Vysoká škola ekonomická v Praze

Fakulta informatiky a statistiky

Návrh webové aplikace založené na přístupu Progressive Web Apps

BAKALÁŘSKÁ PRÁCE

ve studijním programu Aplikovaná informatika

Autor: Ho Anh Tu

Vedoucí bakalářské práce: Ing. Zuzana Šedivá, Ph.D.

Praha, květen 2019

Prohlášení

Prohlašuji, že jsem bakalářskou práci s názvem Návrh webové aplikace založené na přístupu Progressive Web Apps vypracoval samostatně za použití v práci uvedených pramenů a literatury.

V Praze dne 6. května 2019 ……..

Ho Anh Tu

Poděkování

Chtěl bych poděkovat rodině a přátelům za důvěru a podporu v těžkých chvílích. Dále bych rád poděkoval Ing. Zuzaně Šedivé, Ph.D. za její dohled a trpělivost nad mojí práci.

Abstrakt

Tato práce se zabývá návrhem a implementací webové aplikace na přístupu Progressive Web Apps a je určena pro čtenáře, kteří se zajímají o přístup Progressive Web Apps a jeho tvorbu. Cílem práce je návrh a vytvoření webové aplikace SoundForYou s využitím metodiky pro malé softwarové projekty (MMSP) a přístupy mobile first a Progressive Web Apps. Práce je rozdělená na teoretickou a praktickou část. V teoretické časti práce jsou v první kapitole stručně popsány webové a mobilní aplikace a jejich porovnání, dále je v samostatné kapitole popsán přístup Progressive Web Apps. V praktické části je hlavním cílem práce návrh a tvorba webové aplikace pomocí metodiky MMSP a s využitím přístupů mobile first a Progressive Web Apps.

Výstupem bakalářské práce je webová aplikace a zhodnocení parametrů nově vytvořené aplikace založené na přístupu Progressive Web Apps.

Klíčová slova

Progressive Web Apps, MMSP, webová aplikace, mobile first.

Abstract

This thesis deals with the design and implementation of a web application on Progressive Web Apps approach and is designed for readers interested in Progressive Web Apps approach and its creation. The goal of this thesis is to design and create web application SoundForYou using the methodology for small software projects (MMSP) and mobile first and Progressive Web Apps approaches. The thesis is divided into theoretical and practical part. In the theoretical part of the thesis, the first chapter briefly describes the web and mobile applications and their comparison. In the practical part, the main goal of the thesis is to design and create a web application according to the methodology MMSP and using mobile first and Progressive Web Apps approaches.

The result of the thesis is a web application and evaluation of parameters of a newly created application on Progressive Web Apps approach.

Keywords

Progressive Web Apps, MMSP, web application, mobile first.

Obsah

Úvod ... 11
Cíle práce ... 12
Předpoklady práce ... 12
Omezení práce ... 12
Přínos práce ... 12
Struktura práce ... 12
Rešerše ... 13
1 Webové a mobilní aplikace ... 14
1.1 Typy mobilních aplikací ... 14
Webová aplikace ... 14
Nativní mobilní aplikace ... 14
Hybridní mobilní aplikace ... 14
1.2 Náklady na vývoj ... 15
1.3 Porovnání mobilních operačních systému ... 15
2 Progressive Web Apps ... 16
2.1 Vlastnosti PWA ... 16
2.2 Komponenty PWA ... 17
2.2.1 Manifest ... 17
2.2.2 Service Workers ... 18
2.2.3 Aplikační shell ... 19
2.2.4 Web push notifikace ... 19
3 Použitá metodika pro návrh a tvorbu webové aplikace ... 21
MMSP ... 21
4 Použité technologie a nástroje pro návrh a tvorbu webové aplikace ... 23
CSS ... 23
HTML ... 23
JavaScript ... 23
jQuery ... 23
Lighthouse ... 24
Sublime Text ... 24
Web Server for Chrome ... 24
5 Průzkum trhu podobných aplikací ... 25

6 Specifikace požadavků na aplikaci podle MMSP ... 26
6.1 Vymezení vize ... 26
6.1.1 Vymezení problému ... 26
6.1.2 Vymezení produktu ... 26
6.1.3 Popis všech zúčastněných subjektů ... 26
6.1.4 Uživatelské rozhraní ... 27
6.1.5 Požadavky a funkce ... 27
6.2 Požadavky ... 28
6.2.1 Funkční požadavky ... 28
6.2.2 Nefunkční požadavky ... 28
6.2.2.1 Použitelnost ... 28
6.2.2.2 Spolehlivost ... 29
6.2.2.3 Výkon ... 29
6.2.3 Rozhraní ... 29
6.2.3.1 Uživatelské rozhraní ... 29
6.2.3.2 Vzhled ... 29
6.2.3.3 Rozložení a navigace ... 30
6.2.3.4 Konzistence ... 30
6.2.4 Personalizace a customizace ... 30
6.2.5 Obchodní pravidla ... 30
6.2.5.1 Čerpání dat ... 30
6.2.6 Systémová omezení ... 30
6.3 Případy užití ... 31
6.3.1 Diagram případů užití ... 31
6.3.2 Definice aktérů ... 31
6.3.3 Specifikace případu užití ... 31
7 Návrh webové aplikace ... 35
7.1 Zadání ... 35
7.2 Specifikace požadavků ... 35
7.2.1 Obecné požadavky ... 35
7.2.2 Detailní požadavky ... 35
7.3 Definování cílové skupiny ... 36
7.4 Tvorba loga a ikonek ... 36
7.5 Drátový model aplikace ... 36

7.6 Minimalistický design aplikace ... 36
7.7 Responzivní přístup ke tvorbě aplikace ... 36
8 Implementace aplikace ... 37
8.1 Zdrojový kód aplikace ... 37
8.1.1 Struktura souborů ... 37
8.1.2 Základní struktura HTML souborů ... 37
8.1.3 Tvorba menu ... 37
8.1.4 Aplikování stylů ... 38
8.1.5 Volání API ... 40
8.1.5.1 YouTube Data API ... 40
8.1.5.2 SoundCloud API ... 41
8.2 Přidání komponent PWA ... 42
8.2.1 Aplikační shell ... 42
8.2.2 Manifest ... 42
8.2.3 Service Worker ... 43
8.2.4 Push notifikace ... 45
8.3 Testování vytvořené mobilní aplikace ... 45
8.3.1 Testování aspektů PWA aplikace na vybraných webových prohlížečích ... 46
8.3.2 Testování dalších aspektů aplikace ... 46
8.3.3 Výsledek testování pomocí nástroje Lighthouse ... 47
8.4 Shrnutí ... 47
Závěr ... 48
Použitá literatura ... 49
Přílohy ... I
Příloha A: Logo a ikonka aplikace ... I
Příloha B: Drátové modely aplikace ... II
Příloha C: Snímky aplikace ... III
Příloha D: Zdrojový kód aplikace ... X

Seznam obrázků

Obrázek 1 Porovnání celosvětového počtu uživatelů mobilních OS v % (StatCounter, 2019) 15
Obrázek 2 Podpora manifestu v prohlížečích (Deveria, 2019) 18
Obrázek 3 Podpora service worker v prohlížečích (Deveria, 2019) 19
Obrázek 4 Podpora push notifikací v prohlížečích (Deveria, 2019) 20
Obrázek 5 Diagram případů užití (autor) 31
Obrázek 6 Vytvoření nastavení vzhledu aplikace (autor) 38
Obrázek 7 Definování stylů pro všechny elementy (autor) 38
Obrázek 8 Definování zobrazení obsahu menu (autor) 38
Obrázek 9 Stylizace tagů html a footer (autor) 39
Obrázek 10 Stylizace seznamů hudeb a jejich kontejnerů (autor) 39
Obrázek 11 Mediální dotaz pro tablety a větší displeje (autor) 40
Obrázek 12 Výpis parametrů Google API (autor) 40
Obrázek 13 Nastavení klíče a načtení Youtube API v3 (autor) 41
Obrázek 14 Definování parametrů omezení pro hledání YouTube videí (autor) 41
Obrázek 15 Metoda fetch pro přístup k SoundCloud API (autor) 41
Obrázek 16 Generátor souboru manifest společnosti Firebase (Firebase, 2019) 42
Obrázek 17 Připojení manifestu k HTML souboru (autor) 43
Obrázek 18 Registrace service worker (autor) 43
Obrázek 19 Název a definování souborů pro mezipaměť (autor) 43
Obrázek 20 Událost install pro service worker (autor) 44
Obrázek 21 Událost activate pro service worker (autor) 44
Obrázek 22 Událost fetch pro service worker (autor) 44
Obrázek 23 Požadavek o zobrazení notifikace (autor) 45
Obrázek 24 Podmínka pro zobrazení notifikace v události click (autor) 45
Obrázek 25 Funkce pro zobrazení notifikace (autor) 45
Obrázek 26 Testování přístupu PWA aplikace pomocí nástroje Lighthouse (autor) 47

Seznam tabulek

Tabulka 1 Prvky metodiky MMSP (Rejnková, 2011) 22
Tabulka 2 Vymezení problému (autor) 26
Tabulka 3 Vymezení produktu (autor) 26
Tabulka 4 Popis všech zúčastněných subjektů (autor) 26
Tabulka 5 Požadavky a funkce (autor) 27
Tabulka 6 Funkční požadavky (autor) 28
Tabulka 7 Použitelnost (autor) 28
Tabulka 8 Spolehlivost (autor) 29
Tabulka 9 Výkon (autor) 29
Tabulka 10 Vzhled (autor) 29
Tabulka 11 Rozložení a navigace (autor) 30
Tabulka 12 Konzistence (autor) 30
Tabulka 13 Personalizace a customizace (autor) 30
Tabulka 14 Systémová omezení (autor) 30
Tabulka 15 Definice aktérů (autor) 31
Tabulka 16 Specifikace případu užití přehrání hudby (autor) 32
Tabulka 17 Specifikace případu užití přidání ikonky (autor) 33
Tabulka 18 Specifikace případu užití vyhledání hudby (autor) 34
Tabulka 19 Testování aplikace na mobilních prohlížečích (autor) 46

Úvod

Dnešní doba je přehlcena různými zařízeními, pomocí kterých lze přistupovat k webu. Mezi nejprodávanějšími zařízeními patří chytré telefony, u kterých se odhaduje zhruba na 2,7 miliard uživatelů v roce 2019 a předpokládá se, že čísla stále porostou (Statista, 2019).

