Univerzita Hradec Králové
Fakulta informatiky a managementu
Fakulta informatiky a kvantitativních metod

Webový zpěvník

Bakalářská práce

Autor: David Jiroudek
Studijní obor: Aplikovaná informatika

Vedoucí práce: Ing. Karel Malý, Ph.D.

Hradec Králové **červen 2022**

Prohlášení:

Prohlašuji, že jsem bakalářskou práci zpracoval samostatně a s použitím uvedené literatury.

V Hradci Králové dne 28. 6. 2022 David Jiroudek

Poděkování

Tímto bych chtěl poděkovat vedoucímu bakalářské práce panu Ing. Karlu Malému, Ph.D. za rady, připomínky a odborné vedení této práce.

Anotace

Bakalářská práce se zabývá vývojem webové aplikace, přesněji webového zpěvníku, pro zobrazování a přidávání textů písní s hudebními prvky, za použití nejnovějších vývojových technologií jako je HTML5, CSS3, React, .net core 6 aj. Projekt je zaměřen na uživatelskou přívětivost, přesnost zanesených dat, škálovatelnost a responzivní design. Tato práce poskytuje stručný popis využitých technologií, vysvětlení použité hudební teorie a charakteristiku jednotlivých stránek z hlediska jejich funkce a uspořádání obsahu. Dále je zde popsána funkcionalita backendové části aplikace a závěrečné poznatky z dosažených výsledků.

Annotation

Title: The Web Song Book

The bachelor thesis deals with the development of a web application, more precisely a web songbook, for displaying and adding lyrics with musical elements using the latest development technologies such as HTML5, CSS3, React, .net C #, etc. The project focuses on user-friendliness, accuracy of data entry, scalability and responsive design. This thesis provides a brief description of used technologies, an explanation of the technologies used, an explanation of the music theory used, and a characterization of each site in terms of its function and content layout. Furthermore, the functionality of the backend part of the application and the final findings from the results obtained are presented.

Obsah

1 Úvod

Jedním z nejčastějších problémů, týkajících se webových zpěvníků, je nemožnost přidání hudebních značek přesně na místa, kde mají být hrány. Toto představuje problém hlavně pro začínající umělce, kteří kvůli nedostatku zkušeností nejsou schopni samostatně rozeznat, kdy se má daná značka hrát, což má za příčinu nesprávné zahrání písně. Spjatým problémem je dodržování předepsaného kodexu při přidávání skladby, který je často příliš složitý a zdlouhavý. Nejčastěji se značky vkládá uživatel přímo do textu v podobě předepsaného znaku v závorkách, ty poté stránka zpracuje do potřebné podoby. Výsledkem je právě již zmíněné špatné přiřazení značek k dané části textu a zdlouhavost zápisu a úprav písní.

Další překážkou je neznalost hudební nauky, nejčastěji chyby při enharmonické záměně v tónině skladby. Většině těchto chyb lze předejít přívětivým uživatelským prostředím a kontrolou vstupu. S touto myšlenkou se autor rozhodl vytvořit tento webový zpěvník, který si klade za cíl vyřešit tyto problémy a poskytnout uživatelům příjemné prostředí pro vytváření, editaci i přehrávání skladeb.

Historie projektu

Původní návrh zpěvníku z roku 2014 vznikl díky iniciativě mého bratra, se kterým jsme se v té době každý rok účastnili kytarového soustředění. Zde jsme ale u večerního hraní písní neměli dostatečný počet kopií textů s akordy, abychom se mohli zapojit všichni. Zpěvník se měl stát souhrnným médiem pro všechny písně, které se na soustředění hrají, v jednotném formátu, psaný v originálních verzích a dostupný pro všechny studenty. Tím se měl stát i pomocnou učební pomůckou pro studenty.

2 Cíl práce

Hlavním cílem projektu je vytvořit uživatelsky přívětivou webovou aplikaci pro přidávání, editaci a zobrazování písní, za použití nejnovějších vývojových prostředků jako je React, C# .net 5.0, Elasticsearch aj.

Tento hlavní cíl jde dále rozčlenit na několik dalších podcílů, kterými jsou: vytvoření značkovacího jazyka, jímž budou popsány jednotlivé prvky skladby, naprogramování editoru písní, kterým bude moci uživatel právě zmíněné prvky zanést do textu a implementovat zobrazovací algoritmus. Ten bude dynamicky umísťovat vizuální reprezentace značek na své pozice.

Dále je potřeba navrhnout a utvořit vhodné seskupení databází a jejich strukturu. Posledním bodem je část aplikace napsaná v C#, jež bude zajišťovat komunikaci mezi všemi částmi a zaznamenávat chybová hlášení.

Teoretická část

3 Použitá hudební teorie

Webový zpěvník, tedy soubor textů a jejich hudebních doprovodů, zveřejněných na webové stránce, se obsahem a strukturou neliší od tištěných variant. V obou nalezneme titulní stranu, informace o zpěvníku, obsah a jednotlivé písně. Webový zpěvník má tyto informace pouze rozmístěné na internetu, což je pohodlnější pro uživatele, protože je v dnešní době dostupný kdykoli a kdekoli. Nemusí se tak řešit nedostatečný počet kopií a jejich konzistentnost, pokud si chce skupina lidí společně zazpívat.

Další nespornou výhodou je možnost transpozice, kterou není nutné vymýšlet z hlavy, jelikož většina webových zpěvníků tuto možnost poskytuje jako funkci.

Stejně jako normální tištěný zpěvník i webový zpěvník podléhá autorským právům. Pro zanesení písně do zpěvníku je tedy, z právního hlediska, nutný souhlas autora se zveřejněním nebo zakoupení licence. Drtivá většina online zpěvníků ani jedno nemá. Většina autorů tedy pro tento druh zápisu jejich díla koupení práv nevyžaduje.

Na konec je tedy nutné uvést základní terminologii a teorii, které se běžně při tvorbě, ať už tištěných nebo webových zpěvníků, používá.

Tónina

Tóninou se v hudební terminologii rozumí příslušnost tónů skladby k určité stupnici. Této znalosti je potřeba jak pro kontrolu správnosti vložených akordů z hlediska enharmonické záměny transpozice, tak pro zapisování tabulatur.

Akord

Akord je souzvuk nejméně tří tónů. (1) Pro doprovod je nejdůležitější součástí písně, jelikož určuje, jaké tóny se při hraní mají hrát.

Repetice

Repetice jsou dvojité taktové čáry (často opatřeny šikmými křidélky nad notovou osnovou, díky nimž vypadají jako závorky). Znamená jednoduše opakovat hudební úsek, který je jimi vyznačen. Obvykle se předpokládá jedno opakování, pokud jich má být více, je nutno to výslovně označit. (1)

Prima a seconda volta

„Prima volta“ a „seconda volta“ (hrát poprvé a podruhé). Když hrajeme úsek označen repeticí poprvé, zahrajeme takty označené prima voltou. Při hraní téhož úseku podruhé, přeskočíme takty označené prima voltou a hrajeme ty označené seconda voltou. (1)

Označení částí písní

Jednotlivé části písně lze rozčlenit do několika typů:

- **Sloky** – většinou mají stejnou melodii, ale odlišný text. Značí se arabskými číslicemi podle pořadí jednotlivých slok.
- **Refrén** – většinou mají stejnou melodii i text. V písni se tato část vyskytuje více než jednou.
- **Mezihra** – převážně pouze instrumentální, slouží k oddělení jednotlivých částí.
- **Recitace** – část písně, která, jak název napovídá, má být odrecitována. Většinou bez melodického doprovodu.

Enharmonická záměna

Enharmonická záměna je možnost zaměnit dva (nebo tři) různě zapsané tóny, které se však stejně hrají. Enharmonicky zaměňovat lze jak jednotlivé tóny, tak i stupnice, intervaly i akordy. (2)

Transpozice

Termín transpozice se v hudební nauce využívá ve významu převedení skladby či její části z jedné tóniny do jiné při zachování vztahů (intervalů) mezi jed-

notlivými tóny skladby. Pro jakýkoli zpěvník je nutností proto, aby vyhovoval hlasovým rozsahům zpěváků nebo změnám originálních akordů na akordy snadněji hratelné.

Praktické použití

Při transpozici v zobrazení skladby nebo vytváření skladby je třeba kontrolovat, zdali jsou dané tóny/akordy součástí tóniny, ve které je píseň zapsána. Tímto bude zajištěna hudební konzistentnost, popřípadě upozorní na případné špatné zvolení akordu při vytváření písně. Např. v tónině C dur by se v neměl (v převážné většině případů) vyskytovat akord C# dur.

Tabulatury

Tabulatury představují alternativní možnost zápisu tónů. Na rozdíl od standardní notace, kde je zapisována výška tónu, je zde tón zapsán jako prstoklad určitého nástroje. Je tudíž vhodný pro všechny umělce, nehledě na to, zdali umí číst noty ve standardní notové osnově, či nikoli. (3)

Dynamika

Dynamika označuje intenzitu přednesu, tedy jak silně a kdy bude umělec danou část skladby hrát, případně jak má v průběhu času dynamiku měnit. Většinou je popsána italským názvoslovím, od něhož se odvíjí i zkratky, jimiž se zapisují. Pro účely tohoto projektu poslouží jen základní terminologie, tedy:

ppp – piano pianissimo – co nejslaběji

pp – pianissimo – velmi slabě

p – piano – slabě

mp – mezzopiano – středně slabě

mf – mezzoforte – středně silně

f – forte - silně

ff – fortissimo – velmi silně

fff – forte fortissimo – co nejsilněji

cresc. (<) - crescendo - zesilovat

decresc. (>) - decrescendo – zeslabovat

(1)

4 Použité technologie

4.1 HTML

HTML (HyperText Markup Language) je nejzákladnějším stavebním kamenem webu. Definuje význam a strukturu webového obsahu. (4) Využívá párových a nepárových tagů, jež obalováním obsahu tvoří samotný web. Dalším aspektem je možnost vkládání hypertextový odkazů, které slouží k přesunu na jakékoli stránky na webu. První verze byla vydána v roce 1993 a v průběhu let se dočkala mnoha verzí, nejnovější je 5.2 z roku 2017. (5)

4.2 CSS

CSS (Cascading Style Sheets) je technologií pro definování stylů jednotlivých elementů, které určují, jak se budou na stránce zobrazovat. Existuje několik způsobů, jak tyto styly zapsat:

- do párových tagů s názvem „Style“ v jednotlivých HTML souborech
- přímo do určitého tagu elementu
- do souboru s příponou.css, který je poté nutné importovat do HTML stránky. (6)

Významnou roli hraje CSS při dynamickém zobrazení stránky pro jednotlivá zařízení s rozdílnými rozměry obrazovky, pro které lze stanovit rozložení a vzhled stránky podle potřeb.

