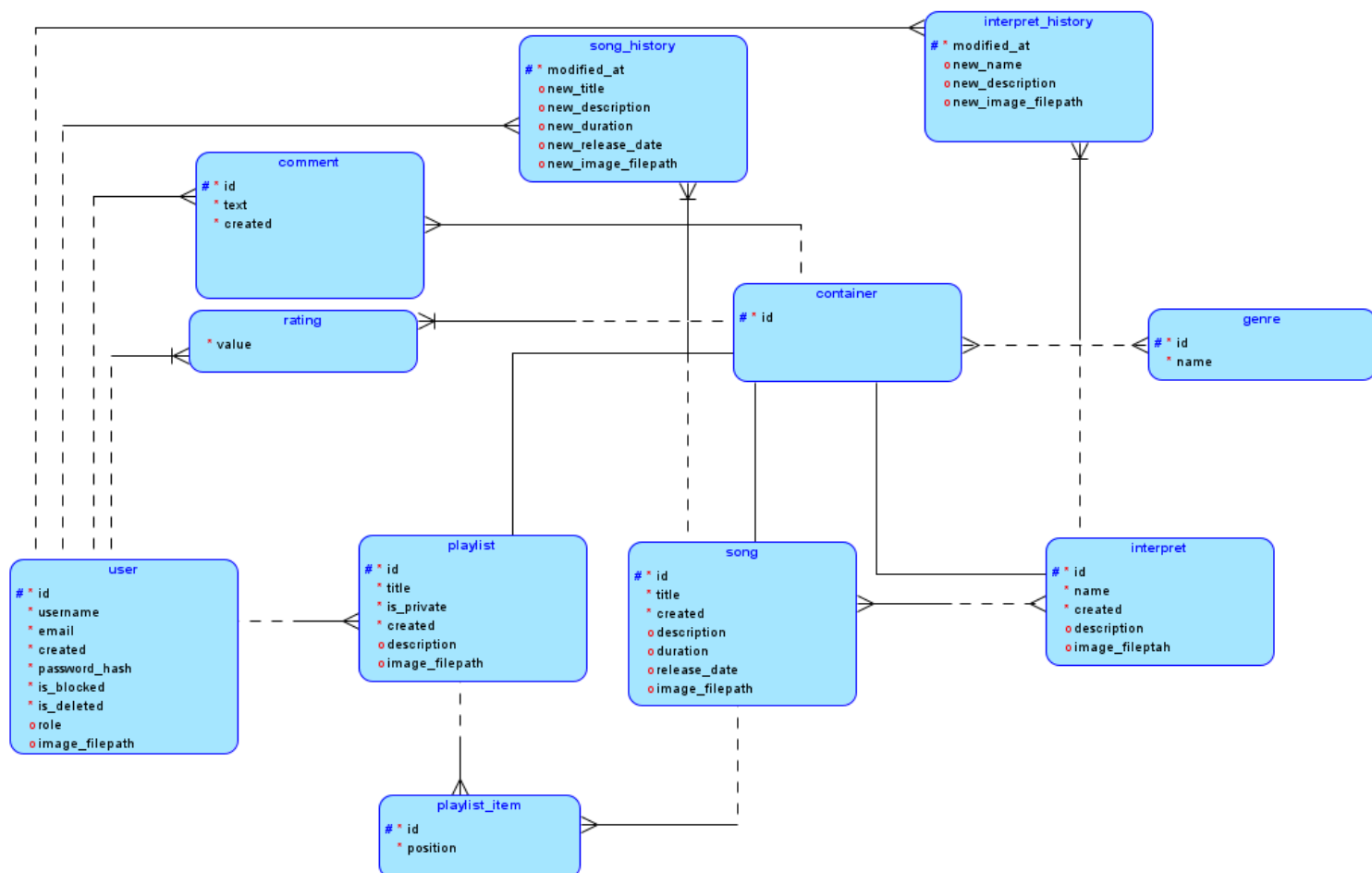
V případě obsahového vandalizmu(editování obsahu se zlým úmyslem) na více věcech, bude potřeba **funkce pro hromadné a jednoduché zrušení všech způsobených změn daným uživatelem** a uvedení dané věci do předchozího stavu.

Funkce, která **přidá skladbu jen v případě**, že se název a její interpreti neshodují s již evidovanou skladbou. Tedy nevznikne duplicitně evidovaná skladba.

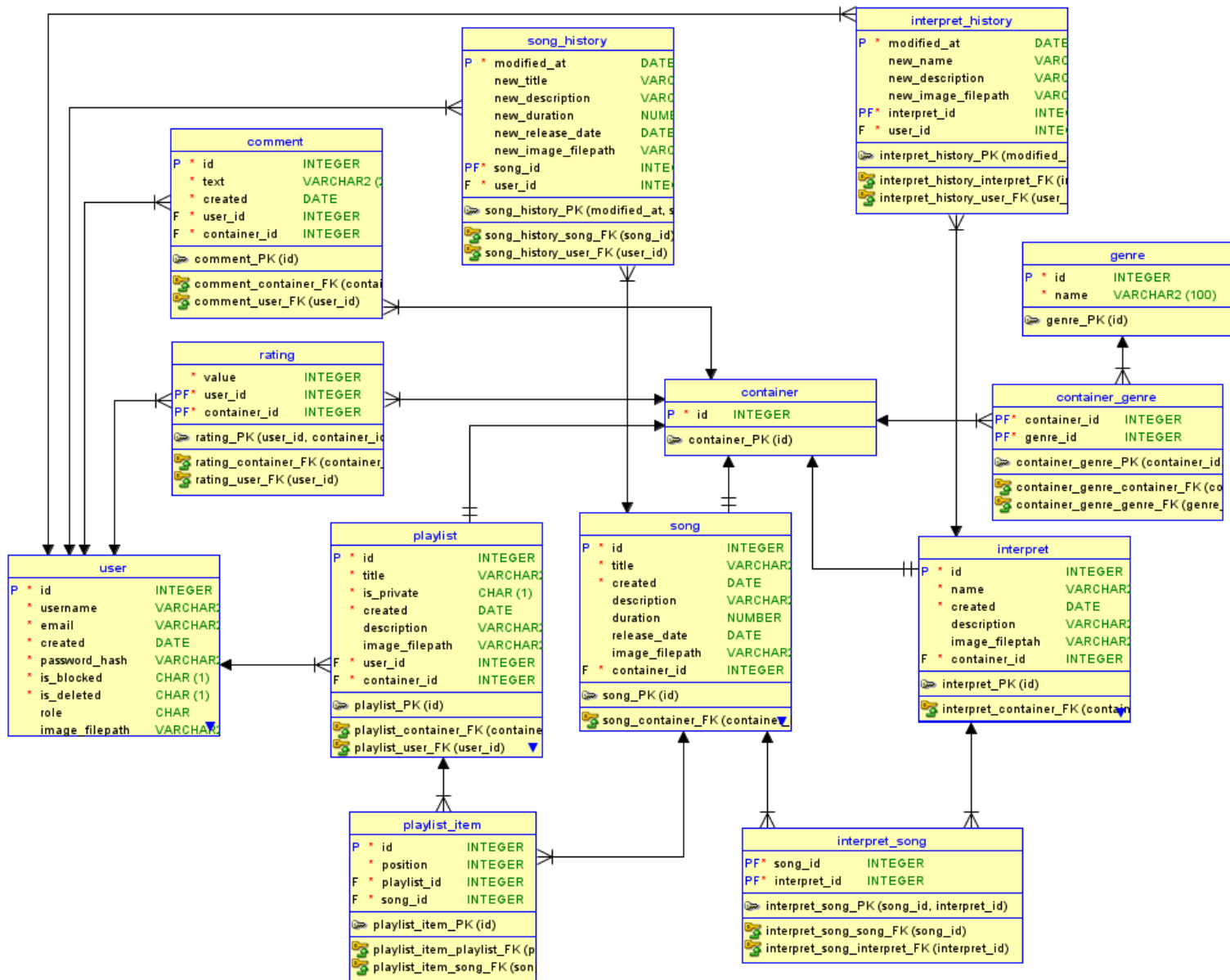
Funkce, která jedním voláním **odstraní uživatele a s ním spojené playlisty, komentáře, hodnocení**.