Výše uvedená fakta jsou jedním z důvodů, proč se firmy vytvářející SW aplikace v posledních letech zaměřují na mobilní trh. Již delší dobu neslouží mobilní telefony pouze k přijímání a odesílání SMS zpráv či k volání, ale také slouží k provádění různých bankovních operací, brouzdání na internetu nebo k online nakupování.

Jedním z nevýhod nativních mobilních aplikací je to, že se jejich tvorba zaměřuje na určitý operační systém, který vyžaduje specifické technologie na tvorbu aplikace a tím jsou náklady na jejich vývoj a údržbu aplikace pro různé operační systémy razantně vyšší než v případě webových aplikací.

Webové aplikace se řadí mezi multiplatformní aplikace. Jejich přístup je možný z jakéhokoliv webového prohlížeče a z jakéhokoliv operačního systému nebo zařízení podporující webové prohlížeče. Vzhledem k tomu, že se jedná o tvorbu aplikace s nejnižšími náklady na vývoj a údržbu, jedná se o často využívanou formu při vývoji aplikace pro firmy, zejména startupy.

Pro vývojáře platí úkol zaměřit se na vývoj webových a jiných multiplatformních aplikací, které jsou přizpůsobené většině mobilních zařízení. Jedním z přístupů je vytvořit responzivní design. Tento přístup umožňuje přizpůsobit webové aplikace konkrétnímu zařízení v závislosti na velikosti displeje.

U tvorby responzivního designu lze využít různých frameworků, které šetří čas a usnadňují práci. Mezi nimi patří zejména CSS Frameworky, jedny nejvyužívanější jsou Bootstrap a Foundation. Jedná se o knihovny, které obsahují předem definované třídy a komponenty. Vývojáři tak nemusí vytvářet vlastní styly, mohou využít frameworků.

Vývojářská komunita i uživatelé aplikací vyžadují více než lepší zážitek z webových aplikací. Chtějí, aby se aplikace chovaly jako nativní aplikace. Uživatelé některých webových aplikací také požadovali, aby se daná aplikace načetla i bez připojení k internetu. Toho lze dosáhnout pomocí progresivního webového přístupu zvaného Progressive Web Apps.

V dnešní době se webové stránky už dávno neprohlížejí pouze na desktopových zařízeních, o prvenství je sesadila mobilní zařízení. Pokud je prioritou udělat dojem a přístupnost webových stránek, tyto technologie k nim nesmí chybět. Tvorba aplikace založené na webových technologiích jsou přístupné každému, kdo vlastní zařízení s přístupem na webový prohlížeč a internet a mají nejnižší náklady na vývoj a údržbu ve srovnání s ostatními typy aplikací. A to je důvod, proč jsem si vybral toto téma bakalářské práce.

Cíle práce

Hlavním cílem této bakalářské práce je návrh a vytvoření webové aplikace SoundForYou s využitím metodiky MMSP a přístupy mobile first a Progressive Web Apps.

Výsledkem bakalářské práce je webová aplikace a zhodnocení parametrů nově vytvořené aplikace založené na přístupu Progressive Web Apps.

Předpoklady práce

Práce je zaměřená převážně na začátečníky v oblasti tvorby webové aplikace a implementace přístupu Progressive Web Apps, přesto by čtenář měl mít alespoň základní znalosti jazyka HTML, CSS a JavaScriptu.

Omezení práce

Přestože existuje mnoho webových prohlížečů pro mobilní zařízení, práce především přizpůsobuje vývoj webové aplikace pro prohlížeč Google Chrome, který jako nejvyužívanější webový prohlížeč na světě plně podporuje webový přístup Progressive Web Apps.

Přínos práce

Přínosem této práce je především základní přehled o webovém přístupu Progressive Web Apps a ucelený návod, jak navrhnout a vytvořit webovou aplikaci využívající webový přístup Progressive Web Apps.

Struktura práce

V první části se čtenář seznámí s multiplatformními mobilními aplikacemi a základními principy webového přístupu Progressive Web Apps.

Praktická část se zabývá návrhem a tvorbou webové aplikace pomocí metodiky MMSP a založenou na přístupech mobile first a Progressive Web Apps. Vedlejší prací je vytvoření loga a ikonky pro aplikaci SoundForYou.

Na závěr je provedeno zhodnocení parametrů pomocí testování aplikace založené na přístupu Progressive Web Apps.

Rešerše

Tato kapitola se věnuje popisu tištěných prací, akademických prací a internetových zdrojů, ze kterých bylo čerpáno a jsou pro tuto práci relevantní, případně danou práci doplňují.

Publikace *Vytváříme mobilní web a aplikace* od trojice autorů Earla Casteldina, Mylese Eftose a Maxe Wheelera (Castledine, et al., 2011) se zabývá tvorbou webu pro mobilní zařízení využívajících funkcí moderních mobilních zařízení. Popsaný návod na tvorbu webu pro mobilní zařízení jsem částečně aplikoval na svoji webovou aplikaci.

Publikace *Responzivní design – profesionálně* od Tima Kadlece (Kadlec, 2013) se zabývá tvorbou webu přizpůsobitelnou pro jakékoliv zařízení pomocí responzivního designu. Popsaná technika tvorby webu pomocí přístupu mobile first pomocí mediálních dotazů byla částečně uplatněna v mé práci.

Publikace *Moderní web – Perfektní stránky v každém prohlížeči a na každém zařízení* od Petra Gasstona (Gasston, 2013) se zabývá tvorbou webu pro různé prohlížeče na různých zařízeních. Obsahuje mimo jiné téma, jak správně rozvrhnout stránku, která byla využita v mé práci.

Bakalářská práce Pavla Břouška s názvem *Evaluation and usage of Google Progressive Web Apps technology* (Břoušek, 2017) se zabývá analýzou přístupu progresivní webové aplikace a tvorbou mobilního terminálu s přístupem Progressive Web Apps pro již existující aplikaci. Moje práce se zaměřuje na kompletní návrh a tvorbu webové aplikace s přístupem progresivní webové aplikace.

Bakalářská práce Parbata Thakura s názvem *Evaluation and Implementation of Progressive Web Application* (Thakur, 2018) se zabývá zhodnocením a implementací přístupu Progressive Web Apps. Autor tvoří aplikaci pomocí javascriptové knihovny pro tvorbu webových komponent React.js, v mé práci vytvářím aplikaci pomocí webových technologií bez využití podobných knihoven.

1 Webové a mobilní aplikace

Webová aplikace je aplikace běžící na webovém prohlížeči a poskytovaná uživatelům z webového serveru přes počítačovou síť Internet nebo intranet (Wikipedie, 2017). Mobilní aplikace je softwarová aplikace vytvořená speciálně pro chytré telefony, tablety a další mobilní zařízení. Mezi nejznámějšími českými mobilními aplikacemi jsou např. IDOS nebo Liftago (Wikipedie, 2018).

1.1 Typy mobilních aplikací

Webová aplikace

Webová aplikace je aplikace, která běží na zařízeních podporujících webový prohlížeč a je nezávislá vůči operačním systémům. V rozmachu využití mobilních zařízení se většinou vytvářejí přístupem *mobile first* (Xia, 2017), kde se prvně zaměřuje na tvorbu aplikace pro mobilní zařízení, poté pro desktopové zařízení. Nevýhodou webové aplikace je, že podporuje nižší využitelnost spolupráce s hardwarem mobilních zařízení.

Nativní mobilní aplikace

Nativní aplikace je aplikace, která je vytvořena a přizpůsobena pro určitý operační systém (dále jen OS), mezi kterými patří např. Android nebo iOS. Charakter nativní aplikace je, že ji není možné přenášet mezi různými platformy, je potřeba vytvořit danou aplikaci pro různé OS zvlášť. Mají plnou podporu využitelnosti hardwaru mobilních zařízení. Nativní aplikace je nutné stáhnout přes specifický obchod s aplikacemi např. obchod Google Play pro zařízení s operačním systémem Android.

Hybridní mobilní aplikace

Hybridní aplikace je využití funkcí webové technologie k tvorbě nativních mobilních aplikací. Využívá webové technologie HTML, CSS a JavaScript k tvorbě aplikace pomocí specializovaných nástrojů jako je Apache Cordova, která se následně konvertuje na nativní aplikace pro různé platformy, tzv. stačí vytvořit aplikaci pouze jednou a publikovat na různé obchody s aplikacemi. Oproti webovým aplikacím nepotřebují k běhu prohlížeč, mají přístup na většinu hardwaru mobilních zařízení a lze si je stáhnout na zařízení.