4.3 JavaScript

JavaScript je po HTML a CSS třetím základním stavebním prvkem pro tvorbu webu. Tento objektově orientovaný, dynamicky typovaný skriptovací jazyk přidává funkcionalitu prvkům webu. Příkladem mohou být funkce tlačítek, validace formulářů, získávání dat ze serveru nebo dynamické přidávání obsahu do stránky. (7) Nad tímto jazykem jsou postaveny nadstavbové knihovny, jako je např., v další části práce vysvětlený, React.

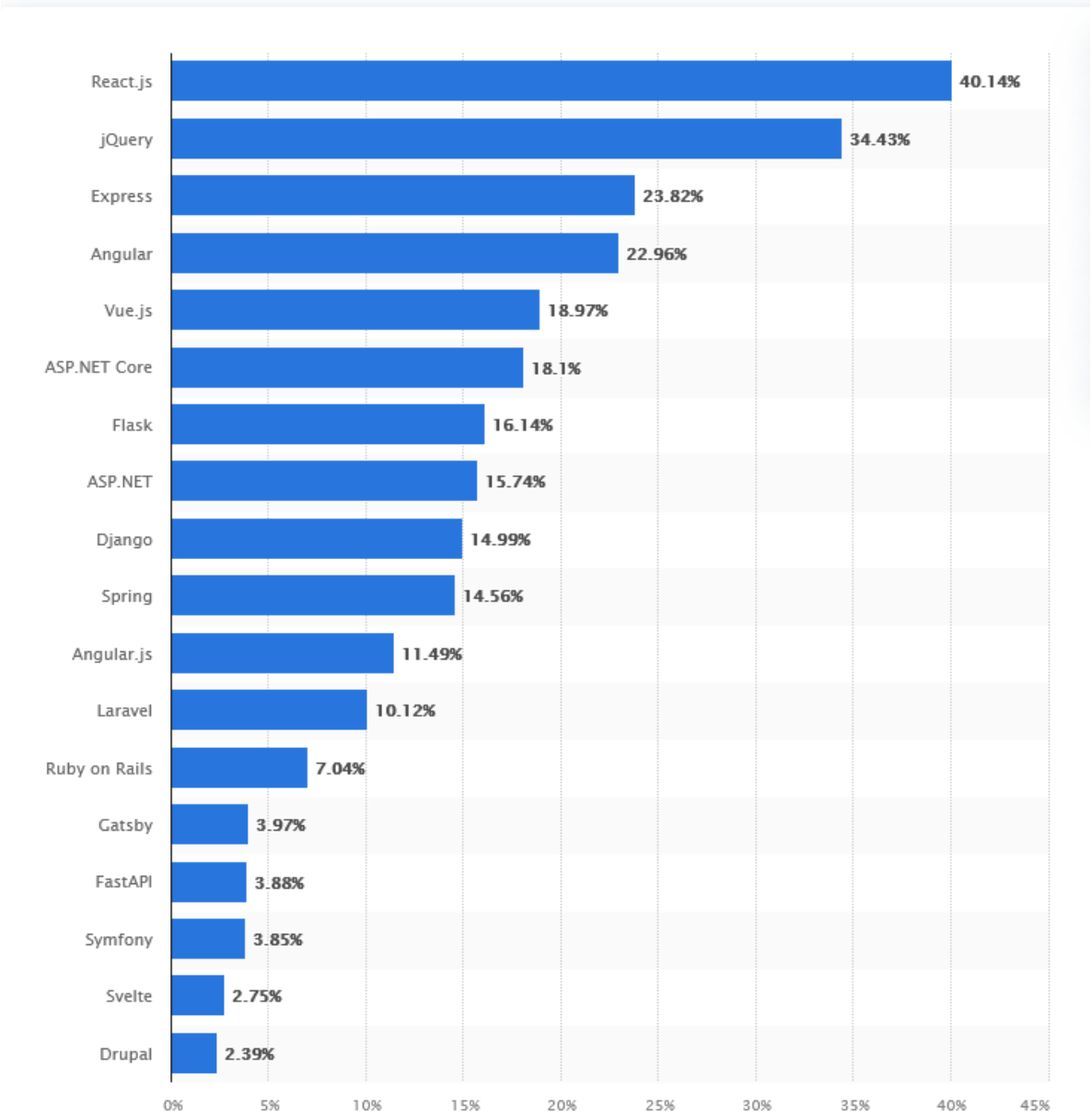
4.4 React

React je deklarativní, efektivní a flexibilní JavaScriptová knihovna pro vytváření uživatelských rozhraní (8). První představení veřejnosti provedla firma Facebook (dnes Meta Platforms) v roce 2013 za účelem vytvoření jazyka pro znovu im-

plementovatelné části rozhraní. React nám umožňuje psát komponenty pomocí doménově specifického jazyka JSX. React jej interně převede na virtuální DOM a nakonec vygeneruje HTML (9). Na této technologii je postaven celý frontend aplikace.

Důvod zvolení: Pro tuto technologii se autor rozhodl z důvodu záměrů vytvoření intuitivního uživatelského rozhraní k vytváření písně, při kterém se často mění obsah stránky. React je také v dnešní době jedna z nejčastěji používaných knihoven pro vývoj jednostránkových webových aplikací.

Obrázek 1– nejpoužívanější webové frameworky roku 2021 (10)



4.4.1 JSX

JSX (JavaScript XML) je rozšířením syntaxe JavaScriptu, která umožnuje přidávat prvky HTML do kódu JavaScriptu tím, že je převeden do elementů reactu. Toto v praxi umožňuje přidávání prvků do DOM bez nutnosti použití funkcí createElement() a appendChild(), což činí vývoj rychlejším a přehlednějším. (11)

4.4.2 Virtual DOM

DOM (Document Object Model) je aplikační programové rozhraní (API) pro HTML a XML dokumenty. Definuje logickou strukturu dokumentů a způsob, jakým se s dokumentem přistupuje a jak se s ním manipuluje. (12)

Pojmem virtual DOM se tedy rozumí další vrstva odkazující na originální DOM. Jakákoli změna v této vrstvě spustí mechanismus, který porovná rozdíly před a po změně a poté v pravém DOM aktualizuje pouze ty prvky, které byly změněny. Tímto docílíme rychlejšího běhu aplikace díky absenci nutnosti aktualizovat celou stránku. (13)

4.4.3 Hooks proti class components

Poté, co React ve verzi 16.8 představil novou metodu hooks, mohou se vývojáři rozhodnout mezi dvěma styly psaní, pomocí class component nebo zmíněných hooks (funkcionálně). Další možností je tyto styly kombinovat. Autoři rovněž zmiňují, že právě hooks by se v nových projektech měly stát hlavním způsobem psaní aplikací, jelikož další vývoj bude zaměřen právě na ně. Z těchto důvodů budou v práci použity. (14)

4.4.4 Redux a Redux Toolkit

Redux je JavaScriptová, open-source knihovna pro spravování stavů aplikace. Tyto stavy tedy mohou být uloženy v „Redux store“, odkud si je poté jakákoli část aplikace může brát nebo naopak může odesílat akce k jejich upravení. Tím pomáhá udržet přehlednost aplikace a při jejím testování. Dan Abramov ji představil v roce 2015, za účelem vytvořit knihovnu pro správu stavu s minimálním API, ale zcela předvídatelným chováním (15)

Redux toolkit

Redux Toolkit je oficiálně doporučený přístup pro psaní logiky Redux. Tento nástroj obaluje Redux jádro, obsahuje balíčky a funkce, které jsou nezbytné pro vytvoření aplikace. Redux Toolkit zjednodušuje většinu úloh Redux, předchází běžným chybám a usnadňuje psaní aplikace. (16)

4.5 C# a ASP.NET Core MVC

C# (vyslovováno „C sharp /šárp/“) je moderní, objektově orientovaný a typově bezpečný programovací jazyk (17). Své základy má v jazyce C a svým stylem psaní je velmi podobný jazykům C++ a Java. C# je standardizovaný asociací ECMA (European Computer Manufacturers Association) International jako ECMA-334. Je vyvíjen firmou Microsoft od roku 2000 jako jejich nejpoužívanější jazyk pro objektové programování. (18)

Důvod zvolení: Z pohledu autora je C# jazykem, se kterým dlouhodobě pracuje a je, dle jeho názoru, ideálním nástrojem pro zpracovávání dat a vytvoření webového backendu.

ASP.NET Core MVC 5

ASP.NET Core MVC je rozsáhlý framework pro vytváření webových aplikací a rozhraní API pomocí návrhového vzoru Model-View-Controller (Model-Pohled-Řadič). Je tedy, jak název napovídá, nadstavbou ASP.NET core multiplatformní open-sourcová architektura pro vytváření moderních aplikací připojených k internetu s podporou cloudu.

MVC vzor

Architektonický vzor MVC rozděluje aplikaci do tří hlavních skupin komponent: Modely, Pohledy a Řadiče. Tento vzor pomáhá dosáhnout oddělení možných problémů. Pomocí tohoto vzoru jsou požadavky uživatelů směrovány do řadiče, který je zodpovědný za práci s modelem za účelem provádění uživatelských akcí a/nebo získávání výsledků dotazů. Řídicí jednotka vybere zobrazení, jež se má zobrazit uživateli a poskytne mu veškerá data modelu, která potřebuje. (19)

4.6 Rest API

REST (Representational State Transfer) API je rozhraní, které dva počítačové systémy používají k bezpečné výměně informací přes internet. REST je softwarová architektura, která klade podmínky na to, jak by mělo API fungovat. Byl původně vytvořen jako vodítko pro řízení komunikace ve složité síti, jako je internet. (20)

4.7 MSSQL

MSSQL (Microsoft SQL Server) je systém pro správu relačních databází vyvinutý společností Microsoft. Jako databázový server se jedná o softwarový produkt s primární funkcí ukládání a získávání dat podle požadavků jiných softwarových aplikací – které mohou běžet buď na stejném počítači, nebo na jiném počítači v síti (včetně internetu). (21)

Důvod zvolení: S tímto typem databáze má autor velké množství zkušeností. V projektu najde funkci zálohy dat a ukládání citlivých dat uživatelů.