## 2. Datová analýza

### Konceptuální datový model



# Relační datový model



# Datový slovník

Popis jednotlivých tabulek

**Tabulka user**

Atribut	Datový typ	Délka	Klíč	Null	IO	Popis
id	int		Primární	Ne		Automaticky inkrementovaný PK
username	varchar	20		Ne		Uživatelské jméno uživatele
email	varchar	50		Ne	3	Emailová adresa uživatele
password_hash	varchar	16		Ne		Heslo v hash podobě
created	datetime			Ne	4	Čas a datum kdy byl účet vytvořen
role	char	1		Ano	1	Role uživatele
is_blocked	boolean			Ne	2	Identifikace zda je účet zablokován
is_deleted	boolean			Ne	2	Identifikace zda je účet smazán
image_filepath	varchar	260		Ano		Cesta k obrázku

**Tabulka container**

Atribut	Datový typ	Délka	Klíč	Null	IO	Popis
id	int		Primární	Ne		Automaticky inkrementovaný PK

**Tabulka rating**

Atribut	Datový typ	Délka	Klíč	Null	IO	Popis
value	int			Ne	5	Hodnota ohodnocení
container_id	int		Primární, Cizí(container)	Ne		Kontejner
user_id	int		Primární, Cizí(user)	Ne		Hodnotitel



**Tabulka comment**

Atribut	Datový typ	Délka	Klíč	Null	IO	Popis
id	int		Primární	Ne		Automaticky inkrementovaný PK
text	varchar	2000		Ne		Text komentáře
container_id	int		Primární, Cizí(container)	Ne		Kontejner
created	datetime			Ne	4	Čas a datum vytvoření
user_id	int		Cizí(user)	Ne		Autor komentáře

**Tabulka genre**

Atribut	Datový typ	Délka	Klíč	Null	IO	Popis
id	int		Primární	Ne		Automaticky inkrementovaný PK
name	varchar	100		Ne		Název žánru

**Tabulka interpret**

\*comment\_rating\_container

Atribut	Datový typ	Délka	Klíč	Null	IO	Popis
id	int		Primární	Ne		Automaticky inkrementovaný PK
name	varchar	100		Ne		Jméno, pseudonym
description	varchar	2000		Ano		Popis
container_id	int		Cizí(container)	Ne		Přidělený kontejner
created	datetime			Ne	4	Čas a datum přidání
image_filepath	varchar	260		Ano		Cesta k obrázku

**Tabulka song**

Atribut	Datový typ	Délka	Klíč	Null	IO	Popis
id	int		Primární	Ne		Automaticky inkrementovaný PK
title	varchar	150		Ne		Název
description	varchar	2000		Ano		Popis
release_date	datetime			Ano		Datum vydání
duration	double			Ano		Doba trvání
container_id	int		Cizí(container)	Ne		Přidělený kontejner
created	datetime			Ne	4	Čas a datum přidání
image_filepath	varchar	260		Ano		Cesta k obrázku

**Tabulka interpret\_song**

Atribut	Datový typ	Délka	Klíč	Null	IO	Popis
interpret_id	int		Primární, Cizí (interpret)	Ne		Interpret
song_id	int		Primární, Cizí (song)	Ne		Skladba

**Tabulka playlist\_item**

Atribut	Datový typ	Délka	Klíč	Null	IO	Popis
id	int		Primární	Ne		Automaticky inkrementovaný PK
position	int			Ne		Pozice v playlistu
song_id	int		Cizí (song)	Ne		Skladba
playlist_id	int		Cizí (playlist)	Ne		Playlist

**Tabulka playlist**

Atribut	Datový typ	Délka	Klíč	Null	IO	Popis
id	int		Primární	Ne		Automaticky inkrementovaný PK
title	varchar	150		Ne		
user_id	int			Ne		Uživatel, vlastník playlistu
description	varchar	2000		Ano		Popis
is_private	boolean			Ne		Identifikátor zda je playlist public nebo private
created	datetime			Ne		Čas a datum přidání
image_filepath	varchar	260		Ano		Cesta k obrázku
container_id	id		Cizí(container)	Ne		Přidělený kontejner

**Tabulka container\_genre**

Atribut	Datový typ	Délka	Klíč	Null	IO	Popis
container_id	int		Primární, Cizí(container)	Ne		Kontejner
genre_id	int		Primární, Cizí(genre)	Ne		Žánr

**Tabulka intepret\_history**

Atribut	Datový typ	Délka	Klíč	Null	IO	Popis
intepret_id	int		Primární	Ne		Interpret u kterého proběhla změna
modified_at	datetime			Ne		Čas a datum změny
new_name	varchar	100		Ano		Nový název
new_description	varchar	2000		Ano		Nový popis
new_image_file path	varchar	260		Ano		Nová cesta k obrázku
user_id	int		Cizí(user)	Ne		Uživatel, který provedl změnu

**Tabulka song\_history**

Atribut	Datový typ	Délka	Klíč	Null	IO	Popis
songt_id	int		Primární	Ne		Skladba u které proběhla změna
modified_at	datetime			Ne		Čas a datum změny
new_title	varchar	150		Ano		Nový název
new_description	varchar	2000		Ano		Nový popis
new_duration	double			Ano		Nová délka trvání
new_release_date	date			Ano		Nové datum vydání
new_image_filepath	varchar	260		Ano		Cesta k obrázku
user_id	int		Cizí(user)	Ne		Uživatel, který provedl změnu

**Integritní omezení:**

1. Role musí mít hodnotu „A“ pro administrátora, „M“ pro moderátora obsahu nebo NULL pro běžného uživatele
2. Výchozí hodnota je nastavena na FALSE
3. Musí být ve tvaru <name>@<domain>
4. Defaultně nastavené na čas vzniku
5. V rozmezí 0 až 5

### 3. Funkční analýza

*\*Poznámka: Obsahem je myšlena ať už skladba, interpret nebo playlist.*

*\*Poznámka: tučným(boldem) jsou označené netriviální funkce a transakce*

#### Seznam funkcí

##### 1. Evidence uživatelů

- 1.1. Vytvoření nového uživatele - registrace  
*Zodpovědnost: nepřihlášený uživatel*
- 1.2. Seznam uživatelů  
*Zodpovědnost: Správce*
- 1.3. Detail uživatele  
*Zodpovědnost: Správce a přihlášený uživatel(pouze svůj záznam)*
- 1.4. Aktualizace uživatele  
*Zodpovědnost: Správce a přihlášený uživatel(pouze svůj záznam)*
- 1.5. Smazání uživatele - nastavení atributu „is\_deleted“ na TRUE
- 1.6. Zablokování uživatele - nastavení atributu „is\_blocked“ na TRUE  
*Zodpovědnost: Správce a moderátor obsahu*
- 1.7. Seznam rolí - podle IO 1  
*Zodpovědnost: Správce*
- 1.8. Porovnání hesla v hash podobě s atributem „password\_hash“ - slouží u loginu  
*Zodpovědnost: Nepřihlášený uživatel*
- 1.9. **Funkce login** - Netriviální funkce. Zkontrolování zda zda uživatel existuje a porovnání hesla.  
*Zodpovědnost: Nepřihlášený uživatel*