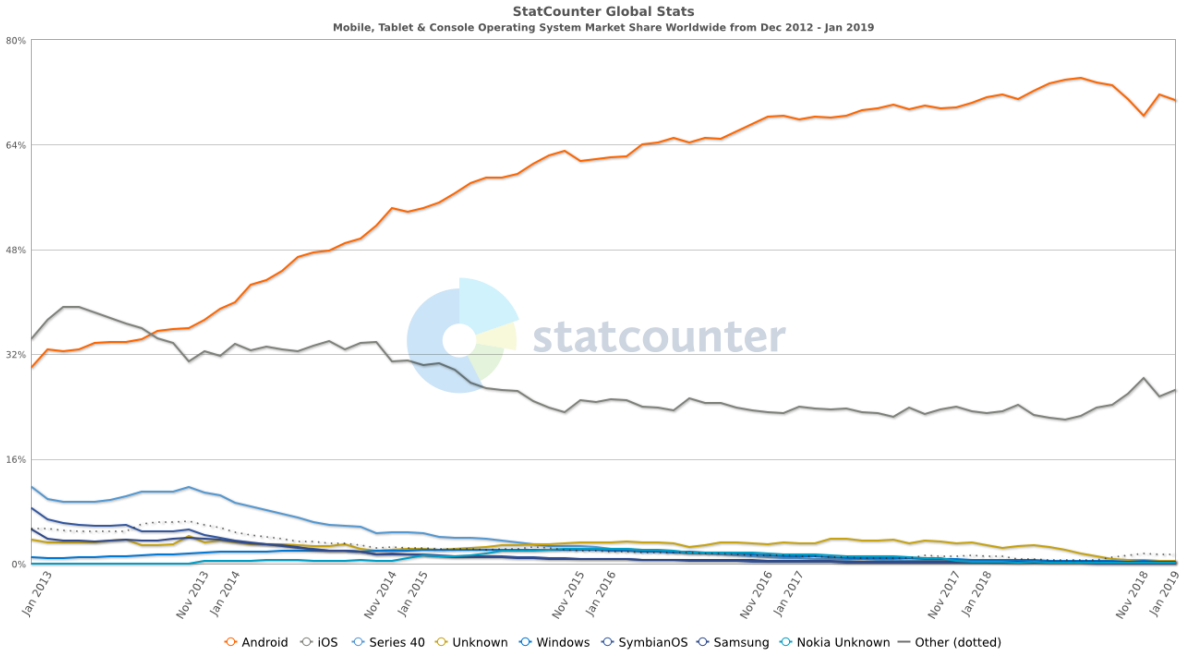
Mezi mobilními aplikacemi patří i progresivní webové aplikace jako nástupce výše zmíněných aplikací a jsou popsány dále.

1.2 Náklady na vývoj

Největší náklady na vývoj patří nativním aplikacím (archer-soft, 2017), kde se vývoj aplikace upírá zejména na OS a požadovaný programovací jazyk na jeho vývoj. Například na vývoj aplikace pro OS Android je potřeba znát programovací jazyk Java, kdežto pro OS iOS je třeba umět programovací jazyk Swift nebo Objective-C. Webové aplikace mají nejmenší náklady na vývoj a údržbu (Prasad, 2018), kde jejich tvorba vyžaduje pouze znalosti HTML, CSS a určitého skriptovacího jazyka a výsledná aplikace běží na kterémkoli prohlížeči jakéhokoliv operačního systému, takže stačí vytvořit aplikaci pouze jednou. Hybridní aplikace jsou levnější než nativní aplikace a při velké podobnosti aplikací vyvíjené na těchto dvou přístupech se ušetří více než polovinu nákladů na vývoj aplikace při vývoji pro OS Android a iOS oproti nativnímu přístupu (iQuest, 2018).

1.3 Porovnání mobilních operačních systému

Mobilní OS je software, který umožňuje chod aplikací a programů na chytrých telefonech, tabletech a dalších mobilních zařízení. Mezi nejpoužívanější OS dnešní doby se řadí Android a iOS.



Obrázek 1 Porovnání celosvětového počtu uživatelů mobilních OS v % (StatCounter, 2019)

Obrázek 1 zobrazuje celosvětový počet uživatelů operačních systémů v procentech od ledna roku 2013 do ledna roku 2019. V lednu 2013 měl Android okolo 30 % a iOS 34 % celosvětového počtu uživatelů. Zhruba v polovině roku 2013 došlo k převratu mezi operačními systémy Android a iOS, kdy celosvětový počet uživatelů začal rapidně stoupat u Android a klesat u iOS. V lednu 2019 měl Android zhruba 70 % a iOS 28 % celosvětového počtu uživatelů. U ostatních operačních systémů je patrný řádově menší počet uživatelů.

2 Progressive Web Apps

Společnost Google poprvé použila termín Progressive Web Apps (dále jen PWA) v roce 2015 k popsání aplikací využívajících výhod nových funkcí moderních webových prohlížečů jako jsou *Service Workers* a webové aplikační manifesty (Bartík, 2017). Tyto a další funkce jsou nezbytné pro tvorbu webových aplikací, aby se chovaly jako nativní aplikace. PWA si kladou za cíl umožnit webovým aplikacím možnosti využití hardwaru mobilních zařízení podobně jako u nativních mobilních aplikací, jako jsou například *push* notifikace, možnost spustit aplikaci bez připojení na internet a režim celé obrazovky.

Výhodou PWA oproti nativním aplikacím je, že není závislá na operačním systému zařízení a není nutná instalace přes různé internetové obchody. Další výhodou je, že se vytváří pomocí webových technologií, a proto nese nižší náklady na vývoj a údržbu oproti nativním aplikacím. Mezi světové aplikace používající přístup PWA můžeme řadit např. AliExpress, Google Maps, Instagram, Twitter nebo Uber. Tyto a mnoho dalších společností, které zahrnuly PWA do svých webových aplikací, pocítily nárůst návštěvnosti nebo délky relace uživatele na webové aplikaci (Oragui, 2018).

2.1 Vlastnosti PWA

Společnost Google (Google Developers, 2018) definuje aplikace na přístupu PWA jako:

- Progresivní – fungují pro uživatele jakéhokoliv zařízení
- Responzivní – přizpůsobují se jakémukoliv zařízení
- Nezávislá na konektivitě
- App-like – uživatel má při interakci pocit, jako by používal nativní aplikaci
- Fresh – vždy aktuální data díky procesu update *service workers*
- Zabezpečená – bez implementace HTTPS a SSL certifikátu se PWA neobejde
- Naleznutelná – jsou identifikovány jako aplikace díky W3C manifestu a registrace *service workers* napomáhá vyhledávačům k jejich nalezení
- Znovuzapojení uživatele – usnadňuje znovuzapojení uživatele pomocí funkcí jako jsou *push* notifikace
- Instalovatelná – umožňují uživatelům přidat aplikace na jejich domovskou stránku bez zdlouhavého procesu instalace v App store
- Odkazovatelná – jednoduše sdílená pomocí URL bez nutnosti složité instalace

Aby si uživatel mohl stáhnout PWA, aplikace musí podle (LePage, 2019) splňovat následující kritéria:

- Webový aplikační manifest soubor stránky musí mít alespoň čtyři povinné vlastnosti, tj. `name` a/nebo `short_name`, `icons`, `start_url` a `display`.
- Aplikace musí být zabezpečená.

- Aplikace musí mít registrovaný *service worker*.

Uvádím základní definici termínu PWA podle (Bevacqua, 2017): *„Progressive Web Apps je progresivní vylepšení aplikace, které neovlivní nepodporované prohlížeče. Uživatelé prohlížečů, které Progressive Web Apps podporují, si toho budou velice vážit.“*

2.2 Komponenty PWA

Ke splnění požadavků progresivních webových aplikací založených na přístupnosti, výkonu, *best practices* a SEO je zapotřebí využití komponentů PWA, tj. *Service Workers*, webový aplikační manifest, aplikační *shell* model a web *push* notifikace. Všechny tyto komponenty jsou popsány níže.

2.2.1 Manifest

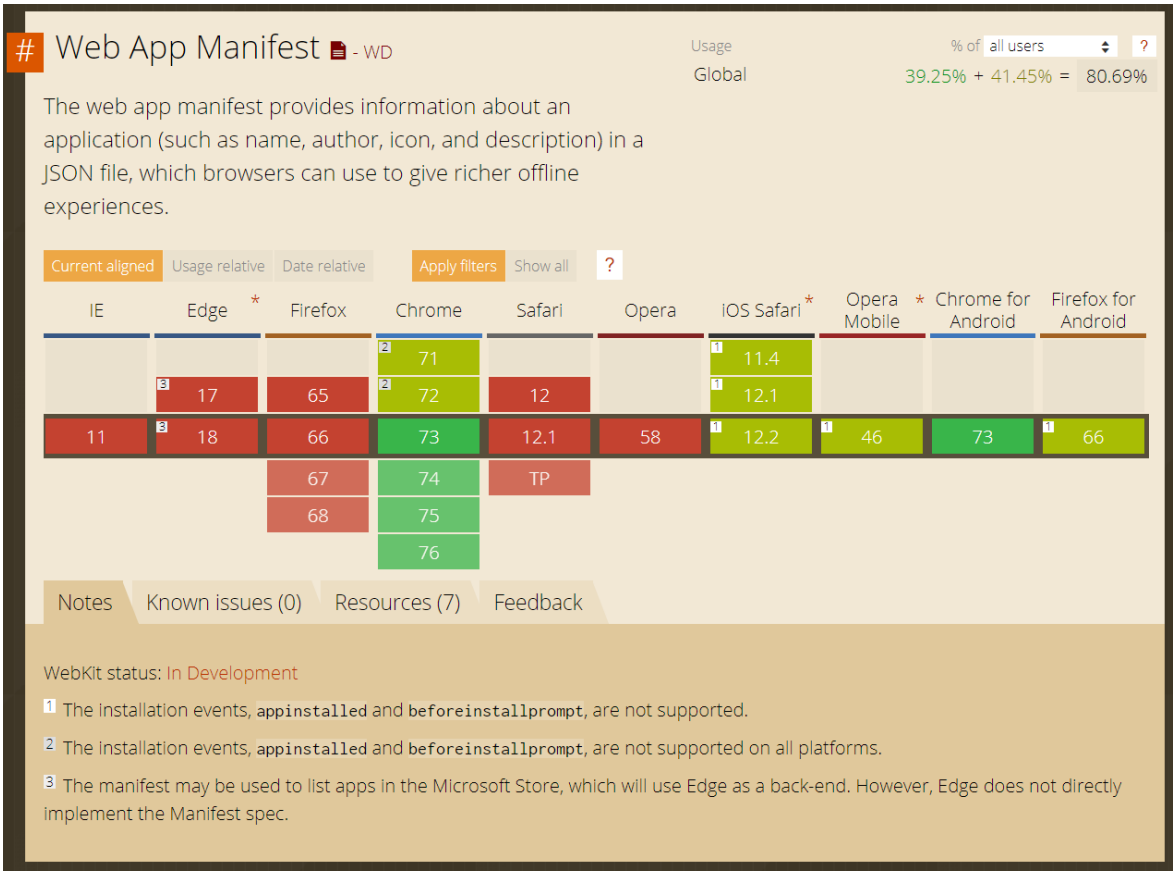
Webový aplikační manifest je jednoduchý JSON soubor, který obsahuje informace webové aplikace a podává je prohlížeči, aby věděl, jak se má aplikace chovat, pokud dojde k jeho stažení. Použití manifestu je důležité k zobrazení výzvy ke stažení webové aplikace na domovskou plochu uživatele, která se následně chová jako nativní aplikace (Gaunt, et al., 2019).