4.8 Elasticsearch

Elasticsearch je distribuovaný, bezplatný vyhledávací a analytický nástroj pro všechny typy dat, včetně textových, numerických, geoprostorových, strukturovaných a nestrukturovaných. Elasticsearch je postaven na Apache Lucene a byl poprvé vydán v roce 2010 společností Elasticsearch NV (nyní známý jako Elastic). Známý pro své jednoduché REST API, distribuovanou povahu, rychlost a škálovatelnost. (22)

Důvod zvolení: Jelikož skoro všechna vyhledávání a filtry jsou v textové podobě, Elasticsearch je ideálním nástrojem jak pro ukládání dat, tak pro rychle vyhledávání v nich.

4.9 OAuth 2.0

OAuth 2.0 je standardní protokol pro autorizaci. Zaměřuje se na jednoduchost klienta pro vývojáře a zároveň poskytuje specifické autorizační toky pro webové aplikace, desktopové aplikace, mobilní telefony a další zařízení jako je třeba chytrá televize. Nespornou výhodou pro vývojáře je taktéž to, že nemusí uchovávat citlivá data uživatelů, jakou jsou přihlašovací údaje. Tato specifikace a její rozšíření jsou vyvíjeny v rámci pracovní skupiny IETF OAuth. (23)

4.10 Serilog

Serilog je diagnostická knihovna pro protokolování v aplikacích .NET. Snadno se nastavuje, má přehledné API a podporu na všech současných platformách .NET. Je užitečná, jak v jednoduchých, tak v komplexních aplikacích, které využívají strukturované protokolování při vyvíjení složitých, distribuovaných a asynchronních aplikací a systémů. (24)

4.11 Redis

Redis je open source úložiště datových struktur v paměti, které se používá jako databáze, cache, zprostředkovatel zpráv a streamovací engine. Aby bylo dosaženo nejvyššího výkonu, pracuje Redis s datovou sadou v paměti. Redis podporuje asynchronní replikaci s rychlou neblokující synchronizací. (25)

4.12 Swagger

Swagger umožňuje popsat strukturu rozhraní API tak, aby je stroje mohly číst. Díky čtení struktury API můžeme automaticky vytvářet přehlednou a interaktivní dokumentaci API. Může také automaticky generovat klientské knihovny v mnoha jazycích nebo automaticky testovat. Základním kamenem je požádání API o vrácení souboru YAML nebo JSON, který obsahuje podrobný popis celého API. Tuto část aplikace lze prohlédnout na relativní adrese „/swagger“ a v pozdějších verzích aplikace bude znepřístupněna nepřihlášeným uživatelům.

4.13 Docker a CaaS

Docker poskytuje možnost zabalit a spustit aplikaci ve volně izolovaném prostředí zvaném kontejner. Izolace a zabezpečení vám umožňuje provozovat mnoho kontejnerů současně na daném hostiteli. Kontejnery jsou lehké a obsahují vše potřebné ke spuštění aplikace, takže se nemusíte spoléhat na to, co je aktuálně nainstalováno na hostiteli. Při práci můžete snadno sdílet kontejnery a mít jistotu, že každý, s kým sdílíte, dostane stejný kontejner, který funguje stejným způsobem. (26)

CaaS (Containers as a Service) je cloudová služba, která umožňuje vývojářům softwaru a IT oddělením nahrávat, organizovat, spouštět, škálovat, spravovat a zastavovat kontejnery pomocí virtualizace založené na kontejnerech. Poskytovatel

CaaS běžně poskytuje rámec, který uživatelům umožňuje využívat službu. Poskytovatelé obvykle využívají volání aplikačního programovacího rozhraní (API) nebo rozhraní webového portálu. (27)

Analýza a návrh

5 Požadavky na aplikaci

5.1 Funkční požadavky

Aplikace poskytuje uživateli možnost:

- registrace
- přihlášení za účelem získání práv k určitým činnostem
- prohlížení a filtraci přidaných písní
- prohlížení a filtraci přidaných autorů a interpretů
- zobrazení písně
- transpozice písně
- vytvoření písně
- editace písně
- smazání písně

5.2 Nefunkční požadavky

- kompatibilita mezi prohlížeči

  Aplikace musí být funkční, pokud možno v co nejvíce druzích webových prohlížečů. Pro tento projekt jsou zvoleny prohlížeče: Google Chrome a Firefox.

- intuitivní zacházení

  Používání kterékoli části nebo funkcionality aplikace musí být pro uživatele co nejrychleji uchopitelné, bez nutnosti přečtení návodů.

- přehlednost

  Uživatel má po navštívení stránky okamžité povědomí, kde najít informace, které hledá.

- responzivní design

  Komponenty frontedu aplikace mění svoje rozpoložení a velikost na základě šířky, výšky a druhu zařízení, na kterém jsou právě zobrazovány. Toto se nevztahuje na editor, kde dotykové prostředí znemožňuje příjemnému přidávání značek.

- zálohování dat

  Administrátoři mají možnost vrátit se k jakékoli předešlé verzi písníčky. Toto předchází případnému úmyslnému i neúmyslnému poškození dat písníčky.

- caching

  Pro tento projekt je tato část pouze pojistkou při případném rozšíření aplikace mezi veřejnost.

- logging

  Zaznamenává chyby v aplikaci, které si poté vývojáři mohou v budoucnu přečíst a analyzovat, potažmo opravit.

- CaaS (Container as a Service)

6 Značkovací jazyk

Pro účely označení prvků v textu jsem se rozhodl vytvořit vlastní značkovací jazyk, který definuje všechny hudební prvky. Formát je struktury XML, tvoří ho tedy párové a nepárové značky ve špičatých závorkách.

6.1 Seznam prvků

6.1.1 Akordy

Jak vyplývá z definice akordu, druhů akordů je nespočetné množství, proto je nelze uvést všechny. Obsah značky mezi počáteční a koncovou značkou obsahuje pouze jediné písmeno nebo znak a žádné ostatní vnořené značky. Parametry udávají, od jakého tónu je akord odvozen, o jaký typ akordu se jedná (durový, mollový, …), jestli je akord v základní podobě. Většina akordů má pro kytaru pouze jeden základní prstoklad. V písních se ale může vyskytnout akord, který má být hrán jiným

prstokladem. Je proto vhodné tyto variace prstokladů odlišit a zapsat jejich podobu do parametru:

Značky: <ch> </ch>

Zobrazení: Emi

Parametry:

- value – výchozí tón
- sufix – typ akordu
- form – zápis alternativního prstokladu

6.1.2 Prvky související s opakováním části skladby

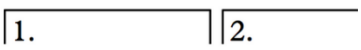
Repetice

Značky: <rep></rep>

Zobrazení: 

Prima a seconda volta

Značky: <volta></volta>

Zobrazení: 

Parametry:

- value – pořadí volty

6.1.3 Označení slok, refrénů a meziher

Sloky, refrény, mezihry a recitace

Značky: <verse></verse>

Zobrazení: 1) R) I) Rec)

Csus4: 200011

Parametry:

- verseType – typ úseku (sloka, refrén, mezihra, recitace)
- verseValue – zobrazované číslo úseku

- verseRefId – referenční odkaz pro zobrazení již přidané části skladby

6.1.4 Prvky související s dynamikou skladby

Dynamika

Značka: <dyn />

Zobrazení: *fff* *ff* *f* *mf* *mp* *p* *pp* *ppp*

Parametry:

- value – typ dynamiky (fff, ff, f, mf, mp, p, pp, ppp)

Crescendo a Decrescendo

Značky: <dynwide></ dynwide >

Zobrazení: 

Parametry:

- value – typ změny dynamiky (<, >)

6.1.5 Prvky související s rytmem skladby

Ritardando a Accelerando

Značky: <rythm></rythm>

Zobrazení: rit. accel.

Parametry:

- value – typ změny rytmiky (rit., accel.)

6.1.6 Další prvky

Změna polohy kapodasteru

Značky: <capo></capo>

Zobrazení: Capo: 1

Parametry:

- value – poloha kapodasteru

Přidání mezery do textu

Značky: <space></space>

Zobrazení: E – tiam

Parametry:

- value – šířka mezery
- dash – zobrazení pomlčky v mezeře (true/false)

Řádky písně

Značky: <row></row>

Obrázek 2 – ukázka zobrazení značek

Hop Trop **aUkazkováPíseňTřiKříže**

15. 8. 2022 Capo: 0

Csus4: 022200

Dmi Capo: 0

1) Dávám sbohem břehům proklatejm,

který v drápech má ďábel sám,

bílou přídí šalupa My Grave

‖:míříím k útesům, který znám. :‖

Csus4 *f*

R) Jen tři kříže z bílýho kamení

fff rit.

někdo do písku poskládal.

accel.

Slzy v očích měl a v ruce znavený,

lodní deník, co sám do něj psal.

I) První kříž má pod sebou jen hřích,

samý pití a rvačky jen.

Chřestot nožů, při kterým přejde smích,

srdce-kámen a jméno Sten.

Rec) Jen tři kříže z bílýho kamení

někdo do písku poskládal.

Slzy v očích měl a v ruce znavený,

lodní deník, co sám do něj psal.

Praktická část

7 Frontend

Celý frontend aplikace je napsán s pomocí HTML, CSS, JavaScriptu a JavaScriptové knihovny React. O stavy se stará knihovna Redux toolkit, která je poskytuje všem komponentům a zpřehledňuje tím kód. Veškerou komunikaci, která je nutná pro získání nebo zápis dat na server a do databází, zajišťují HTTP dotazy na REST API, hostované na backend serveru.

7.1 Části frontendu:

7.1.1 Menu

Je jednotné pro všechny stránky a nachází se, jako hlavička, v horní části stránek. Slouží jako rozcestník mezi ostatními částmi webové aplikace. Obsah se mění v závislosti na tom, zdali je uživatel přihlášený a jaká má práva.

Obsah pro nepřihlášeného uživatele: logo (odkaz na hlavní stranu), pole pro vyhledávání, seznam písní, seznam autorů, přihlásit se.

Obsah pro přihlášeného uživatele: logo, pole pro vyhledávání, seznam písní, seznam autorů, editor, tlačítko uživatele (uživatelovy skladby, odhlásit). Pro adminy se v tlačítku uživatele zobrazí i kolonka uživatelé.

Pozn.: V budoucích verzích je v plánu přidání možnosti výtisku knihy dle parametrů.

7.1.2 Hlavní strana

Hlavní strana je velmi stručná, jelikož se uživatelé budou převážně dostávat na jednotlivé písně přímo prostřednictvím vyhledávače. Najdeme zde, seznam sedmi nejvíce populárních písní a seznam sedmi posledních schválených písní.