##### 2. Evidence komentářů

- 2.1. Vytvoření nového komentáře  
*Zodpovědnost: přihlášený uživatel*
- 2.2. Seznam komentářů k danému obsahu  
*Zodpovědnost: kdokoliv*
- 2.3. Seznam komentářů daného uživatele  
*Zodpovědnost: Správce, moderátor obsahu a přihlášený uživatel(pouze své záznamy)*
- 2.4. Smazání komentáře  
*Zodpovědnost: Správce, moderátor obsahu a přihlášený uživatel(pouze své záznamy)*

### 3. Evidence hodnocení

- 3.1. **Přidání hodnocení** - transakce. Vytvoření hodnocení, aktualizování atributu *average\_rating* s pomocí funkce 3.4 tabulky container.

*Poznámka: Očekává se, že tabulka container bude obsahovat redundantní atribut average\_rating.*

*Zodpovědnost: přihlášený uživatel*

- 3.2. Aktualizace hodnocení

*Zodpovědnost: přihlášený uživatel(pouze svůj záznam)*

- 3.3. Detail hodnocení

*Zodpovědnost: přihlášený uživatel(pouze svůj záznam)*

- 3.4. Spočítání průměrného hodnocení k danému obsahu

*Zodpovědnost: kdokoliv*

### 4. Evidence žánrů

- 4.1. Seznam žánrů - s filtrací a seřazením

*Zodpovědnost: kdokoliv*

- 4.2. Seznam žánrů daného obsahu

*Zodpovědnost: kdokoliv*

- 4.3. Přidání žánru

*Zodpovědnost: Správce*

- 4.4. Aktualizace žánru

*Zodpovědnost: Správce*

- 4.5. Odstranění žánru

*Zodpovědnost: správce*

- 4.6. Seznam žánrů daného obsahu v podobě řetězce - spojí názvy žánrů daného obsahu do jednoho řetězce a oddělí je středníkem

*Zodpovědnost: kdokoliv*

- 4.7. **Přidání žánrů od daného obsahu na základě řetězce** - netriviální dotaz. Rozdělí řetězec s názvy žánrů oddělenými středníkem a vytvoří nové záznamy ve vazební tabulce mezi kontejnerem a žánry.

*Zodpovědnost: přihlášený uživatel*

- 4.8. Odebrání žánrů od daného obsahu na základě řetězce - Rozdělí řetězec s názvy žánrů oddělenými středníkem a pokud najde záznam ve vazební tabulce s konkrétním žánrem, tak ho odstraní.

*Zodpovědnost: přihlášený uživatel*

### 5. Evidence playlistů

- 5.1. Vytvoření nového playlistu

*Zodpovědnost: přihlášený uživatel*

- 5.2. Seznam playlistů - s filtrací a seřazením  
*Zodpovědnost: kdokoliv*
- 5.3. Seznam playlistů od uživatele  
*Zodpovědnost: přihlášený uživatel(své vlastní záznamy)*
- 5.4. Seznam playlist itemů  
*Zodpovědnost: kdokoliv*
- 5.5. Získání počtu playlist itemů - možnost pro filtraci ve funkci 5.2. Např. filtrace na základě minimálního počtu playlist itemů.  
*Zodpovědnost: kdokoliv*
- 5.6. Detail playlistu  
*Zodpovědnost: kdokoliv*
- 5.7. Aktualizace playlistu  
*Zodpovědnost: přihlášený uživatel(své vlastní záznamy)*
- 5.8. Odstranění playlistu - smazání i playlist\_itemů s ním spojených - transakce**  
*Zodpovědnost: přihlášený uživatel(své vlastní záznamy)*
- 5.9. Přidání playlist itemu do playlistu  
*Zodpovědnost: přihlášený uživatel(své vlastní záznamy)*
- 5.10. Odstranění playlist itemu  
*Zodpovědnost: přihlášený uživatel(své vlastní záznamy)*
- 5.11. **Odstranění duplicitních playlist itemů v playlistu** - netriviální dotaz - transakce  
*Zodpovědnost: přihlášený uživatel(své vlastní záznamy)*

## 6. Evidence Interpreta

- 6.1. Vytvoření nového interpreta  
*Zodpovědnost: přihlášený uživatel*
- 6.2. Seznam interpretů - s filtrací a sortováním  
*Zodpovědnost: kdokoliv*
- 6.3. Seznam historie aktualizací  
*Zodpovědnost: přihlášený uživatel*
- 6.4. Seznam interpretů dané skladby  
*Zodpovědnost: kdokoliv*
- 6.5. Seznam interpretů dané skladby v podobě řetězce- spojí názvy interpretů dané skladby do jednoho řetězce a oddělí je středníkem  
*Zodpovědnost: přihlášený uživatel*
- 6.6. Detail interpreta  
*Zodpovědnost: kdokoliv*
- 6.7. Aktualizace interpreta - zároveň vytvoření nového záznamu v „interpret history“  
*Zodpovědnost: přihlášený uživatel*



- 6.8. Přidání interpretů od dané skladby na základě řetězce - viz funkci 4.7., obdobná logika  
*Zodpovědnost: přihlášený uživatel*
- 6.9. Odebrání interpretů od dané skladby na základě řetězce - viz funkci 4.8., obdobná logika  
*Zodpovědnost: přihlášený uživatel*

## 7. Evidence skladby

- 7.1. **Vytvoření nové skladby** - transakce  
*Zodpovědnost: přihlášený uživatel*
- 7.2. **Seznam skladeb** - s filtrací a seřazením, netriviální dotaz  
*Zodpovědnost: kdokoliv*
- 7.3. Seznam skladeb k danému interpretu  
*Zodpovědnost: kdokoliv*
- 7.4. Seznam skladeb k danému žánru  
*Zodpovědnost: kdokoliv*
- 7.5. Detail skladby  
*Zodpovědnost: kdokoliv*
- 7.6. **Aktualizace skladby** - transakce, aktualizace atributů, vytvoření nového záznamu v „song\_history“ a kontrola zda jeho interpreti a žánry existují.  
*Zodpovědnost: přihlášený uživatel*
- 7.7. Seznam historie aktualizací  
*Zodpovědnost: přihlášený uživatel*
- 7.8. **Vygenerování řetězce v normalizovaném formátu** - netriviální dotaz. Ke skladbě vrací řetězec ve formátu "<interprets> - <song title>", interpreti jsou odděleni ampersandem.  
*Zodpovědnost: kdokoliv*

## 8. Evidence kontejneru

- 8.1. Vytvoření nového kontejneru - vytvoří kontejner a vrátí jeho id
- 8.2. **Získání typu obsahu** - netriviální dotaz, slouží k získání typu obsahu na základě id kontejneru. Vrací řetězec s názvem typu. Může se například hodit když potřebujeme ke komentáři najít typ obsahu a podle typu obsahu a id kontejneru najít specifický obsah.  
*Zodpovědnost: přihlášený uživatel*

## Detailní popis funkcí

### Funkce 7.1 - Vytvoření nové skladby

#### Vstupy

- #user\_id - ID uživatele
- #title - název skladby
- #duration - doba trvání skladby nebo NULL
- #release\_date - datum vydání skladby nebo NULL
- #description - popis skladby nebo NULL
- #image\_filepath - cesta k obrázku skladby nebo NULL

#### Výstupy

- #error\_msg - výstupní chybové hlášení (inicializováno na NULL)

**Popis:** Funkce vytvoří nový kontejner pro skladbu, nový záznam skladby a první záznam v historii dané skladby. Funkce bude řešena jako transakce.