Výpis 1 Příklad manifestačního souboru (autor)

```
1  {
2    "name": "SoundForYou",
3    "short_name": "SoundForYou",
4    "display": "standalone",
5    "start_url": "./index.html",
6    "theme_color": "#4da6ff",
7    "background_color": "#4da6ff",
8    "icons": [
14      {
15        "src": "images/icons/icon-192x192.png",
16        "sizes": "192x192",
17        "type": "image/png"
18      },
19      {
20        "src": "images/icons/icon-512x512.png",
21        "sizes": "512x512",
22        "type": "image/png"
23      }
24    ]
25  }
```

Výpis 1 obsahuje ukázkový soubor manifest v datovém formátu JSON. `Name` je název aplikace, který se zobrazí na uvítací obrazovce po spuštění aplikace, `short_name` je zkrácený název

aplikace, který se zobrazuje pod ikonou na domovské obrazovce zařízení uživatele. `Display` určuje, jak se má aplikace pro uživatele zobrazit. Hodnota `standalone` zobrazuje aplikaci ve vlastním okně, odděleně od prohlížeče a skrývá UI elementy prohlížeče jako je adresní řádek. `Start_url` definuje prohlížeči, jakou stránku by měl zobrazit po spuštění aplikace. `Theme_color` nastavuje barvu UI elementů jako je horní panel prohlížeče, `background_color` definuje barvu pozadí, která se zobrazí na uvítací obrazovce po spuštění aplikace. `Icons` je sada objektů předem definovaných ikon obsahující vlastnosti `src`, `sizes` a `type`, které se použijí v případě uložení webové aplikace na domovskou obrazovku podle velikosti rozlišení zařízení uživatele.



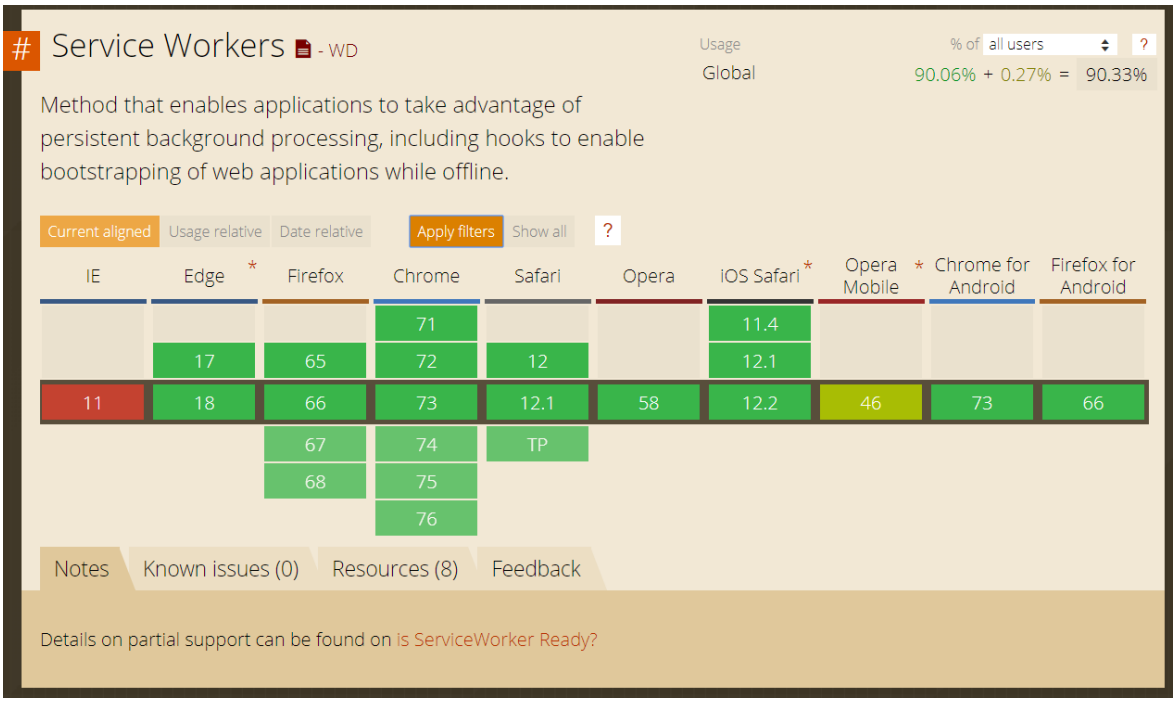
Obrázek 2 Podpora manifestu v prohlížečích (Deveria, 2019)

Obrázek 2 ukazuje, že je manifest plně podporován na prohlížeči Chrome pro desktopy a Chrome pro OS Android od verze 73. Mobilní verze prohlížeče Safari pro OS iOS, Opery a Firefoxu pro OS Android nepodporují některé instalační události, kdežto prohlížeče Internet Explorer, Edge, Mozilla Firefox, Safari a Opera pro desktopy manifest vůbec nepodporují.

2.2.2 Service Workers

Service worker je javascriptový kód, který v prohlížeči běží v pozadí, odděleně od webové stránky, obsahující funkce, které nepotřebují webovou stránku nebo uživatelskou interakci, např. *push* notifikace a synchronizace na pozadí. *Service worker* leží mezi aplikací a sítí a rozhoduje, kdy má získat obsah z mezipaměti a kdy z internetu. Pouze aplikace zabezpečené pomocí protokolu HTTPS mohou registrovat *service worker*, kvůli bezpečnosti přenosu dat

mezi uživatelem a serverem a k obraně proti hrozbám, např. *man-in-the-middle* útokům (Gaunt, 2019).



Obrázek 3 Podpora service worker v prohlížečích (Deveria, 2019)

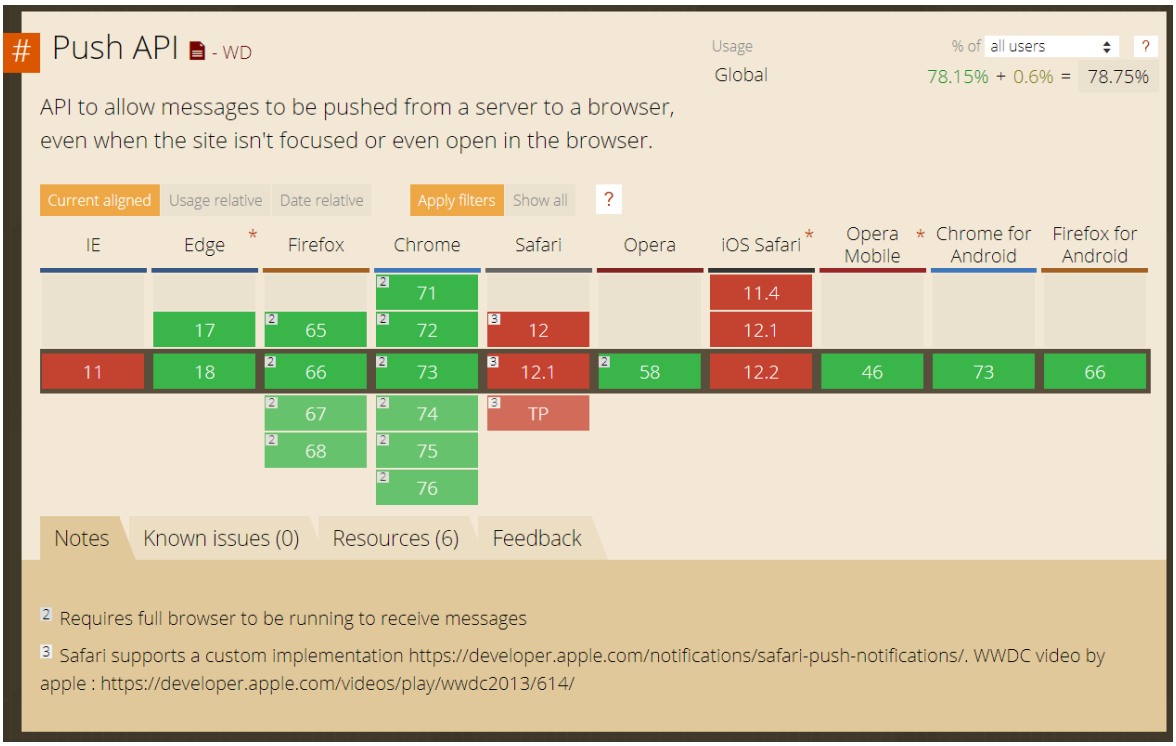
Obrázek 3 ukazuje, že *service worker* podporuje většina nejvyužívanějších prohlížečů jako je Chrome, Mozilla Firefox a Safari, v desktopové i v mobilní verzi. Prohlížeč Internet Explorer, stejně jako v případě manifestu, tuto komponentu nepodporuje.

2.2.3 Aplikační shell

Aplikační *shell* je minimální HTML, CSS a JavaScript potřebný ke správnému běhu uživatelského rozhraní progresivní webové aplikace. Aplikační *shell*, definovaný uživatelem, je uložený do mezipaměti ihned při první načtení webové aplikace pomocí *service worker.* Napomáhá ke spolehlivému a okamžitému načtení webové aplikace při opakované návštěvě. To znamená, že dynamický obsah se načítá z internetu, kdežto kostra aplikace se načte z mezipaměti při každé další návštěvě (Osmani, 2019).