Zpracování dat

- **Seznam populárních písní**
    - API dotaz - „/api/search/GetFilteredSearchResult“
    - požadavek - [1]{"publicable": [0, 1], "page": 1, "resultCount": 7, "validated": true, "sortBy": [{"field": "viewCount", "direction": 0}]}
- **Seznam posledních sválených písní**
    - API dotaz - „/api/search/GetFilteredSearchResult“
    - požadavek - {"publicable": [0, 1], "page": 1, "resultCount": 7, "validated": true, "sortBy": [{"field": "createdDate", "direction": 0}]}

Data jsou uložena do stavů, které po vložení dat vykreslí oba seznamy do vyhrazených polí.

7.1.3 Seznam písní

Hlavním účelem pro tuto stranu je přehled přidaných písní a možnost zobrazit si libovolnou skladbu. Na této stránce můžeme najít jednotlivé filtry pro vyhledávání v seznamech písní. Seznam je ve výchozím nastavení rozdělen do dvou až čtyř částí dle pravomocí uživatele plus jejich alternativy s neschválenými písněmi.

- „Publikovatelné“ – písně vhodné pro kohokoli bez urážlivého obsahu.
- „Nepublikovatelné“ – písně obsahující hanlivá slova, tudíž nevhodné např. pro děti.
- „Neviditelné“ soukromé písně viditelné pouze pro pro adminy a moderátory.
- „Zakázané“ – soukromé písně viditelné pouze pro adminy.

Zpracování dat

- API dotaz - „/api/search/GetFilteredSearchResult“
- Prvotní požadavek - {"publicable": [0], "page": 1, "resultCount": 10, "validated": true, "sortBy": [{"field": "songName", "direction": 0}]}

Z odpovědi se uložením do stavu vytvoří seznam písní. Změnami filtrů se mění požadavek a tím i jeho následná odpověď.

[1] Pro vysvětlení položek v požadavcích viz kapitola 8.2 REST API

7.1.4 Zobrazení písně

Zobrazení písně je uživatelsky nejdůležitější částí celé aplikace. Uprostřed stránky najdeme samotný text se značkami. Grafické reprezentace značek jsou dynamicky zobrazovány nad správnými písmeny kusy textu. Nad polem s textem najdeme tlačítka pro transpozici.

Uživatelům s dostatečnými právy budou zobrazena i tlačítka pro: editaci, smazání a schválení písně.

Zpracování dat

- API dotaz - „/api/song/GetSong“
- požadavek – {"SongId": 0}

Data jsou vložena do jim určených stavů a funkce „rerender()“ se postará o správné zobrazení prvků písně.

Smazání písně

- API dotaz - „/api/song/DeleteSong“
- požadavek – {"SongId": 0}

Schválení písně

7.1.5 Seznam autorů

Seznam autorů je principem velmi podobný seznamu písní, s tím rozdílem, že zde nalezneme autory a interprety přidaných skladeb. Filtrem je pouze typ seřazení autorů. Kliknutím na jméno autora se uživatel dostane k seznamů písní, které se daného autora týkají.

Zpracování dat

- API dotaz - „/api/search/GetFilteredAuthorSearchResult“
- požadavek – {{sortBy: [{"field": "songName", "direction": 0}], “page“: 1, “resultCount“: 10}

Z odpovědi se uložením do stavu vytvoří seznam autorů. Změnami filtrů se mění požadavek a tím i jeho následná odpověď.

7.1.6 Přihlášení a registrace

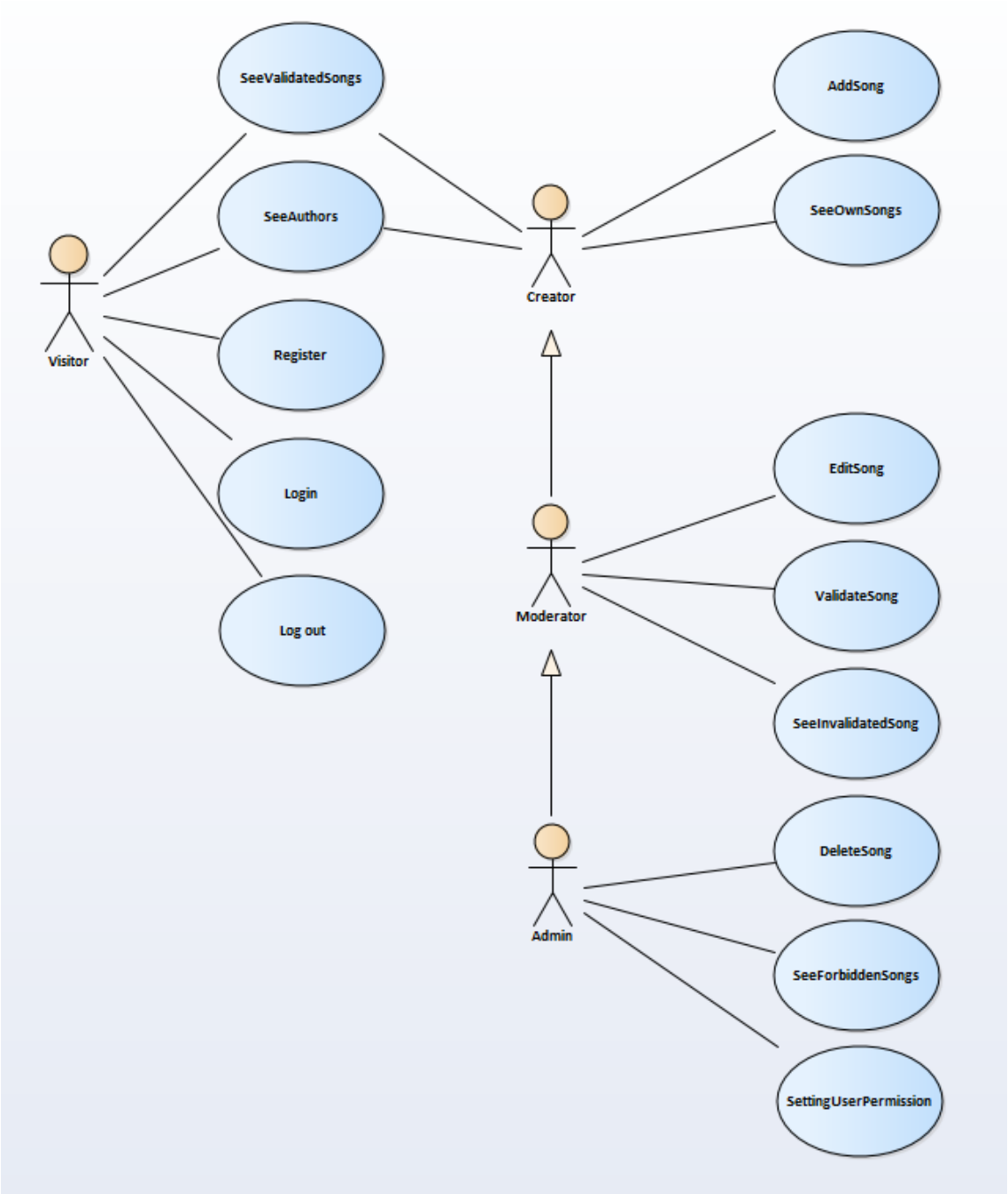
Nejedná se o samostatnou stránku, ale o pop-up (prvek, který se zobrazí nad ostatními elementy a zamezuje interakce s nimi), zobrazitelný na všech stránkách aplikace. (28)

Díky metodě OAuth2.0, si uživatel založí nebo se přihlásí do již vytvořeného účtu, skrze Jednoho ze dvou poskytovatelů této funkce tedy Google nebo Facebook od kterých si aplikace vyžádá ověřovací token, profilový obrázek, celé jméno a emailovou adresu. V databázi aplikace v tabulce „dbo.Users“ se poté vytvoří řádek s jejich emailovou adresou a výchozím nastavením pravomocí, tedy Creator.

Typy uživatelů dle pravomocí:

- Admin (nejvýš) – právo mazat písně, vidět zakázané písně a nastavovat pravomoci ostatním uživatelům.
- Moderator – právo editovat všechny skladby, vidět a schvalovat neschválené skladby.
- Creator – právo přidat skladbu, vidět vlastní přidané skladby.
- Visitor – právo vidět schválené skladby a seznam autorů, právo se přihlásit nebo zaregistrovat a odhlásit se.

Obrázek 3 - oprávnění – diagram případů užití (use case diagram)



7.1.7 Editor

Editor slouží k přidávání písní a jejich případné editaci, jsou-li již součástí databáze zpěvníku. Proces je rozdělen do dvou etap: zadání hlavních informací a přidání hudebních značek.

Zadání hlavních informací

Uživatel zde do připraveného formuláře v levé části stránky vyplní a zvolí údaje o skladbě, konkrétně:

- název skladby
- jméno autora/autorů textu
- jméno autora/autorů hudby
- jméno interpreta/interpretů
- důležitost
- polohu kapodasteru
- „publikovatelnost“ – jestli píseň obsahuje hanlivé výrazy
- URL odkaz písně na stránce www.youtube.com – nepovinné
- doporučený rytmický doprovod – nepovinné

Následovat bude vložení textu písně do pole ve středu stránky. Text musí být zformátován dle návodu zobrazeného pod přidávacím polem, tedy každý verš je na novém řádku a jednotlivé sloky odděluje jeden prázdný řádek. Možnost přidat píseň se nachází v druhé části, kam se uživatel dostane kliknutím na tlačítko „Přidat akordy“.

Přidání hudebních značek

Data z první části se vygenerují do podoby, ve které jsou některé zadané informace zobrazené v horní části. Text je rozdělen na jednotlivé sloky. Přidat prvky lze označením písmena nebo úseku textu a kliknutím na některý prvek z menu v levé části stránky.