1. Do proměnné #now se uloží aktuální časové razítko CURRENT\_TIMESTAMP
2. Vytvoření nového záznamu v tabulce container a uložení jeho id do proměnné #container\_id

```
INSERT INTO container(id) VALUES (DEFAULT) RETURNING id INTO #container_id;
```

3. Vytvoření nového záznamu v tabulce song a uložení jeho id do proměnné #song\_id

```
INSERT INTO song(id, title, description, release_date, duration,
container_id, created, image_filepath)
VALUES (DEFAULT, #title, #description, #release_date, #duration,
#container_id, #now, #image_filepath)
RETURNING id INTO #song_id;
```

4. Vytvoření nového záznamu v tabulce song\_history

```
INSERT INTO song_history(song_id, modified_at, new_title, new_description,
new_duration, new_release_date, new_image_filepath, user_id)
VALUES (#song_id, #now, #new_title, #new_description, #new_duration,
#new_release_date, #new_image_filepath, #user_id);
```

**Poznámka:** Pokud něco selže, nastaví se chybové hlášení #error\_msg na „Nepodařilo se vytvořit skladbu.“ a transakci ukončíme

## Funkce 7.2 - Seznam skladeb

### Vstupy

- #keyword - řetězec hledaný v názvu skladby nebo NULL
- #min\_rating - minimální hodnocení nebo NULL
- #max\_rating - maximální hodnocení nebo NULL
- #min\_number\_of\_comments - minimální počet komentářů nebo NULL
- #interpret - jméno interpreta nebo NULL
- #genre - název žánru nebo NULL
- #min\_duration - minimální doba trvání nebo NULL
- #max\_duration - maximální doba trvání nebo NULL
- #created\_before - datum před kterým skladba byla přidána nebo NULL
- #created\_after - datum po kterém skladba byla přidána nebo NULL
- #released\_before - datum před kterým skladba byla vydána nebo NULL
- #released\_after - datum po kterém skladba byla vydána nebo NULL
- #order\_by - seřazení podle jedné z možností
  - možnosti: ["title", "rating", "duration", "created", "release\_date"]

### Výstupy

- #error\_msg - výstupní chybové hlášení (inicializováno na NULL)
- seznam skladeb

**Popis:** Funkce vrací seznam skladeb společně. Podmínka WHERE je dynamicky sestavena na základě vstupů. Kontrolují se rozmezí na vstupu.

1. Jestliže neplatí  $0 \leq \text{min\_rating} \leq \text{max\_rating} \leq 5$ , nastavíme chybové hlášení #error\_msg na „Neplatný rozsah hodnocení.“ a funkci ukončíme.
2. Jestliže #min\_number\_of\_comments < 0, nastavíme chybové hlášení #error\_msg na „Počet komentářů nemůže být záporný.“ a funkci ukončíme.
3. Jestliže neplatí  $0 \leq \text{min\_duration} \leq \text{max\_duration}$ , nastavíme chybové hlášení #error\_msg na „Neplatný rozsah doby trvání“ a funkci ukončíme.
4. Seznam skladeb

```
SELECT song.id, song.title, song.image_filepath
FROM song
      JOIN container ON container.id = song.container_id
WHERE (viz sestavení podmínky WHERE níže)
ORDER BY (CASE WHEN #order_by = 'title' THEN song.title END) ASC,
         (CASE WHEN #order_by = 'rating' THEN
           f_average_rating(container.id) END) DESC,
         (CASE WHEN #order_by = 'duration' THEN song.duration END) DESC,
         (CASE WHEN #order_by = 'created' THEN song.created END) DESC,
         (CASE WHEN #order_by = 'release_date' THEN
           song.release_date END) DESC;
```

**Podmínka WHERE** bude sestavena z následujících výrazů spojených logickým součinem (AND):

- Jestliže #keyword IS NOT NULL:  
`(song.title LIKE '%' + #keyword + '%')`
- Jestliže #min\_rating IS NOT NULL:  
`(#min_rating <= f_average_rating(container.id))`
- Jestliže #max\_rating IS NOT NULL:  
`(#max_rating >= f_average_rating(container.id))`
- Jestliže #min\_number\_of\_comments IS NOT NULL:  
`(min_number_of_comments >= (SELECT COUNT(*) FROM comment WHERE comment.container_id = container.id))`
- Jestliže #interpret IS NOT NULL:  
`(EXISTS(SELECT * FROM interpret_song is_ JOIN interpret on interpret.id = is_.interpret_id WHERE is_.song_id = song.id and interpret.name = #interpret))`
- Jestliže #genre IS NOT NULL:  
`(EXISTS(SELECT * FROM container_genre cg JOIN genre on genre.id = cg.genre_id WHERE cg.container_id = container.id AND genre.name = #genre))`
- Jestliže #min\_duration IS NOT NULL:  
`(#min_duration <= song.duration)`
- Jestliže #max\_duration IS NOT NULL:  
`(#max_duration >= song.duration)`
- Jestliže #created\_before IS NOT NULL:  
`(#created_before >= song.created)`
- Jestliže #created\_after IS NOT NULL:  
`(#created_after <= song.created)`
- Jestliže #released\_before IS NOT NULL:  
`(#released_before <= song.release_date)`
- Jestliže #released\_after IS NOT NULL:  
`(#released_after <= song.release_date)`

**Poznámka:** Funkce `f_average_rating()`, konkrétně funkce 3.4., vypočítává průměrné hodnocení k danému obsahu. Jedná se však o dost neefektivní řešení protože se průměrné skóre bude vypočítávat mnohem vícekrát než je třeba. Efektivnější by to bylo skrze cachování výsledků funkcí pokud to zvolený databázový systém podporuje, nebo ještě lépe by se dal využít redundantní atribut pro průměrné hodnocení.

## Funkce 7.3 - Seznam skladeb k danému interpretu

### Vstupy

- #interpret\_id - id interpreta

```
SELECT song.id, song.title
FROM interpret_song is_
      JOIN song ON song.id is_.song_id
WHERE is_.interpret_id = #interpret_id
```

## Funkce 7.4 - Seznam skladeb k danému žánru

### Vstupy

- #genre\_id - id žánru

```
SELECT song.id, song.title
FROM container_genre gs
      JOIN container ON gs.container_id = gs.container_id
      JOIN song ON song.container_id = container.id
WHERE gs.genre_id = #genre_id
```

## Funkce 7.5 - Detail skladby

### Vstupy

- #song\_id - id skladby

```
SELECT song.id, song.title, song.description, song.release_date,
song.duration, song.image_filepath, container.id
FROM song
      JOIN container ON container.id = song.id
WHERE song.id = #song_id
```

## Funkce 7.6 - Aktualizace skladby

### Vstupy

- #user\_id - id uživatele
- #song\_id - id skladby
- #song\_new – hodnoty atributů aukce po aktualizaci (jako strukturovaná proměnná, pokud to zvolený databázový systém podporuje, pokud ne tak budou vstupem proměnné odpovídající postupně všem atributům aukce)

### Výstupy

- #error\_msg - výstupní chybové hlášení (inicializováno na NULL)

**Popis:** Funkce aktualizuje záznam skladby a vytváří nový záznam v historii dané skladby.

Funkce bude řešena jako transakce.