2.2.4 Web push notifikace

Push notifikace je textová zpráva, která se zobrazí uživateli např. na stavovém řádku mobilního zařízení, pokud dojde ke spuštění události. Události může být několik, např. pokud uživatel zapne notifikaci pro oznámení o dostupnosti konkrétního zboží na internetovém obchodu, dojde ke zobrazení notifikace, pokud se dané zboží stane dostupné.



Obrázek 4 Podpora push notifikací v prohlížečích (Deveria, 2019)

Obrázek 4 ukazuje, že desktopové i mobilní verze prohlížečů Chrome, Mozilla Firefox a Opery tuto funkci plně podporují, naopak Internet Explorer a mobilní prohlížeč Safari ji vůbec nepodporují. Desktopová verze prohlížeče Safari nepodporuje klasickou webovou *push* notifikaci, má vlastní způsob implementace *push* notifikace s názvem *Apple Push Notification service* (APNs) vytvořenou společností Apple (Apple, 2018).

3 Použitá metodika pro návrh a tvorbu webové aplikace

Pro návrh a tvorbu webové aplikace byla využita níže uvedená metodika.

MMSP

Metodika pro Menší Softwarové Projekty (MMSP), která byla tématem diplomové práce Ing. Petry Rejnkové (Rejnková, 2011) a je možné k ní přistupovat z webové adresy https://spicenter.vse.cz/ Vysoké školy ekonomické v Praze, byla vytvořena lokalizací a přizpůsobením agilní metodiky OpenUP verze 1.5.1.1 a sloužila především pro projekty v rámci výuky softwarového inženýrství na Vysoké škole ekonomické v Praze. Může být využívána i na reálných projektech v praxi, je rozšířitelná a lze ji upravit podle unikátních požadavků konkrétních projektů.

Metodika je vhodná především pro tým o maximálně deseti členech a s projektem v délce do šesti měsíců, který není extra kritický. Je založena na myšlence co nejrychlejšího vývoje softwaru, který je předložen zákazníkovi a na základě zpětné vazby ho dále iterativně vylepšovat. Obsahuje popis rolí, disciplín, úloh, pracovních produktů a životního cyklu vývoje softwaru. Metodika se zaměřuje primárně na vývoj, provoz a údržba softwaru jsou v metodice zmíněny pouze okrajově.

Metodika MMSP je tvořena z prvků, které dohromady utvářejí proces tvorby softwaru. Tabulka 1 zobrazuje prvky metodiky, jejich popis a příklady. Mezi role se řadí *Projektový manažer*, *Analytik*, *Tester*, *Vývojář*, *Architekt*, *Zainteresovaná osoba* a *Nedefinovaná role*, kdy u menších projektů může člen týmu zastupovat více rolí.

Metodika definuje disciplíny následovně:

- Řízení projektu
- Řízení změn a konfigurací
- Požadavky
- Architektura
- Vývoj
- Testování

Metodika dělí životní cyklus produktu do fází (fáze *Zahájení*, *Rozpracování*, *Konstrukce* a *Zavedení*), iterací, aktivit a úloh. Aktivity jsou tvořeny řadou úloh, které jsou obsaženy v iteraci každé fáze, každá fáze může být tvořena jednou nebo větším počtem iterací a končí dosažením milníku. Testování se provádí opakovaně v rámci každé iterace, když je řešení modifikováno (Rejnková, 2011).

Jelikož tým tvoří pouze jeden člověk, v práci zastupuje většinu rolí a některé disciplíny jako jsou řízení projektu a řízení změn a konfigurací nejsou v práci popsány. Dále byly podle šablony metodiky MMSP stanoveny a definovány některé pracovní produkty úloh pouze z disciplíny *Požadavky*, ze kterého kapitola 6 vychází.

Tabulka 1 Prvky metodiky MMSP (Rejnková, 2011)

Název	**Popis**	**Příklady**
Návod	Šablony, příklady, rady a další pomůcky, které je možné při procesu vývoje využít	Požadavky – šablona, Vize – šablona
Pracovní produkt	To, co je v rámci procesu vývoje vytvářeno, modifikováno či používáno specifickou rolí	Požadavky, Případ užití, Vize
Proces	Popis struktury práce a workflow činností, který spojuje dohromady úlohy, role a pracovní produky	Životní cyklus, Testování
Role	Charakteristika člena týmu, který provádí v rámci procesu vývoje jednotlivé úlohy a vytváří požadované pracovní produkty. Role definují schopnosti a zodpovědnosti, ne konkrétní osoby.	Analytik, Vývojář, Tester
Úloha	Popis, jak pracovat k dosažení stanovených cílů nebo k vytvoření určitých pracovních produktů. Úlohy jsou vykonávány rolemi a jsou obvykle definovány jako série dílčích úkolů.	Vytvoření vize projektu

4 Použité technologie a nástroje pro návrh a tvorbu webové aplikace

Pro návrh a tvorbu webové aplikace a její testování byly využity níže uvedené webové technologie a nástroje.

CSS

CSS (*Cascading Style Sheet*) je zkratka pro kaskádové styly, které umožňují stylizaci a grafickou úpravu webových stránek. Jazyk vznikl kolem roku 1994, kdy jeho koncept publikoval Håkon Wium Lie. Kaskádové znamená, že se na sebe mohou vrstvit definice stylů, přestože platí pouze ta poslední. Jedním z důvodů tvorby jazyka CSS byla v přehlednosti kódu HTML dokumentu, tzv. oddělit obsah od grafické stránky (Wikipedie, 2019).

HTML

HTML (*Hypertext Markup Language*) je značkovací jazyk využívaný pro tvorbu webových stránek. Jazyk HTML je charakterizován značky (tzv. tagy) a jejich vlastností (tzv. atributů). Základní kostra HTML dokumentu obsahuje značky `<!DOCTYPE html>`, `<html>`, `<head>`, `<body>`. Pro psaní HTML kódu postačí klasický poznámkový blok společnosti Microsoft, ale existují editory kódů, které zpřehledňují kódy a usnadňují práci (Janovský, 2017).

JavaScript

JavaScript je multiplatformní, objektově orientovaný, událostmi řízený, klientský skriptovací jazyk. Za jeho vznikem stojí Brendan Eich z tehdejší společnosti Netscape v roce 1995. Veškerý javascriptový kód se vykonává na straně klienta. Nejnovější verze je ECMAScript 6, kterou normuje nezisková organizace ECMA International (Wikipedie, 2019).

jQuery

jQuery je javascriptová knihovna, které místo zdlouhavého javascriptového kódu definuje kratší kód a tím šetří čas a velikost souboru. Byl vytvořen Johnem Resigem v roce 2006 a nyní je pod licencí univerzity MIT. Mezi jeho výhody patří například jednodušší manipulace s DOM elementy a volání API (Wikipedie, 2019).

Lighthouse

Lighthouse je open-source, automatizovaný nástroj pro testování kvality webových stránek společnosti Google. Obsahuje detailní kontrolu výkonu, přístupnosti, PWA, *best practices* a SEO webových stránek. Nástroj je implementován v prohlížeči Chrome (Google Developers, 2018).

Sublime Text

Sublime Text je multiplatformní kódový editor. Podporuje většinu programových a značkovacích jazyků, je dále možno rozšířit jeho funkcionalitu pomocí pluginů vytvářených jeho komunitou. Nejnovější verze nese název Sublime Text 3 (Sublime Text, 2019).

Web Server for Chrome

Web Server for Chrome je open source plugin pod licencí univerzity MIT pro desktopový prohlížeč Chrome, jehož funkcí je tvorba lokální serveru. Je vhodný zejména pro vytváření a testování webových stránek a aplikací i bez připojení k internetu (Google, 2017).

5 Průzkum trhu podobných aplikací

Před samotným definováním požadavků, návrhem a implementací aplikace byl proveden průzkum trhu aplikací, které se něčím podobají vytvářené aplikace v práci. Aplikace je podobná všem hudebním aplikacím v tom, že umožňují vyhledávat hudbu. Níže uvádím některé nativní mobilní aplikace.

Free Music

Aplikace je dostupná zdarma z obchodu Google Play platformy Android. Tato aplikace umožňuje vyhledávat hudbu z hudební platformy SoundCloud. Dalšími funkcemi jsou výběr skladeb pomocí žánrů, vytvoření si vlastního seznamu skladeb a uložení oblíbených skladeb. Také slouží jako přehrávač lokálních skladeb a je možné aplikaci umístit na domovskou obrazovku mobilního zařízení jako *widget*.

Free Music: Unlimited for YouTube Stream Player

Aplikace je zdarma ke stažení z obchodu Google Play platformy Android. Tato aplikace slouží jako vyhledávač hudby z hudební platformy YouTube. Umožňuje výběr hudby pomocí žánrů nebo také zobrazit týdenní nové skladby a nejlepší skladby podle zemí. Aplikace slouží také jako rádio přehrávač a má implementovanou možnost vytvoření vlastního seznamu skladeb.

Unison Music Player

Tato aplikace je zdarma ke stažení z obchodu Google Play platformy Android. Aplikace umožňuje vyhledávat hudbu z hudební platformy Spotify. Obsahuje možnost zobrazení vlastního seznamu hudeb vytvořený v hudební platformě Spotify, zobrazení současného trendu nebo uložení oblíbených hudeb. Aplikace slouží také jako přehrávač lokálních skladeb.

YouTube Music

Tato aplikace je dostupná ke stažení zdarma z obchodu App Store platformy iOS. Aplikace umožňuje vyhledávání hudby z hudební platformy YouTube. Obsahuje výběr hudby podle současného trendu a také zobrazuje doporučenou hudbu podle vkusu, lokace nebo času. Po získání prémiového účtu lze uložit hudbu pro offline poslouchání.

6 Specifikace požadavků na aplikaci podle MMSP

Touto kapitolou začíná praktická část práce. Specifikace požadavků byly vytvořena v rámci disciplíny *Požadavky* podle šablony metodiky MMSP, podle kterého je tato kapitola zpracována. Mezi pracovní produkty disciplíny *Požadavky* byly definovány dokumenty *Vymezení vize*, *Požadavky* a *Případy užití*. Pro definování dokumentů podle šablon byly dále vybrány pouze sekce, které bylo možné specifikovat.