Obsah menu:

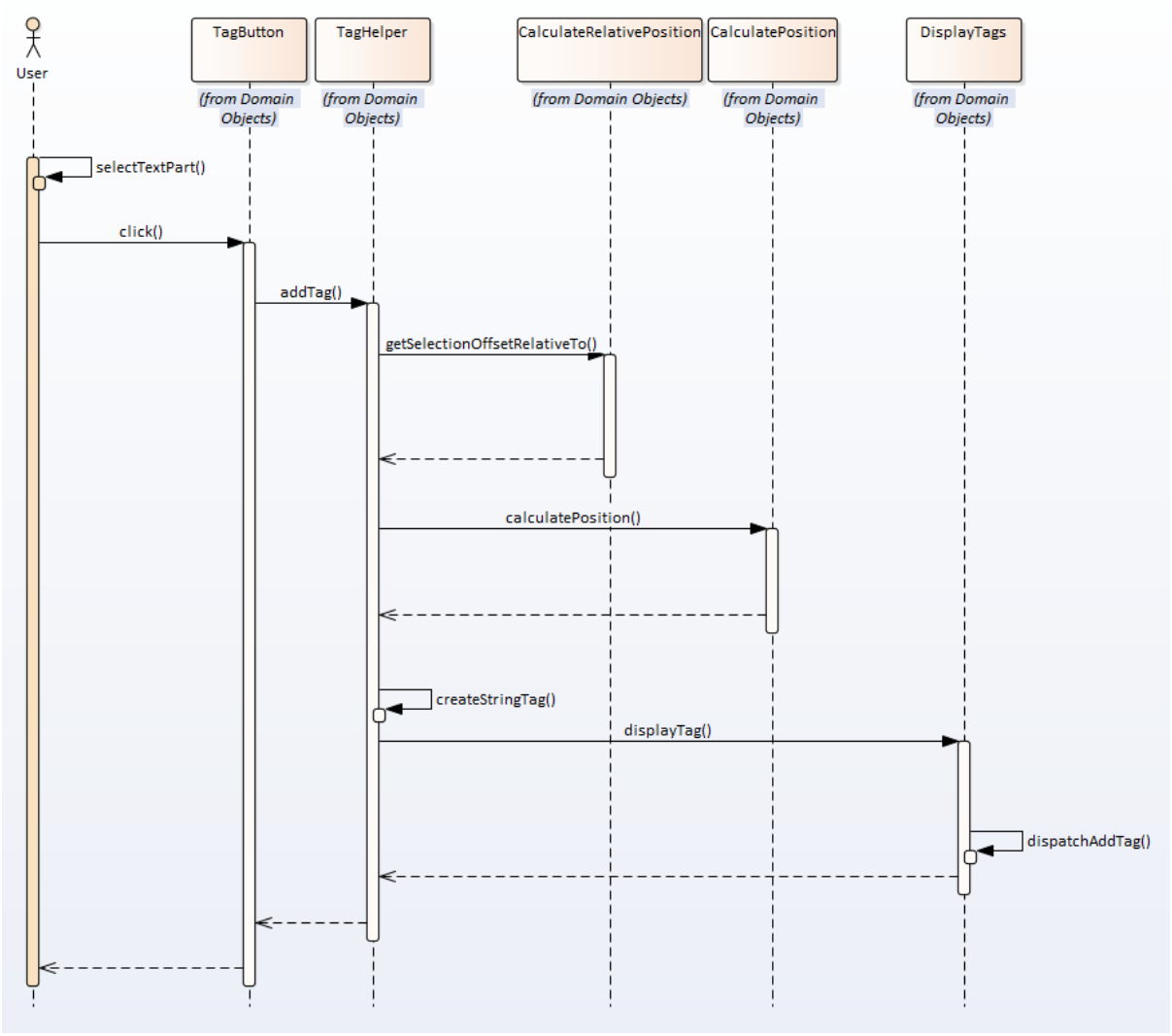
- možnost vybrat si akordy z předepsaných kadencí
- možnost přidání nového akordu
- prvky související s opakováním části skladby
- prvky související s dynamikou skladby
- tlačítka pro editaci textu

Popis algoritmu přidání značky:

Jednotlivá tlačítka mají již při generaci definované parametry a funkci, do které tyto parametry posílají. Přesněji je to funkce „addTag“, jež bere jako své parametry: typ značky, hodnotu značky, příponu značky, pořadí vložené značky a dispatch funkci pro získání hodnot z Redux store. Zde se získá relativní pozice označení textu (začátek a konec označení), ověří se, zda označení se opravdu týká editovaného textu a poté se spočítá pozice k relativně ~~k~~ nejbližší značce typu „VERSE“. Tyto pozice se poté znovu přepočítají, aby odpovídaly reálným hodnotám innerHTML objektu. Na ně se poté v innerHTML vloží textová reprezentace značky, jejíž součástí jsou určité parametry odlišné dle typu. Tím je tedy námi zvolená část textu obalena danou značkou.

Funkce „displayTag“ podle id vyhledá element přidané značky a získá potřebné informace. Ty použije k vytvoření vizuální reprezentace značky, která je doplněná o další informace, jako je pozice prvku a styl zobrazení. Poté jsou uložené do stavu, který po zjištění aktualizace znovu vykreslí objekt s vizuálními reprezentacemi znaků.

Obrázek 4 – přidání značky – sekvenční diagram



Dynamické umisťování prvků

Jelikož jsou vizuální reprezentace prvků umisťovány s absolutní pozicí vázanou na značku, je třeba je při každé změně rozměrů okna znovu přepočítat. Tuto funkci právě zastává funkce „rerender“, která najde všechny přidané značky a znovu přepočítá pozici jejich vizuálních protějšků dle aktuálního umístění značek v textu.

Tato funkce je volána i při prvotním načtení editoru, kvůli zobrazení již předgenerovaných značek a také při samotném zobrazování písně pro uživatele.

Menu aktivních akordů:

Zde se nachází uživatelem vybrané akordy. Jelikož právě akordy jsou elementem, který se přidává nejčastěji, je toto pole co nejblíže editačnímu poli a ve svislé poloze pro rychlý přístup kurzorem.

Editace hudebních značek

Přidané prvky lze kliknutím označit a upravit prostřednictvím editačního menu, zobrazeného pod menu s prvky. Pro každý druh prvku jsou zobrazené jemu dané možnosti editace. Všechny prvky jde odstranit a nebo upravovat jejich parametry.

Konec Editace

Po skončení editace uživatel může stiskem tlačítka „Odeslat“ poslat píseň a všechna její zadaná data ke zpracování serveru.

- API dotaz - „/api/song/SaveSong“

Editace uložené písně

Vybranou uloženou píseň může autor nebo moderátor znovu odeslat k editaci. Tato píseň se poté znovu zobrazí ve stejném formátu jako při přidávání písně a jde stejným způsobem editovat. Nejde však již změnit rozložení jednotlivých slok.

- API dotaz - „/api/song/EditSong“

8 Backend

8.1 Backend server

Server, zprostředkovávající jak přenos dat mezi frontendem a databázemi, tak zpracovává i Redis cache. Je napsaný v programovacím jazyku C# s ASP.NET Core MVC. Všechny části aplikace jsou řešeny pomocí CaaS (Container as a Service), tedy jejich image běží pod Dokrem/Kuberetes a jsou nastavovány v projektu „docker-compose“.

8.2 REST API

V API aplikaci můžeme najít tato volání (získáno ze Swagger UI):

Všechna volání se posílají metodou POST.

/song/GetSong – získá detaily písně nalezené podle id písně.

- požadavek – {“ SongId“: 0}
- odpověď – {"SongId": 0, "Date": "2022-08-08T00:00:00", "SongName": "string", "Validated": true, "Capo": 0, "Text": "string", "Authors": [{"name": "string", "type": 0}], "Rythms": []}

/song/SaveSong – uloží píseň do databáze.

- požadavek - {"SongName": "string", "Publicable": 0,"Priority": 0,"Capo": 0, "Url": "string", "Authors": [{"name": "string", "type": 0}], "Rythms": [0], "Text": "string"}
- odpověď – {}

/song/GetSongForEdit – získá detaily písně pro její editaci

- požadavek - {“ SongId“: 0}
- odpověď – {"Url": "string", "Rythms": [], "Authors": [{"type": 0, "name": "string"}], "Priority": 0, "Creator": 0, "Publicable": 0, "SongId": 0, "Song-Name": "string", "Text": "string", "Capo": 0}

/song/EditSong – uloží píseň do databáze jako novou verzi upravené písně.

- požadavek - {"Son-gId": 0, "SongName": "string", "Creator": 0, "Publicable": 0, "Capo": 0, "Url": "string", "Authors": [{"name": "string", "type": 0}], "Priority": 0, "Rythms": [0], "Text": "string", "Reprint": true}
- odpověď – {}

/song/DeleteSong – odstraní píseň z databáze.

- požadavek – {“ SongId“: 0}
- odpověď – {}

/song/ValidateSong – schválí píseň.

- požadavek – {“ SongId“: 0}
- odpověď – {}

/user/GetAllUsers – získá informace všech registrovaných uživatelů.

- požadavek – {}
- odpověď – [{“Id“: 0, “Name“: “string“, “Email“: “string“, “Permissions“: 0}]

/user/Login – přihlásí uživatele.

- požadavek – {"Type": "string", "Token": "string"}
- odpověď – {"Token": "string"}

/user/ChangeUserPermission – nastaví pravomoc uživateli.

- požadavek – {“Id“: 0, “Permissions“: 0}
- odpověď – {}

Search – vyhledá (parametrizovaná) data pomocí textového řetězce.

- požadavek – {“value“: “string“, “type“: 0}
- odpověď – závislé na parametrech

Výčtové typy:

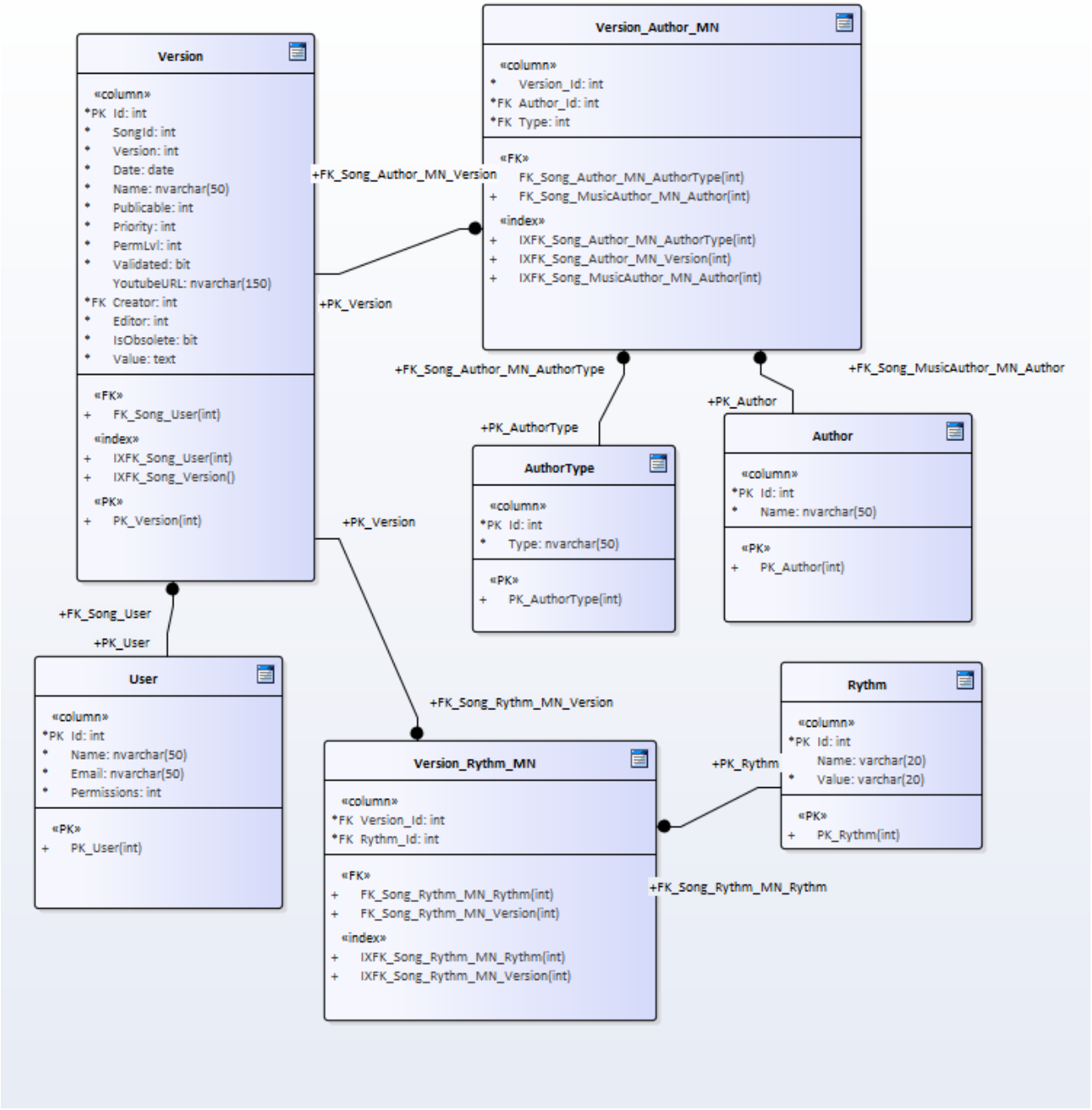
- UserType – seřadit podle
    - 0 – Visitor
    - 1 – Creator
    - 2 – Moderator
    - 3 – admin
- Groups
    - 0 – „publikovatelné“
    - 1 – „nepublikovatelné“
    - 2 – „neviditelné“
    - 3 – „zakázané“

8.3 Návrh databází

8.3.1 MSSQL databáze

O uchování a zálohu dat se stará relační databáze MSSQL. Zde jsou uchovány všechny historické verze písní a data uživatelů. Z této databáze získává svá data NoSQL databáze Elasticsearch a dále se na ni dotazuje aplikace z důvodů získání podrobných dat písní nebo uživatelů.

Obrázek 5 – návrh MSSQL databáze



Obrázek 6 – datový slovník MSSQL databáze

tabulka	sloupec	klíč	datový typ	nullable	autoinkrementace	popis	definiční obor
Version	Id	PK	int		✓	unikátní idenifkátor verze	
	SoingId		int			unikátní identifikátor písně	
	Date		date			datum přidání/upravení písně	
	Name		nvarchar(50)			název písně	
	Publicable		int			kategorie publikovatelnosti písně	0-2
	Priority		int			důležitost/povědomí o písni - 1 až 5 (5 - nejvyšší)	1-5
	PermlLvl		int			nejmešní úroveň pravomocí potřebný k zobrazení písně	0-3
	Validated		bit			označení zkotrolování správnosti písně	
	YoutubeURL		nvarchar(150)	✓		URL odkaz na píseň na serveru youtube.com	
	Creator	CK	int			Identifikátor uživatele, který původně píseň přidal.	
	Editor		int			Identifikátor uživatele, který píseň upravil a tím vytvořil novou verzi.	
	IsObsolete		bit			Určuje jestli se má píseň zobrazovat ve vysledcích	
	Value		text			XML písně	
User	Id		int		✓	unikátní idenifkátor uživatele	
	Name		nvarchar(50)			jméno uživatele	
	Email		nvarchar(50)			email uživatele	
	Permissions		int			úroveň pravomocí uživatele	
Version_Author_MN	Version_Id	CK	int			unikátní idenifkátor verze - cizí klíč	
	Author_Id	CK	int			unikátní idenifkátor autora - cizí klíč	
	Type		int			unikátní idenifkátor typu autora - cizí klíč	1-3
Author	Id	PK	int		✓	unikátní idenifkátor autora	
	Name		nvarchar(50)			jméno autora	
AuthorType	Id	PK	int		✓	unikátní idenifkátor typu autora	1-3
	Type		nvarchar(50)			název typu autora	
Version_Rythm_MN	Version_Id	CK				unikátní idenifkátor verze - cizí klíč	
	Rythm_Id	CK				unikátní idenifkátor rytmu - cizí klíč	
Rythm	Id	PK			✓	unikátní idenifkátor rytmu	
	Name		nvarchar(20)	✓		název rytmu	
	Value		nvarchar(20)			znaková interpretace rytmu	

Popis tabulek:

- Version – uchovává všechny údaje o skladbě, které nejsou ve vazbě m:n
- Autor – jména vložených autorů.
- AuthorType – typy autorů (autor textu, autor hudby, interpret…).
- User – informace o uživatelích a jejich přiřazená práva.
- Rythm – typy základních rytmů.
- Version_Author_MN – tabulka pro vazbu s m:n multiplicitou mezi tabulkami Version a Song, doplněna o typ autora pro danou vazbu.
- Version_Rythm_MN – tabulka pro vazbu s m:n multiplicitou mezi tabulkami Version a Rythm

Uložené procedury:

- dbo.EditSong – vytvoří novou verzi již přidané písně a starou verzi označí jako zastaralou.
- dbo.GetSong – získá potřebná data k zobrazení písně
- dbo.Login – zajišťuje zaznamenávání nových uživatelů
- dbo.GetAllSongs – získá potřebná data všech aktuálních písní a autorů pro uložení do Elastic databáze.

- dbo.GetAllAuthors – získá potřebná data všech aktuálních autorů pro uložení do Elastic databáze.
- dbo.GetElasticSong – získá data jedné skladby pro Elastic
- dbo.SaveSong – přidá píseň do databáze.

Ostatní získávání dat z databáze je prováděno formou přímých dotazů, jelikož nejsou tak složité a dlouhé pro nutnost tyto dotazy mít uložené v databázi.

8.3.2 NoSQL databáze

Tuto funkci zastupuje vyhledávácí a analytický nástroj Elasticsearch, který funguje i jako velmi efektivní NoSQL databáze, vhodná pro vyhledávání v textových datech. V této databázi jsou uložena pouze data aktuálních verzí písní, která jsou potřebná k vyhledávání a filtrování, doplněná o indexy pro co nejrychlejší vyhledávání. Data jsou tedy převedena do struktury, kde je možné tyto indexy na požadovaná místa doplnit. Dotazy na čtení i zápis dat zprostředkovává C# backend aplikace.

Celková synchronizace dat oproti zálohovací databázi, tedy smazání a znovunačtení databáze, probíhá každý den. Děje se tak, aby se předešlo možným chybným synchronizacím, které mohou nastat přidáváním dat do obou databází.

Obrázek 7 – Ukázka odpovědi databáze Elastic na fulltext vyhledávání

8.3.3 Cache

Redis cache, jakožto databáze, která uchovává nedávno vyžádaná data, je přizpůsobena k rychlému návratu velkého množství dat. Tato data ukládá na stanovenou dobu a poté je maže. Redis cache zapisuje data ve formátu <klíč, objekt> a klíčem

v tomto případě bývá celý požadavek na API, spojený v jeden řetězec, kde jsou jednotlivé členy odděleny dvojtečkou. Data je možné ukládat asynchronně, tedy při ukládání aplikace nečeká a výsledek nezpomaluje běh aplikace.

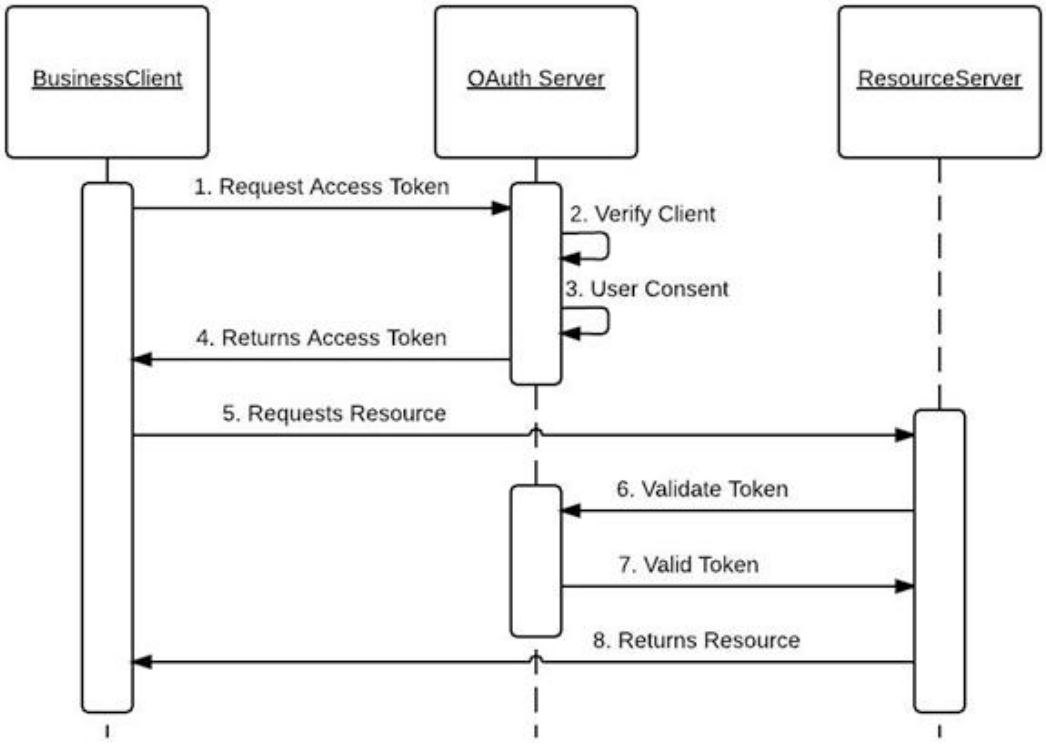
8.3.4 Dotazování pro získání dat

Aplikace po dotazu nejprve vyžádá data právě z Redis cache, pokud databáze hledané výsledky neobsahuje nebo jsou neúplné vzhledem k rozsahu dotazu, pokusí se tato data doplnit z MSSQL databáze nebo z Elastic databáze. Získaná data poté uloží do cache pod danými klíči. Tento postup je zvolen z důvodu nesouladu zobrazení uložených písní, kde některé jsou zobrazovány n-krát denně a jiné jsou zobrazeny jednou do týdne. Tímto je tedy docíleno rychlého načtení skladeb, které jsou navštěvovány nejčastěji.