1. Do strukturované proměnné #song\_old načteme aktuální hodnoty všech atributů skladby

```
SELECT * FROM song WHERE song.id = #song_id
```

2. Do proměnné #now se uloží aktuální časové razítko CURRENT\_TIMESTAMP
3. Vytvoření nového záznamu v tabulce song a uložení jeho id do proměnné #song\_id

```
INSERT INTO song_history(song_id, modified_at, new_title,  
new_description, new_duration, new_release_date, new_image_filepath,  
user_id)  
VALUES (#song_id,  
    #now,  
    COALESCE(#song_new.title, #song_old.title),  
    COALESCE(#song_new.description, #song_old.description),  
    COALESCE(#song_new.duration, #song_old.duration),  
    COALESCE(#song_new.release_date, #song_old.release_date),  
    COALESCE(#song_new.image_filepath, #song_old.image_filepath)  
    #user_id);
```

4. Provedení aktualizaci skladby

```
UPDATE song  
SET title          = COALESCE(#song_new.title, #song_old.title),  
    description    = COALESCE(#song_new.description,  
                                #song_old.description),  
    Release_date   = COALESCE(#song_new.release_date,  
                                #song_old.release_date),  
    duration       = COALESCE(#song_new.duration, #song_old.duration),  
    image_filepath = COALESCE(#song_new.image_filepath,  
                                #song_old.image_filepath)  
WHERE id = #song_id
```

**Poznámka:** Pokud něco selže, nastaví se chybové hlášení #error\_msg na „Nepodařilo se aktualizovat skladbu.“ a transakci ukončíme

## Funkce 7.7 - Seznam historie aktualizací

### Vstupy

- #song\_id - id skladby

```
SELECT sh.song_id, sh.modified_at, sh.user_id, user.username
FROM song_history sh
      JOIN user ON user.id = sh.user_id
WHERE sh.song_id = #song_id
```

## Funkce 7.8 - Vygenerování řetězce v normalizovaném tvaru

### Vstupy

- #song\_id - id skladby

### Výstupy

- #result - řetězec ve formátu "<interprets> - <song title>"

**Popis:** Funkce spojuje interprety a název skladby do jednoho řetězce ve formátu "<interprets> - <song title>", kde interpreti jsou odděleni ampersandem. Výsledný řetězec je vrácený jako přes výstup.

1. Načtení názvu skladby do #song\_title

```
SELECT song.title
FROM song
WHERE song.id = #song_id
```

2. Vytvoření kurzoru, který bude procházet interprety dané skladby. Aktuálně procházený záznam bude uložen ve strukturované proměnné #row.

```
SELECT interpret.name
FROM interpret_song is_
      JOIN interpret ON interpret.id = is_.interpret_id
WHERE is_.song_id = #song_id
```

3. V cyklu se bude procházet kurzor a pokaždé se k řetězci #result připojí název interpreta a řetězec ' & '

```
#result := CONCAT (#result, CONCAT(interpret.name, ' & '));
```

4. Odstraní se přebývajícího sufixu ' & ' z řetězce #result

```
#result:= RTRIM(#result, ' & ');
```

5. Přidání oddělovací pomlčky a názvu skladby do řetězce #result

```
#result := CONCAT (#result, CONCAT(' - ', #song_title));
```



## 4. Návrh uživatelského rozhraní

### Seznam formulářů

Formulář 4.1.	<b>Přehled skladeb</b> zobrazeno pro všechny uživatele
Formulář 4.2.	<b>Přehled interpretů</b> zobrazeno pro všechny uživatele
Formulář 4.3.	<b>Přehled playlistů</b> zobrazeno pro všechny uživatele
Formulář 4.4.	<b>Přehled žánrů</b> zobrazeno pro všechny uživatele
Formulář 4.5.	<b>Advanced search</b> zobrazeno pro všechny uživatele
Formulář 4.6.	<b>Sekce komentářů</b> zobrazeno pro všechny uživatele
Formulář 4.7.	<b>Vytvoření skladby</b> zobrazeno přihlášenému uživateli
Formulář 4.8.	<b>Vytvoření interpreta</b> zobrazeno přihlášenému uživateli
Formulář 4.9.	<b>Vytvoření playlistu</b> zobrazeno přihlášenému uživateli
Formulář 4.10.	<b>Vytvoření žánru</b> zobrazeno správci
Formulář 4.11.	<b>Detail skladby</b> zobrazeno pro všechny uživatele
Formulář 4.12.	<b>Detail interpreta</b> zobrazeno pro všechny uživatele
Formulář 4.13.	<b>Detail playlistu</b> zobrazeno pro všechny uživatele
Formulář 4.14.	<b>Skladby playlistu</b> zobrazeno pro všechny uživatele
Formulář 4.15.	<b>Detail žánru</b> zobrazeno pro všechny uživatele
Formulář 4.16.	<b>Editace skladby</b> zobrazeno přihlášenému uživateli
Formulář 4.17.	<b>Editace interpreta</b> zobrazeno přihlášenému uživateli