6.1 Vymezení vize

6.1.1 Vymezení problému

Tabulka 2 Vymezení problému (autor)

Definice problému	Aplikace je vytvořena za účelem zjednodušení hledání hudby na hudebních platformách YouTube a SoundCloud.
Koho problém ovlivňuje	Uživatele, kteří využívají hudební platformy YouTube a SoundCloud k hledání hudeb.
Dopady problému	Uživatelé hledající hudbu na více aplikacích mohou zatěžovat systém zařízení, ztrácejí čas proklikáváním aplikací.
Přínosy úspěšného řešení problému	Zlepšení zážitku z hledání hudby, snížení počtu otevřených oken aplikací a tím snížení zatěžování systému.

6.1.2 Vymezení produktu

Tabulka 3 Vymezení produktu (autor)

Název vyvíjeného IS/ICT	SoundForYou
Pro	Kteréhokoliv uživatele zařízení podporující webový prohlížeč.
Který	Zobrazí uživateli požadovanou hudbu současně z platforem YouTube a SoundCloud.

6.1.3 Popis všech zúčastněných subjektů

Tabulka 4 Popis všech zúčastněných subjektů (autor)

Název	**Popis**	**Odpovědnost**
Uživatel	Uživatel webové aplikace	Uživatel má přístup ke všem funkcím aplikace.

6.1.4 Uživatelské rozhraní

Uživatelské rozhraní nabídne podobné rozvržení elementů aplikace a také jednoduchost a přístupnost, jak jsou uživatelé nativních mobilních aplikací zvyklí. Nabídne také možnost nastavení vzhledu aplikace a spuštění aplikace přes ikonku na domovské obrazovce jako u nativní aplikace pro prohlížeče, které podporují přístup Progressive Web Apps.

6.1.5 Požadavky a funkce

Tabulka 5 Požadavky a funkce (autor)

Požadavek (potřeba)	**Priorita**	**Podrobný popis**	**Plánované spuštění**
Hudba	1	Vyhledat hudbu, kterou uživatel zadal do vyhledávacího pole	-
Ikonka	1	Možnost spuštění aplikace přes ikonku a bez horního panelu prohlížeče	-
Přehrávač hudby	1	Spustit přehrávač s obsahem hudby, na který uživatel kliknul	-
Menu	2	Možnost zobrazit obsah sekcí v menu	-
Notifikace	2	Zobrazit notifikaci o výpadku připojení k internetu	-
Vzhled	2	Možnost změny vzhledu aplikace	-
Offline režim	3	Možnost spustit aplikaci i bez připojení k internetu	-

6.2 Požadavky

6.2.1 Funkční požadavky

Tabulka 6 Funkční požadavky (autor)

ID požadavku	Název požadavku	Podrobný popis	Priorita (1… nejvyšší priorita, 5…nejnižší priorita)	Odkaz na případ užití
F001	Přehrání hudby	Uživatel může přehrát hudbu.	1	UC1
F002	Přidání ikonky	Uživatel může přidat ikonku k aplikaci na domovskou obrazovku.	1	UC2
F003	Vyhledání hudeb	Uživatel má možnost vyhledat hudbu.	1	UC3
F004	Nastavení vzhledu	Uživatel může nastavit vzhled aplikace na světlé nebo tmavé téma.	2	UC4
F005	Zobrazení informací	Uživatel může zobrazit informace o aplikaci	2	UC5
F006	Zobrazení notifikace	Uživateli se zobrazí notifikaci při ztrátě připojení k internetu.	2	UC6
F007	Zobrazení obsahu sekcí	Uživatel může zobrazit obsah jednotlivých sekcí v menu	2	UC7
F008	Napsání e-mailu	Uživatel může napsat autorovi reakci ohledně aplikace	3	UC8
F009	Spuštění v offline režimu	Uživatel může spustit aplikaci v offline režimu.	3	UC9

6.2.2 Nefunkční požadavky

6.2.2.1 Použitelnost

Tabulka 7 Použitelnost (autor)

ID požadavku	Název požadavku	Podrobný popis	Priorita (1… nejvyšší priorita, 5…nejnižší priorita)
POU001	Jazyk	Aplikace bude v českém jazyce	1
POU002	Jednoduchost	Aplikace bude využívat jednoduchý design a musí být srozumitelná i bez návodu	1
POU003	Přístup PWA	Webový prohlížeč bude podporovat přístup PWA	1
POU004	Webový prohlížeč	Ke spuštění aplikace je zapotřebí jakýkoliv webový prohlížeč	1

6.2.2.2 Spolehlivost

Tabulka 8 Spolehlivost (autor)

ID požadavku	Název požadavku	Podrobný popis	Priorita (1… nejvyšší priorita, 5…nejnižší priorita)
SPO001	Dostupnost	Aplikace musí být po prvním spuštění aplikace za jakéhokoliv stavu připojení k internetu spustitelná	1
SPO001	Informativnost	Aplikace musí upozornit uživatele v případě ztráty připojení k internetu	1

6.2.2.3 Výkon

Tabulka 9 Výkon (autor)

ID požadavku	Název požadavku	Podrobný popis	Priorita (1… nejvyšší priorita, 5…nejnižší priorita)
VYK001	Rychlost	Aplikace musí načítat hudbu v řádech několika sekund	1
VYK002	Spotřeba	Aplikace spotřebovává data podle využití uživatele	2

6.2.3 Rozhraní

6.2.3.1 Uživatelské rozhraní

Webová aplikace bude vypadat a chovat se po instalaci jako nativní aplikace operačního systému Android ve smyslu rozvržení elementů aplikace a vlastnostech jednoduchosti a přehlednosti.

6.2.3.2 Vzhled

Tabulka 10 Vzhled (autor)

ID požadavku	Název požadavku	Podrobný popis	Priorita (1… nejvyšší priorita, 5…nejnižší priorita)
VZH001	Ikonka	Aplikace bude mít svoji vlastní ikonku	1
VZH002	Logo	Aplikace bude mít své vlastní logo, které bude viditelné v aplikaci	1
VZH003	Barva	Aplikace bude mít příjemné barvy a vzhled pro denní a noční režim	2
VZH004	Uvítací obrazovka	Aplikace zobrazí uvítací obrazovku po spuštění aplikace přes ikonku	2

6.2.3.3 Rozložení a navigace

Tabulka 11 Rozložení a navigace (autor)

ID požadavku	Název požadavku	Podrobný popis	Priorita (1… nejvyšší priorita, 5…nejnižší priorita)
ROZ001	Hlavní obrazovka	Hlavní obrazovka bude zobrazovat vyhledávací funkci aplikace a obsahovat menu	1
ROZ002	Menu	Menu se nachází na pravém horním rohu a obsahuje tři tlačítka: O aplikaci, Kontakt a Nastavení	1
ROZ003	Sekce v menu	Každá sekce se po rozkliknutí zobrazí na hlavní obrazovce	1

6.2.3.4 Konzistence

Tabulka 12 Konzistence (autor)

ID požadavku	Název požadavku	Podrobný popis	Priorita (1… nejvyšší priorita, 5…nejnižší priorita)
KON001	Vzhled	Uživatel se při ovládání aplikace musí cítit jako v nativní aplikaci operačního systému Android	1

6.2.4 Personalizace a customizace

Tabulka 13 Personalizace a customizace (autor)

ID požadavku	Název požadavku	Podrobný popis	Priorita (1… nejvyšší priorita, 5…nejnižší priorita)
PER001	Téma	Uživatel si může nastavit světlé nebo tmavé téma.	1

6.2.5 Obchodní pravidla

6.2.5.1 Čerpání dat

Pokud uživatel hledá a poslouchá více hudeb delší dobu, pak dojde k větší spotřebě dat a naopak. Toto pravidlo se vztahuje zejména pro mobilní data.

6.2.6 Systémová omezení

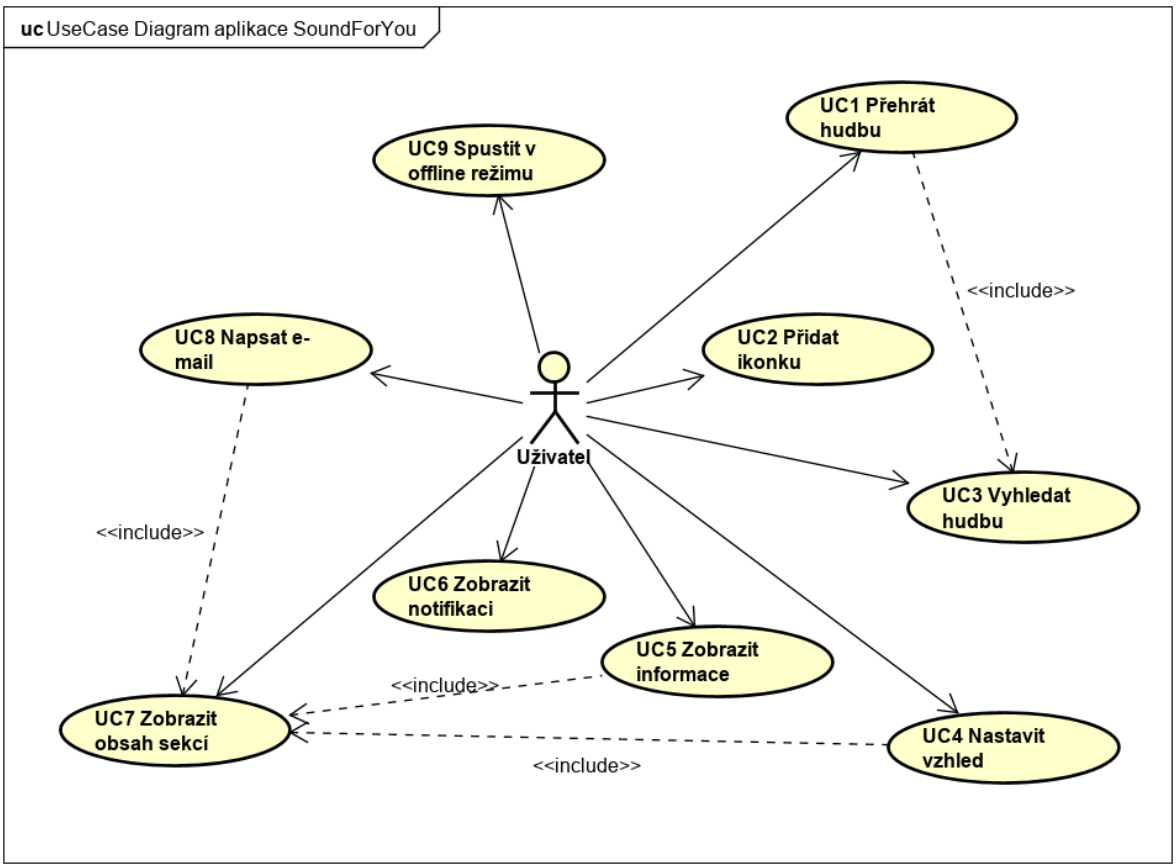
Tabulka 14 Systémová omezení (autor)