8.4 Přihlášení a registrace

Přihlášení i registrace probíhají skrze OAuth2.0 protokol. Samotné ověřování přihlašovacích údajů tedy neprobíhá v aplikaci, ale skrze jeden ze dvou nastavených autentizačních serverů, přesněji Google a Facebook. Tyto servery ověří totožnost uživatele a v případě nového uživatele poskytnou aplikaci jeho jméno a emailovou adresu, které se uloží do databáze a tím vytvoří nový účet. Přihlašovací údaje tedy nejsou uchovávány nikde v aplikaci ani v databázích. V obou případech po ověření uživatel získá přístupový token, kterým autorizuje většinu požadavků na webové API.

Obrázek 8 – OAuth 2.0 – sekvenční diagram (32)



9 Hosting a CaaS

Aplikace je nahrána na cloud serverech společnosti Google.com, přesněji na jejich službě Google Cloud a je dostupná na adrese www.blivnik.eu.

Všechny části aplikace jsou sestaveny jako kontejnery, ty jsou vystavené do cloud prostředí s nastavenými globálními proměnnými a vnitřními porty, které lze nalézt v projektu docker-compose v souborech .env a docker-compose.yml.

10 Použitá vývojová prostředí

Visual Studio 2019

Visual Studio 2019 je 64bitové integrované vývojové prostředí od firmy Microsoft, zaměřené na vývoj mobilních, desktopových a webových aplikací. Programátoři nejvíce docení jeho propracovaný editor kódu s funkcí IntelliSense (napovídá a doplňuje kód), pokročilé refaktorování, debug, diagnostiku náročnosti a různorodé druhy sestavení aplikace s možností publikace na vzdálený server. Vlajkovými programovacími jazyky, kterými je možné v tomto prostředí psát, jsou C++, C#, F# a Python. Pomocí dodatečných zásuvných modulů lze docílit podpory pro více než 36 programovacích jazyků. (29) Autor použil toto prostředí převážně pro vývoj backendu aplikace.

Visual Studio Code

Visual Studio Code je lehký, ale výkonný editor zdrojového kódu, který běží na vašem počítači a je k dispozici pro Windows, MacOS a Linux. Dodává se s vestavěnou podporou pro JavaScript, TypeScript, Node.js a má bohatý ekosystém rozšíření pro další jazyky a běhové prostředí (např. C++, C#, Java, Python, PHP, Go, .NET). (30) Autor použil toto prostředí pro vývoj frontendu aplikace.

Microsoft SQL server management studio

SQL Server Management Studio (SSMS) je integrované prostředí pro správu jakékoli infrastruktury SQL, od SQL Serveru po Azure SQL Database. SSMS poskytuje nástroje pro konfiguraci, sledování a správu instancí SQL Serveru a databází. Pomocí SSMS se nasazují, monitorují a aktualizují komponenty datové vrstvy používané aplikacemi a sestavují dotazy a skripty. (31)

11 Shrnutí výsledků a závěr

Současná podoba aplikace splňuje stanovené cíle. Jednotlivé části webu jsou uživatelsky přívětivé a designově uspokojivé. Editor, jenž byl hlavní částí vývoje frontendu, se podařilo vytvořit tak, aby bylo přidávání akordů a ostatních hudebních značek velmi jednoduché, jak co se týče srozumitelnosti, tak i samotného procesu přidání. Bohužel se nepodařilo implementovat některé z hudebních značek a prvků jako jsou tabulatury, transpozice skladby a hlídání vstupu z hlediska enharmonické záměny. Tyto dílčí prvky budou v průběhu dalšího vývoje zpěvníku implementovány.

Kód v celé aplikaci je dostatečně strukturovaný a ošetřen proti většině možných útoků. Aplikace je extrémně škálovatelná díky využití CaaS, hostování na Google cloud a dobrému navržení databází, které je schopné odolat i velkému vytížení ze strany uživatelů. Velký důraz byl kladen na možnost dalšího vývoje aplikace, kdy mohou v budoucnu přibývat další funkcionality, jako jsou: tisk knihy, zobrazení informací o autorovi, možnost přidat další data k písni (např. album, rok vydání, zajímavosti aj.). Konečným produktem by měl být zpěvník, který bude splňovat veškeré potřeby zápisu písní.

Projekt má v budoucnu, dle autorova názoru, velký potenciál stát se populárním mezi hudebníky, a i v případě nezájmu veřejnosti bude využíván alespoň pro soukromé účely jako zpěvník a učební pomůcka.

12 Seznam použité literatury

1. **Neužil, Jiří a Benko, Matěj.** Skripta základů hudební teorie. *klavir.klusik.cz.* [Online] [Citace: 15. Květen 2022.] http://klavir.klusik.cz/wp-content/uploads/2017/06/skripta-hudebn%C3%AD-teorie.pdf.

2. **ZUŠ Semily.** zussemily.cz. *ZÁKLADNÍ HUDEBNÍ TEORIE.* [Online] [Citace: 11. Květen 2022.] https://www.zussemily.cz/images/ke_stazeni/ZAKLADNI_POJMY.pdf.

3. **Weiss, Piero a Taruskin, Richard .** *Music in the Western World.* místo neznámé : Cengage Learning, 2007. 053458599X, 9780534585990.

4. **Mozilla Foundation.** HTML: HyperText Markup Language. *mozilla.org.* [Online] 20. Duben 2022. [Citace: 1. Květen 2022.] https://developer.mozilla.org/en-US/docs/Web/HTML.

5. **Refsnes Data.** HTML Introduction. *w3schools.com.* [Online] [Citace: 1. květen 2022.] https://www.w3schools.com/html/html_intro.asp.

6. —. CSS Introduction. *w3schools.com.* [Online] [Citace: 9. Květen 2022.] https://www.w3schools.com/css/css_intro.asp.

7. **Mozilla Foundation.** What is JavaScript? *developer.mozilla.org.* [Online] 23. Březen 2022. [Citace: 27. Červen 2022.] https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript.

8. **Meta Platforms.** tutorial. *reaktjs.org.* [Online] 2022. [Citace: 15. Leden 2022.] https://reactjs.org/tutorial/tutorial.html.

9. **Creative Commons BY-SA.** React JS Notes for Professionals book. *goalkicker.com.* [Online] 16. Duben 2018. [Citace: 28. Leden 2022.] https://goalkicker.com/ReactJSBook/.

10. **Vailshery, Lionel Sujay.** Most used web frameworks among developers worldwide, as of 2021. *statista.com.* [Online] 23. únor 2021. [Citace: 15. červen 2022.] https://www.statista.com/statistics/1124699/worldwide-developer-survey-most-used-frameworks-web/.

11. **Refsnes Data.** React JSX. *w3schools.com.* [Online] [Citace: 11. Květen 2022.] https://www.w3schools.com/react/react_jsx.asp.

12. **Robie, Jonathan.** What is the Document Object Model? *w3.org.* [Online] 10. Leden 1998. [Citace: 28. Leden 2022.] https://www.w3.org/TR/REC-DOM-Level-1/introduction.html.

13. **JavaTpoint.** React Features. *javatpoint.com.* [Online] [Citace: 10. Květen 2022.] https://www.javatpoint.com/react-features.

14. **Meta Platforms.** Hooks FAQ. *reactjs.org.* [Online] [Citace: 10. Květen 2022.] https://reactjs.org/docs/hooks-faq.html.

15. **JavaTpoint.** React Redux. *javatpoint.com.* [Online] [Citace: 11. Červen 2022.] https://www.javatpoint.com/react-redux.

16. **Abramov, Dan.** Redux. *github.com.* [Online] [Citace: 11. Červen 2022.] https://github.com/reduxjs/redux.

17. **Microsoft.** .NET Programming Languages. *dotnet.microsoft.* [Online] 2022. [Citace: 15. Leden 2022.] https://dotnet.microsoft.com/en-us/languages.

18. **Hejlsberg, Anders, a další.** *The C# Programming Language.* místo neznámé : Pearson Education, 2008. 0321592255, 9780321592255.

19. **Smith, Steve.** docs.microsoft.com. *Overview of ASP.NET Core MVC.* [Online] 6. Březen 2022. [Citace: 26. Červen 2022.] https://docs.microsoft.com/en-us/aspnet/core/mvc/overview?WT.mc_id=dotnet-35129-website&view=aspnetcore-6.0.

20. **Amazon Web Services.** What is RESTful API? *amazon.com.* [Online] [Citace: 20. Červen 2022.] https://aws.amazon.com/what-is/restful-api/.

21. **Huges, Adam.** Microsoft SQL Server. *techtarget.com.* [Online] Květen 2019. [Citace: 15. Leden 2022.] https://www.techtarget.com/searchdatamanagement/definition/SQL-Server.

22. **Elasticsearch B.V.** What is Elasticsearch? *elastic.co.* [Online] [Citace: 22. Květen 2022.] https://www.elastic.co/what-is/elasticsearch.

23. **Parecki, Aaron.** OAuth 2.0. *oauth.net.* [Online] [Citace: 13. Červen 2022.] https://oauth.net/2/.

24. **Serilog Contributors.** serilog. *github.com.* [Online] [Citace: 22. Červen 2022.] https://github.com/serilog/serilog.

25. **Redis Ltd.** Introduction to Redis. *redis.io.* [Online] [Citace: 22. Červen 2022.] https://redis.io/docs/about/.

26. **Docker Inc.** Docker overview. *docs.docker.com.* [Online] [Citace: 25. Červen 2022.] https://docs.docker.com/get-started/overview/.

27. **Bigelow, Stephen.** Containers as a Service (CaaS). *techtarget.com.* [Online] Únor 2019. [Citace: 26. Červen 2022.] https://www.techtarget.com/searchitoperations/definition/Containers-as-a-Service-CaaS.

28. **IT-Slovnik.cz team.** Co je to Pop-up? *it-slovnik.cz.* [Online] [Citace: 15. Květen 2022.] https://it-slovnik.cz/pojem/pop-up.

29. **Microsoft.** Welcome to the Visual Studio IDE. *docs.microsoft.com.* [Online] 30. Duben 2022. [Citace: 25. Červen 2022.] https://docs.microsoft.com/en-us/visualstudio/get-started/visual-studio-ide?view=vs-2019.