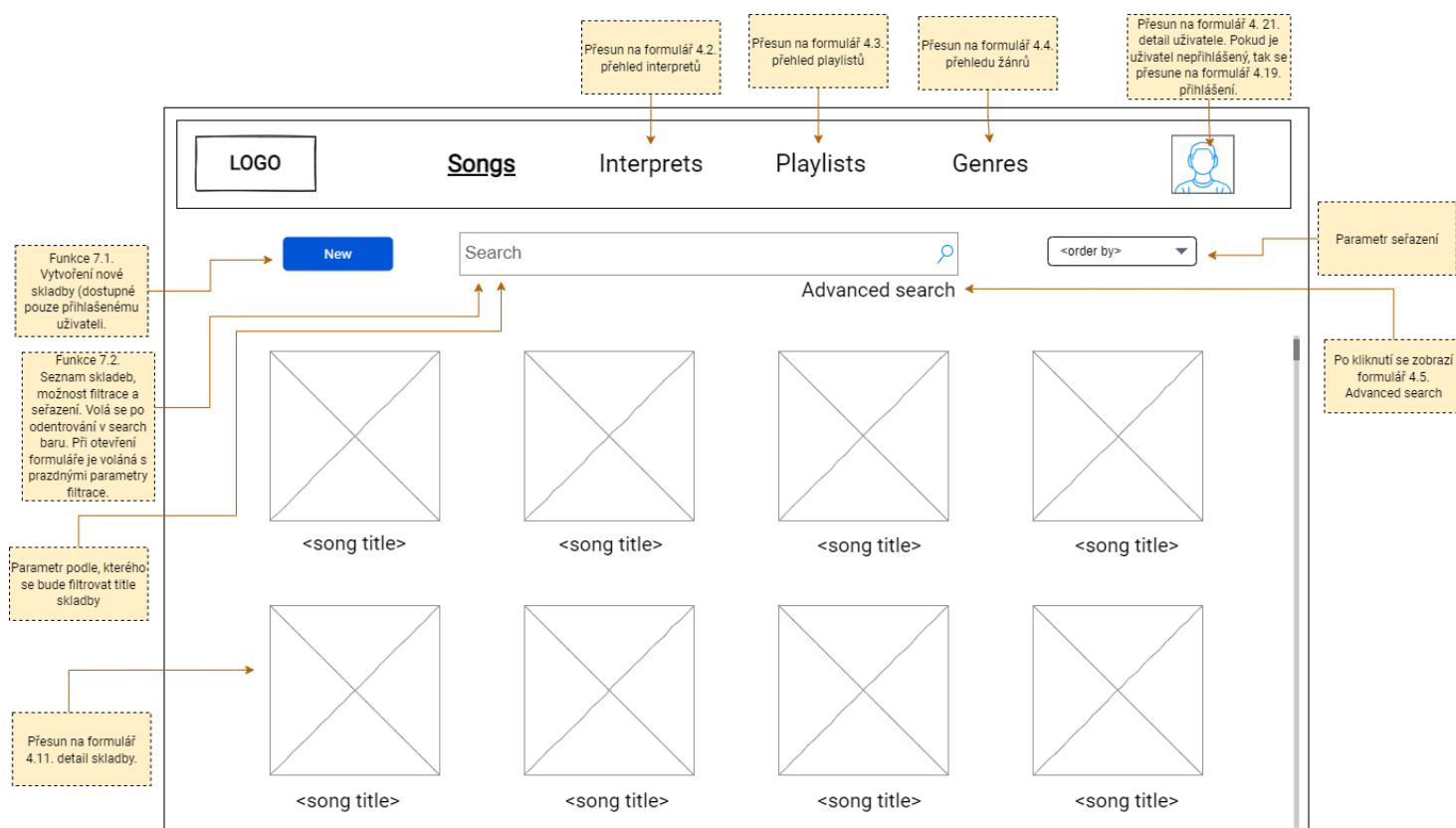


<b>Formulář 4.18.</b>	<b>Editace playlistu</b> zobrazeno pro přihlášeného uživatele (pouze vlastní záznam)
<b>Formulář 4.19.</b>	<b>Přihlášení</b> zobrazeno pro nepřihlášeného uživatele
<b>Formulář 4.20.</b>	<b>Registrace</b> zobrazeno pro nepřihlášeného uživatele
<b>Formulář 4.21.</b>	<b>Detail uživatele</b> zobrazeno pro přihlášeného uživatele (pouze vlastní záznam)
<b>Formulář 4.22.</b>	<b>Přehled vlastních komentářů</b> zobrazeno přihlášenému uživateli (pouze vlastní záznamy)
<b>Formulář 4.23.</b>	<b>Přehled vlastních playlistů</b> zobrazeno přihlášenému uživateli (pouze vlastní záznamy)

## Detailní popis formulářů

**Poznámka:** Jelikož některé formuláře jsou variantou jiných, tak neuvádím detailní popis každého formuláře. Např. formulář skladeb a formulář interpretů mají identický interface, jen namísto seznamu skladeb obsahuje seznam interpretů a proto uvedu jen detailní popis formuláře "přehled skladeb"


### Formulář 4.1 - Přehled skladeb




#### Poznámka:

- Přehled interpretů, playlistů a žánrů vypadá velice obdobně.
- Tlačítko New v případě přehledu žánrů je dostupné pouze správci.

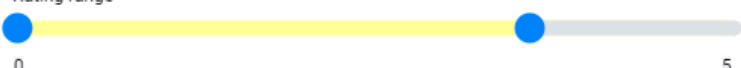
## Formulář 4.5 - Advanced search

Search  

Advanced search

<order by> 

**Filters**


Rating range 


Minimum comments:


Genres:


Interprets:

Min duration:  Max duration:

Created before:  

Created after:  

Released before:  



Released after:  


**Annotations:**

- Funkce 5.2. Seznam skladeb, možnost filtrace a seřazení. Volá se po odeslání v search baru.
- Paremetr seřazení
- Parmetry filtrace.

## Formulář 4.6 - Sekce komentářů

**Comments**

 **<myself>**  
Text...  Datetime

 **Username**  
Text... Datetime

**Annotations:**

- Funkce 2.1. Vytvoření komentáře. (dostupné pouze přihlášenému uživateli)
- Funkce 2.2. Seznam komentářů k danému kontejneru
- Funkce 2.4. Odstranění komentáře (dostupné pouze pro autora/správce/moderátora obsahu)

## Formulář 4.7 - Vytvoření skladby

Formulář 4.7 - Vytvoření skladby. Obsahuje tlačítko "Choose image" vedle obrazovky s křížem. Pod ním jsou pole pro "Title:", "Interpret(s):", "Genres(s):", "Duration:" a "Release date:", každé s příkladem formátu (<title>, <interprets seperated by semicolon>, <genres seperated by semicolon>, <duration>, <releasedate>). Dále je sekce "Description" s textovým polem (<description>). Na konci jsou tlačítka "Cancel" a "Create".

Návrat zpět na formulář 4.1. přehled skladeb

Funkce 7.1. Vytvoření skladby

Funkce 4.7. Přidání žánrů na základě řetězce

Funkce 6.8. Přidání interpretů od dané skladby na základě řetězce

## Formulář 4.9 - Vytvoření playlistu

Formulář 4.9 - Vytvoření playlistu. Obsahuje tlačítko "Choose image" vedle obrazovky s křížem. Pod ním je pole pro "Title:" s příkladem (<title>). Následují radiové tlačítka "Public" (výběr) a "Private". Dále je sekce "Description" s textovým polem (<description>). Na konci jsou tlačítka "Cancel" a "Create".