ID požadavku	Název požadavku	Podrobný popis	Priorita (1… nejvyšší priorita, 5…nejnižší priorita)
SYS001	Přístup PWA	Aplikace musí podporovat přístup PWA	1
SYS002	Webový prohlížeč	Pro spuštění aplikace je nutný webový prohlížeč.	1

6.3 Případy užití

6.3.1 Diagram případů užití

Diagram případu užití byl vytvořen pomocí aplikace Astah UML, který je dostupný ke stažení na adrese http://astah.net/.



Obrázek 5 Diagram případů užití (autor)

6.3.2 Definice aktérů

Tabulka 15 Definice aktérů (autor)

Název aktéra	Popis aktéra
Uživatel	Uživatel je osoba, která má přístup ke všem funkcím aplikace. Jediný aktér aplikace.

6.3.3 Specifikace případu užití

Specifikace případu užití proběhlo pouze pro ty případy, které mají nejvyšší prioritu. Ostatní případy užití si lze domyslet.

Tabulka 16 Specifikace případu užití přehrání hudby (autor)

<table>
<tr><th>Název případu užití</th><td>Přehrání hudby</td></tr>
<tr><th>Identifikace případu užití</th><td>UC1</td></tr>
<tr><th>Identifikace funkčního požadavku</th><td>F001</td></tr>
<tr><th>Cíl případu užití</th><td>Přehrát hudbu</td></tr>
<tr><th>Primární aktér(ři)</th><td>Uživatel</td></tr>
<tr><th>Pomocný aktér(ři)</th><td>-</td></tr>
<tr><th>Vstupní podmínky</th><td>Uživatel klikne na tlačítko spuštění požadované hudby</td></tr>
<tr><th>Výstupní podmínky</th><td>Uživateli se zobrazí přehrávač s požadovanou hudbou</td></tr>
<tr><th>Základní scénář</th><td>
<table>
<tr><th>Krok</th><th>Role</th><th>Akce</th></tr>
<tr><td>1</td><td>Uživatel</td><td>Uživatel zadá textový řetězec a stiskne tlačítko pro vyhledávaní</td></tr>
<tr><td>2</td><td>Systém</td><td>Systém vygeneruje seznam hudeb pro hudební platformy YouTube a SoundCloud</td></tr>
<tr><td>3</td><td>Uživatel</td><td>Uživatel klikne na tlačítko pro spuštění požadované hudby</td></tr>
<tr><td>4</td><td>Systém</td><td>Systém načte přehrávač pro požadovanou hudbu</td></tr>
</table>
</td></tr>
<tr><th>Alternativní scénář</th><td>
<table>
<tr><th>Krok</th><th>Role</th><th>Akce</th></tr>
<tr><td>2 a 4</td><td>Systém</td><td>Systém nenačte požadovanou hudbu z důvodu výpadku připojení k internetu</td></tr>
<tr><td>2 a 4</td><td>Systém</td><td>Systém zobrazí notifikaci z důvodu výpadku připojení k internetu</td></tr>
</table>
</td></tr>
</table>

Tabulka 17 Specifikace případu užití přidání ikonky (autor)

<table>
<tr><th>Název případu užití</th><td>Přidání ikonky</td></tr>
<tr><th>Identifikace případu užití</th><td>UC2</td></tr>
<tr><th>Identifikace funkčního požadavku</th><td>F002</td></tr>
<tr><th>Cíl případu užití</th><td>Přidat ikonku</td></tr>
<tr><th>Primární aktér(ři)</th><td>Uživatel</td></tr>
<tr><th>Pomocný aktér(ři)</th><td>-</td></tr>
<tr><th>Vstupní podmínky</th><td>Uživatel stáhne aplikaci</td></tr>
<tr><th>Výstupní podmínky</th><td>Uživateli se zobrazí ikonka na domovské obrazovce</td></tr>
<tr><th>Základní scénář</th><td>
<table>
<tr><th>Krok</th><th>Role</th><th>Akce</th></tr>
<tr><td>1</td><td>Uživatel</td><td>Uživatel zadá adresu aplikace na webovém prohlížeči</td></tr>
<tr><td>2</td><td>Systém</td><td>Systém zobrazí aplikaci na webovém prohlížeči</td></tr>
<tr><td>3</td><td>Systém</td><td>Systém umožní přidat aplikaci na domovskou obrazovku</td></tr>
<tr><td>4</td><td>Uživatel</td><td>Uživatel potvrdí přidání aplikace na domovskou obrazovku</td></tr>
<tr><td>5</td><td>Systém</td><td>Systém přidá ikonku k aplikaci na domovskou obrazovku</td></tr>
</table>
</td></tr>
<tr><th>Alternativní scénář</th><td>
<table>
<tr><th>Krok</th><th>Role</th><th>Akce</th></tr>
<tr><td>2</td><td>Systém</td><td>Systém nezobrazí aplikaci na prohlížeči z důvodu ztráty připojení k internetu</td></tr>
<tr><td>3</td><td>Systém</td><td>Systém neumožní přidat aplikaci na domovskou obrazovku, protože prohlížeč tuto funkci nepodporuje.</td></tr>
</table>
</td></tr>
</table>

Tabulka 18 Specifikace případu užití vyhledání hudby (autor)

<table>
<tr><th>Název případu užití</th><td>Vyhledání hudby</td></tr>
<tr><th>Identifikace případu užití</th><td>UC3</td></tr>
<tr><th>Identifikace funkčního požadavku</th><td>F003</td></tr>
<tr><th>Cíl případu užití</th><td>Vyhledat hudbu</td></tr>
<tr><th>Primární aktér(ři)</th><td>Uživatel</td></tr>
<tr><th>Pomocný aktér(ři)</th><td>-</td></tr>
<tr><th>Vstupní podmínky</th><td>Uživatel zadá textový řetězec do vyhledávacího pole a stiskne tlačítko pro vyhledávaní nebo klávesu Enter</td></tr>
<tr><th>Výstupní podmínky</th><td>Uživateli se zobrazí požadovaný seznam hudeb</td></tr>
<tr><th>Základní scénář</th><td>
<table>
<tr><th>Krok</th><th>Role</th><th>Akce</th></tr>
<tr><td>1</td><td>Uživatel</td><td>Uživatel zadá do vyhledávacího pole textový řetězec</td></tr>
<tr><td>2</td><td>Uživatel</td><td>Uživatel stiskne tlačítko pro vyhledávaní nebo klávesu Enter</td></tr>
<tr><td>3</td><td>Systém</td><td>Systém načte požadovaný seznam hudeb pro hudební platformy YouTube a SoundCloud</td></tr>
</table>
</td></tr>
<tr><th>Alternativní scénář</th><td>
<table>
<tr><th>Krok</th><th>Role</th><th>Akce</th></tr>
<tr><td>1 a 2</td><td>Systém</td><td>Systém zobrazí notifikaci z důvodu výpadku připojení k internetu</td></tr>
<tr><td>3</td><td>Systém</td><td>Systém nenačte požadovanou hudbu z důvodu výpadku připojení k internetu</td></tr>
</table>
</td></tr>
</table>

7 Návrh webové aplikace

Tato kapitola se zabývá definováním návrhu a požadavků aplikace.

Pro nově vytvářenou aplikaci jsem vybral aplikaci s názvem SoundForYou, která zobrazuje hudbu z hudebních platforem YouTube a SoundCloud. Aplikace je cílena především pro uživatele, kteří využívají obě varianty hudebních platforem YouTube a SoundCloud k poslechu hudby. K vytvoření aplikace byl vytvořen *wireframe*, neboli drátový model, aplikace. Webová aplikace je vytvořena pomocí responzivního přístupu *mobile first*. Aplikace načítá data z platforem YouTube a SoundCloud pomocí API volání. Aplikace je vytvářena a testována na lokálním serveru prohlížeče Chrome pomocí nástroje Web Server for Chrome.

7.1 Zadání

Cílem této práce je vytvoření webové aplikace pro vyhledávání hudeb současně na hudebních platformách YouTube a SoundCloud a s přístupem PWA.

7.2 Specifikace požadavků

7.2.1 Obecné požadavky

Aplikace je určená pro jakékoliv zařízení minimálně podporující webový prohlížeč a přinejlepším přístup Progressive Web Apps. Aplikace musí být přehledná, tzn. srozumitelná pro svůj účel i bez návodu a nesmí využívat mnoho ovládacích prvků, aby byla přístupná i nezkušenému uživateli. Aplikace musí být dostupná v českém jazyce a mít možnost změny vzhledu aplikace.