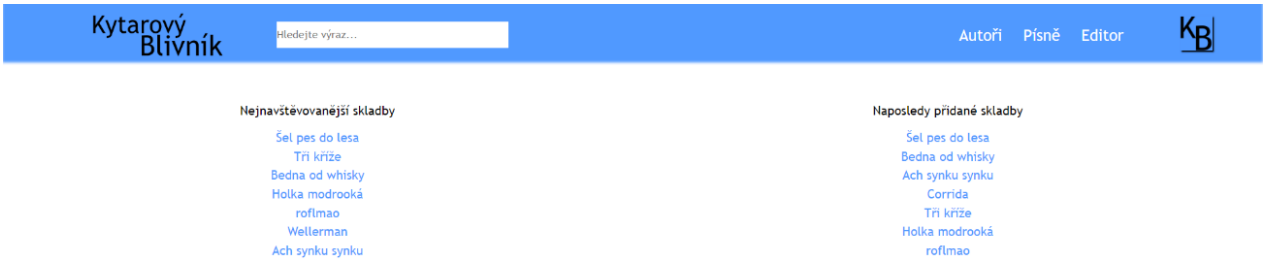
30. —. Getting Started. *code.visualstudio.com.* [Online] [Citace: 25. Červen 2022.] https://code.visualstudio.com/docs.

31. —. Download SQL Server Management Studio (SSMS). *docs.microsoft.com.* [Online] 21. Červen 2022. [Citace: 25. Červen 2022.] https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver16.

32. **Oracle.** OAuth 2.0. *docs.oracle.com.* [Online] [Citace: 23. Červen 2022.] https://docs.oracle.com/cd/E82085_01/160027/JOS%20Implementation%20Guide/Output/oauth.htm.

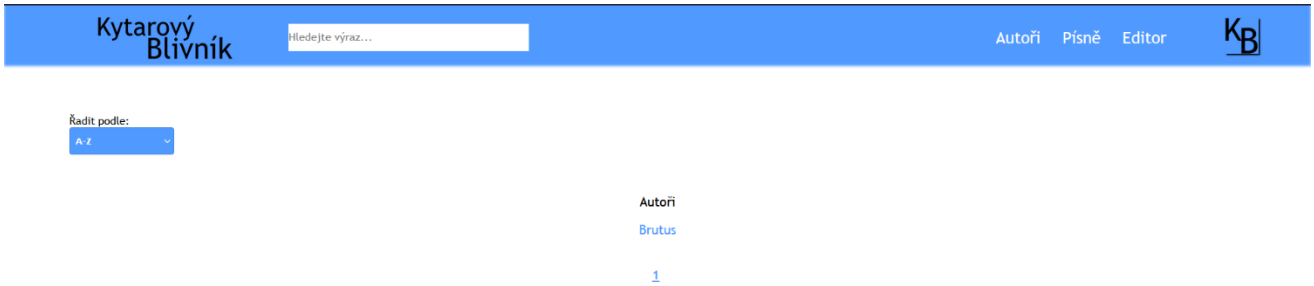
13 Přílohy

Obrázek 9 – hlavní strana

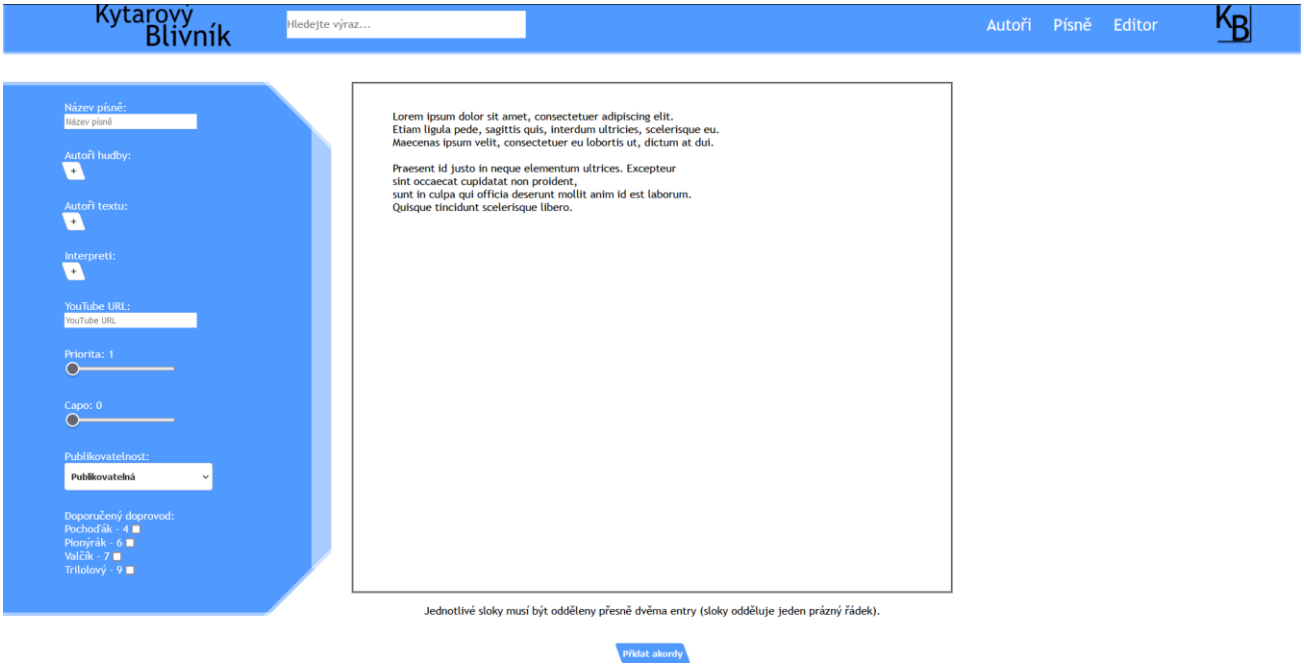


Obrázek 10 – seznam písní

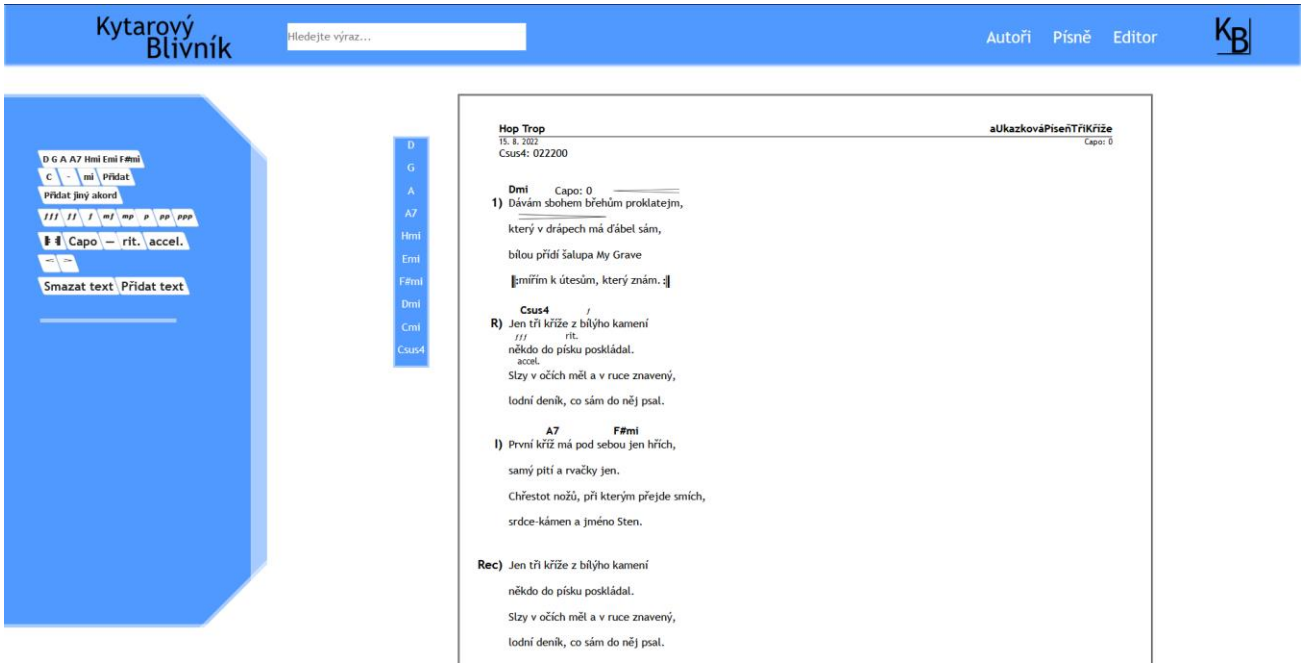
Obrázek 11 – seznam autorů



Obrázek 12 – první fáze přidání písně



Obrázek 13 – druhá fáze přidání písně



Obrázek 14 – zobrazení písně

Obrázek 15 – autorovy písně

14 Seznam obrázků

15 Zadání práce

UNIVERZITA HRADEC KRÁLOVÉ
Fakulta informatiky a managementu
Akademický rok: 2020/2021

Studijní program: Aplikovaná informatika
Forma studia: Prezenční
Obor/kombinace: Aplikovaná informatika (ai3-p)

Podklad pro zadání BAKALÁŘSKÉ práce studenta

Jméno a příjmení:	David Jiroudek
Osobní číslo:	I1900731
Adresa:	Riegrova 769, Velký Osek, 28151 Velký Osek, Česká republika
Téma práce:	Webový zpěvník
Téma práce anglicky:	The Web Song Book
Vedoucí práce:	Ing. Karel Malý, Ph.D. Katedra informatiky a kvantitativních metod

Zásady pro vypracování:

Cílem práce je vytvoření webové aplikace pro zápis písní a hudebních komponentů se zaměřením na jejich přesnost vůči textu. Důraz bude kladen na implementaci přívětivého uživatelského rozhraní pro vytváření a editaci. V rámci práce budou popsány základní vlastnosti použitých technologií.
Osnova:
Úvod
Teoretická část
Analýza a návrh
Praktická část
Závěr
Literatura

Seznam doporučené literatury:

Dokumentace k jazyku C# [online]. Dostupné z https://docs.microsoft.com/cs-cz/dotnet/csharp.
React [online]. Dostupné z https://reactjs.org.
SQL Server technical documentation [online]. Dostupné z https://docs.microsoft.com/en-us/sql/sql-server/?view=sql-server-ver15.

Podpis studenta: Jiroudek Datum: 29. 6. 2022

Podpis vedoucího práce: Malý Datum: 29. 6. 2022