Návrat zpět na formulář 4.3. přehled playlistů

Funkce 5.1. Vytvoření skladby

## Formulář 4.11 - Detail skladby

**Formulář 4.11 - Detail skladby**

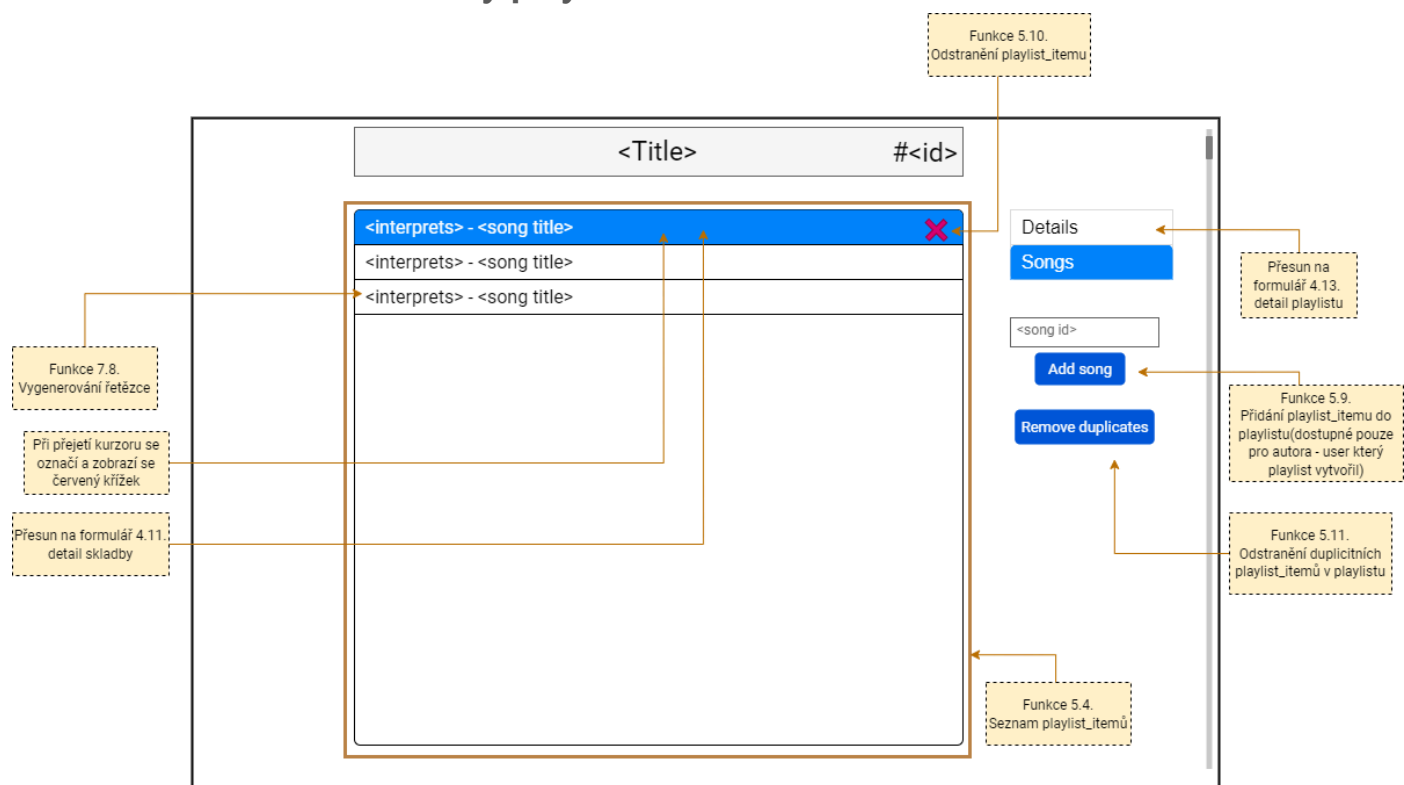
The form displays the details of a song. It includes a title field with a placeholder `<Title>` and an ID field with a placeholder `#<id>`. Below the title is a placeholder for an album cover image. To the right of the image are five stars for rating, with callouts for **Funkce 3.1. Funkce 3.2. Přidání/aktualizace hodnocení** and **Funkce 3.4. Průměrné hodnocení**. Below the stars are fields for **Interpret(s):**, **Genres(s):**, **Duration:**, and **Release date:**. A **Description** field is located below these. A **Comments** section contains a text input for a comment, a **Comment** button, and a list of comments with **Username** and **Datetime** fields. Callouts include **Funkce 4.2. Seznam žánrů k danému kontejneru(skladbě)** and **Formulář 4.6 seznam komentářů**. On the right, a **Current version** dropdown and an **Edit** button are present, with callouts for **Funkce 7.7. Zobrazení historie** and **Přesun na formulář 4.16. editace skladby**. A callout **Dostupné pouze přihlášenému uživateli** points to the top right area.

## Formulář 4.13 - Detail playlistu

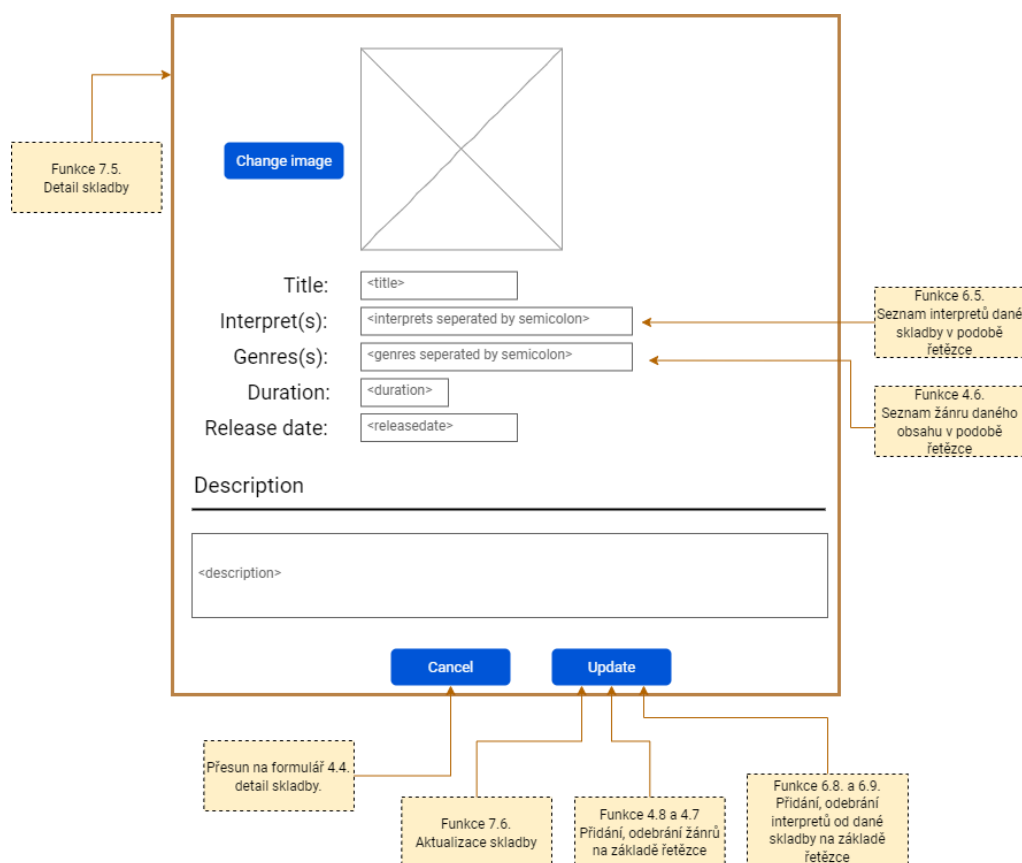
**Formulář 4.13 - Detail playlistu**

The form displays the details of a playlist. It includes a title field with a placeholder `<Title>` and an ID field with a placeholder `#<id>`. Below the title is a placeholder for a playlist cover image. To the right of the image are five stars for rating, with callouts for **Funkce 3.1. Přidání hodnocení** and **Funkce 3.4. Průměrné hodnocení**. Below the stars are fields for **Genres(s):**, **Duration:**, **Created:**, and **User:**. A **Description** field is located below these. A **Comments** section contains a text input for a comment, a **Comment** button, and a list of comments with **Username** and **Datetime** fields. Callouts include **Funkce 4.2. Seznam žánrů k danému kontejneru(skladbě)** and **Formulář 4.6. sekce komentářů**. On the right, an **Edit** button and a **Details** dropdown with a **Songs** option are present. Callouts include **Přesun na formulář editace playlistu**, **Přesun na formulář 4.14. skladby playlistu**, and **Dostupné pouze přihlášenému uživateli**.

## Formulář 4.14 - Skladby playlistu



## Formulář 4.16 - Editace skladby



## Formulář 4.19 - Přihlášení

The diagram shows a 'Sign in' form with two input fields: 'username' and 'password'. Below these fields are two buttons: 'Login' and 'Register'. Annotations include a dashed box pointing to the 'Register' button with the text 'Přesun na formulář 4.20. registrace', and another dashed box pointing to the 'Login' button with the text 'Funkce 1.8, 1.9 Převodění hesla na hash. Porovnání s databází.'

Sign in

username

password

Login

Register

Přesun na formulář 4.20. registrace

Funkce 1.8, 1.9  
Převodění hesla na hash. Porovnání s databází.

## Formulář 4.20 - Registrace

The diagram shows a 'Create account' form with four input fields: 'email', 'username', 'password', and 'repeat password'. Below these fields is a 'Register' button. An annotation in a dashed box points to the 'Register' button with the text 'Funkce 1.1. Vytvoření nového účtu'.

Create account

email

username

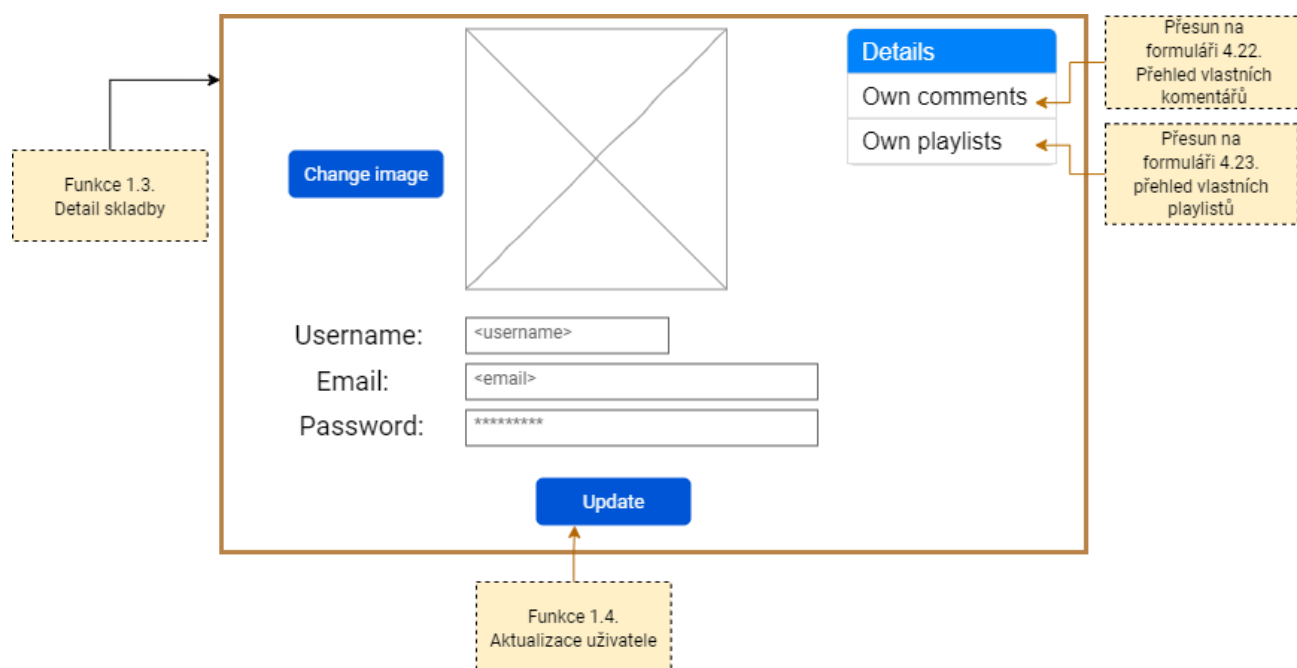
password

repeat password

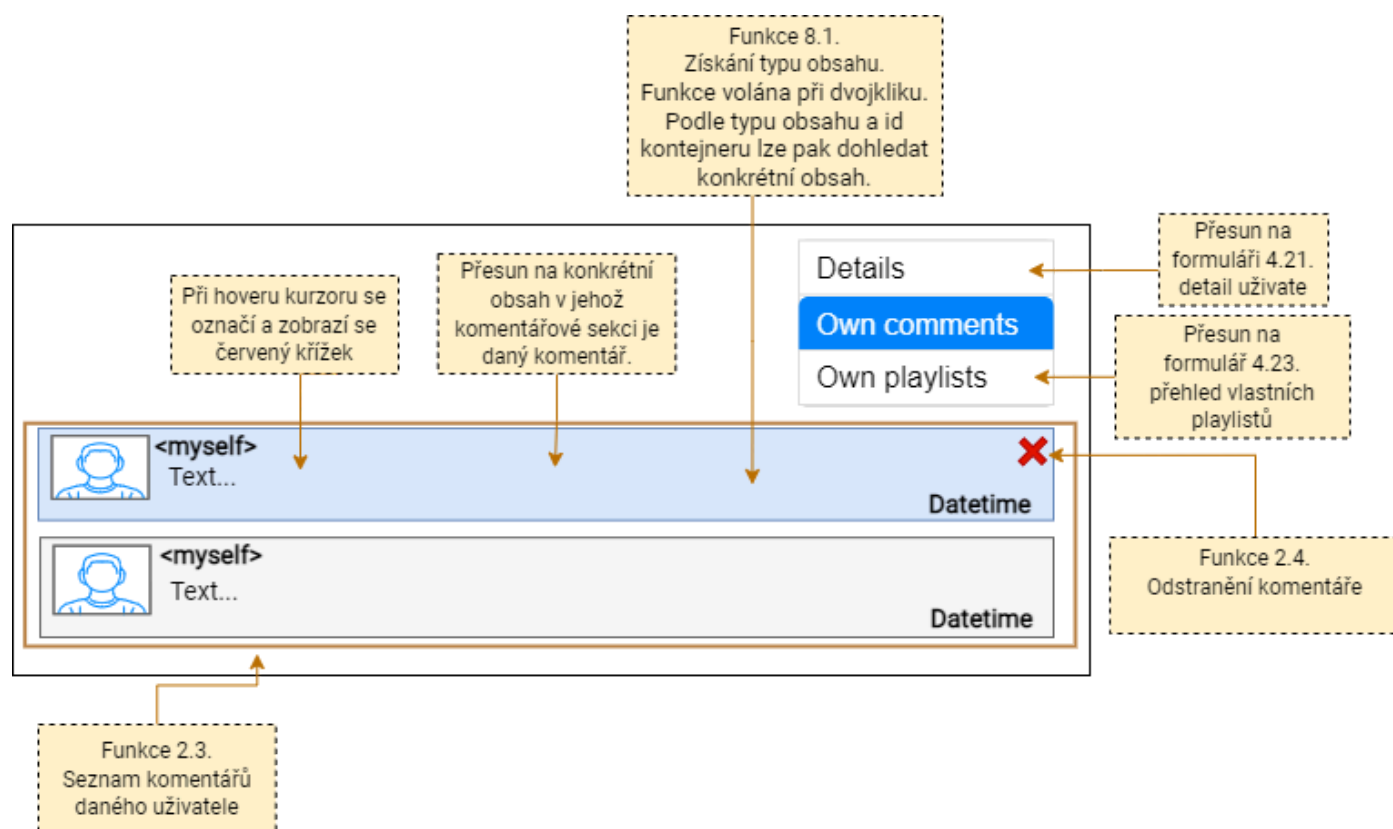
Register

Funkce 1.1.  
Vytvoření nového účtu

## Formulář 4.21 - Detail uživatele



## Formulář 4.22 - Přehled vlastních komentářů





## Formulář 4.23 - Přehled vlastních playlistů