7.2.2 Detailní požadavky

Po spuštění aplikace na webovém prohlížeči se uživateli zobrazí, zda si přeje povolit notifikace. Poté bude mít uživatel možnost přidat aplikaci na domovskou obrazovku, ať už tím, že se zobrazí vyskakovací okno nebo pomocí tlačítka na prohlížeči pro možnost přidání aplikace na domovskou obrazovku. Úvodní stránka obsahuje logo, menu, název, titulek, vyhledávací pole a tlačítko na vyhledávání a patičku aplikace. Po zadání textového řetězce a následném stisknutí tlačítka pro vyhledávání nebo klávesy Enter se uživateli zobrazí podle textového řetězce požadovaná hudba pro hudební platformy YouTube a SoundCloud. Pro spuštění hudby klikne uživatel na tlačítko požadované hudby. Pokud se uživatel pokusí vyhledat aplikaci v režimu bez připojení k internetu, spustí se notifikace o oznámení stavu bez připojení k internetu. Pokud se uživatel dostane k aplikaci přes ikonku po přidání aplikace na domovskou obrazovku, zobrazí se mu uvítací obrazovka a okno aplikace bez horního panelu prohlížeče a následně jsou kroky pro použití aplikace podobné jako při použití aplikace v prohlížeči.

7.3 Definování cílové skupiny

Cílová skupina uživatelů je definovaná následovně:

- Mají rádi hudbu.
- Využívají alespoň jednu z hudebních platforem YouTube nebo SoundCloud, v nejlepším případě obě zároveň.
- Používají hudební platformu YouTube pouze k vyhledání hudby.
- Neradi stahují aplikace z různých internetových obchodů.

7.4 Tvorba loga a ikonek

K tvorbě loga aplikace byla použita aplikace Adobe Illustrator CC. Výhodou aplikace je, že umožňuje ukládat obrázek ve více formátech. Pro logo i ikonky byl využit formát PNG. Ikonek bylo vytvořeno celkem osm z toho důvodu, aby byly aplikovatelné na zařízení s různými rozlišeními pomocí manifestu. Výsledné logo a ikonky jsou zobrazeny v Příloze A.

7.5 Drátový model aplikace

Před samotným kódováním webové aplikace je doporučené si vytvořit drátový model, který poskytne představu o tom, jak bude vypadat aplikace po grafické stránce. Drátový model lze nakreslit ručně nebo pomocí různých nástrojů. Drátový model byl vytvořen pomocí online nástroje Wireframe.cc, který je dostupný na adrese https://wireframe.cc/. Drátový model pro mobilní a desktopové zařízení je obsažen v Příloze B.

7.6 Minimalistický design aplikace

Aplikace využívá minimalistického přístupu designu kostry aplikace, kdy se snaží nevyužívat žádné rušivé elementy a používá jednoduché barvy, převážně v černé, bílé a v jejich odstínech. Je využita také modrá barva, která je považována za barvu klidu, uspokojení a mající antistresové účinky. Tyto vlastnosti jsou charakteristické při poslechu hudby. Dalšími barevnými prvky aplikace jsou ikonky a videa.

7.7 Responzivní přístup ke tvorbě aplikace

Existují dva přístupy ke tvorbě responzivních stránek, přístup *mobile first* a *desktop first*. U přístupu *mobile first* se prvně navrhuje stránka přizpůsobená mobilním zařízením a následně se aplikují styly pro počítače. Druhou možností je metoda desktop first, kdy se optimalizují stránky nejprve pro počítače a různé zařízení s monitory vysokého rozlišení. Aplikování stylů pro jiné rozlišení se provádí pomocí mediálních dotazů. Jelikož je webová aplikace zaměřená na mobilní zařízení, byl využit přístup *mobile first*.

8 Implementace aplikace

V této praktické části práce je popsána tvorba aplikace podle definovaných požadavků a návrhu z předchozích kapitol.

8.1 Zdrojový kód aplikace

8.1.1 Struktura souborů

Všechny soubory se nacházejí v *rootu*, tzv. kořenu aplikace. Složky aplikace jsou sestaveny následovně:

- assets
- css
- js

V složce *assets* se nacházejí všechny ikony a obrázky použité v aplikaci. Ve složce *css* se nachází veškeré použité kaskádové styly a složka *js* obsahuje hlavní javaskriptový soubor *main.js* a vedlejší javascriptové soubory *menu-content.js* a *push-notification.js*. V *rootu* se nachází HTML soubory, manifestační soubor a soubor *service worker*.

8.1.2 Základní struktura HTML souborů

Hlavní HTML soubor, *index.html*, je vytvořen tak, aby obsahoval minimální kód. Obsah stránky je pak dynamicky generována pomocí javaskriptu. Soubor využívá konvenci značkovacího jazyka HTML5, v prvním řádku obsahuje deklaraci typu dokumentu pomocí kódu `<!DOCTYPE html>`. Veškerý kód souboru obaluje značka `<html lang="cs">`, kdy stanovení jazyka dokumentu pomocí atributu `lang` určuje jazyk webu a napomáhá překládání a kontrole pravopisu prohlížečů. Uvnitř značky `<html>` se nacházejí značky `<head>` a `<body>`. Uvnitř první značky je tag ke stanovení názvu aplikace `<title>`, meta tagy `<meta>` ke stanovení znakové sady, popisu, barvy panelu a oblasti rozvržení aplikace a tagy `<link>` k odkazování se na kaskádové styly a manifestačního souboru. Druhá značka definuje obsah aplikace. Vedlejší HTML soubor, *menu-content.html* obsahuje statické obsahy menu, každá obalená do vlastního tagu `<div>`, které se zobrazují na obrazovku pomocí javaskriptu.

8.1.3 Tvorba menu

Pro vytvoření obsahu menu je v HTML souboru *menu-content.html* definovány pomocí značky `<div>` obsah sekcí O aplikaci, Kontakt a Nastavení. Sekce O aplikaci a Kontakt obsahují převážně text, z tohoto důvodu zde nejsou popsány. Sekce Nastavení obsahuje nastavení vzhledu aplikace, konkrétněji vzhled se světlým nebo tmavým tématem.

```
<div class="container-static" id="settings">
    <h3>Vzhled aplikace</h3>
    <select id="colour">
        <option value="#fff" id="light">Světlé téma</option>
        <option value="#282828" id="dark">Tmavé téma</option>
    </select>
    <div class="btn-color">OK</div>
    <div class="container-close">x</div>
</div>
```

Obrázek 6 Vytvoření nastavení vzhledu aplikace (autor)

Obrázek 6 zobrazuje kód, který obsahuje tag `<select>` pro vytvoření rozbalovací nabídky, který obsahuje značky `<option>`, který definuje jednotlivé možnosti. Pro vzhled aplikace jsou definovány možnosti světlého nebo tmavého tématu. V souboru *main-content.js* je definována funkce, která po vybrání možnosti a potvrzení tlačítkem změní vzhled aplikace na vybrané téma. Veškerý obsah jednotlivých sekcí jsou dynamicky zobrazovány pomocí javaskriptu, který je obsažen v souboru *main-content.js*.

8.1.4 Aplikování stylů

Všechna stylizace proběhla v souboru *style.css* v složce *css*. Kaskádové styly byly vytvářeny ručně bez využití frameworků a s pomocí současného testování na mobilním a desktopovém zařízení. Následují vysvětlení zajímavých kódů ze souboru.

```
* {
    box-sizing: border-box;
    font-family: "Roboto", "Arial", sans-serif;
    text-align: center;
}
```

Obrázek 7 Definování stylů pro všechny elementy (autor)

Selektor `*` nastavuje styly pro všechny elementy obsažené v HTML souboru, dokud nedojde u některého elementu k přepsání daného stylu. V tomto případě se nastavuje `box-sizing` hodnoty `border-box`, díky kterému se do elementu počítá výška a šířka jeho případného rámečku, dále se pomocí `font-family` nastavují font písma, kdy se využije písmo `Arial`, pokud prohlížeč nepodporuje písmo `Roboto`, a nakonec se pro veškerý obsah elementů definuje jejich umístění na střed.

```
nav:hover .dropdown-content {
    display: block;
    border-top: 1px solid #fff;
    font-weight: bold;
}
```

Obrázek 8 Definování zobrazení obsahu menu (autor)

Obrázek 8 obsahuje styl pro zobrazení obsahu po rozkliknutí menu. Zobrazení probíhá pomocí selektoru `:hover`, který je především využíván pro zařízení s myší, kdy se definují styly pro situaci přejetí myší na daný element. Selektor `:hover` je u dotykových zařízení především využíván např. pro tvorbu rozevíracího menu, které bylo v práci využito.

```
html {
    position: relative;
    min-height: 100vh;
}

footer {
    position: absolute;
    bottom: 0;
    left: 0;
    width: 100vw;
    background-color: #e5e5e5;
}
```

Obrázek 9 Stylizace tagů html a footer (autor)

Selektorem `html` stanovuje `min-height` minimální výšku celé aplikace, v tomto případě se nastavuje minimální výška aplikace na celou výšku okna prohlížeče a `position` relativní pozicování, které společně s vlastností `min-height` při generování dynamického obsahu dynamicky změní minimální výšku celé aplikace. Pro patičku aplikace stanovuje `position` absolutní pozicování vzhledem k rodičovskému elementu, tj. elementu `html`, `bottom` a `left` s hodnotou nula stanoví, že se bude patička aplikace vždy zobrazovat na spodní straně aplikace.

```
.results, .results2{
    overflow: auto;
    white-space: nowrap;
    padding-bottom: 15px;
}

.thumbnail{
    width: 200px;
    display: inline-block;
    border: 1px solid #e5e5e5;
    box-shadow: 5px 10px 8px #888888;
    margin: 0 10px;
}

.heading {
    font-size: 16px;
    overflow: hidden;
    text-overflow: ellipsis;
}
```

Obrázek 10 Stylizace seznamů hudeb a jejich kontejnerů (autor)

`Display` hodnoty `inline-block` u kontejnerů jednotlivých hudeb, `overflow` hodnoty `auto` a `white-space` hodnoty `nowrap` u obalujícího elementu všech kontejnerů jednotlivých hudeb nastaví možnost rolování všech zobrazených videí vedle sebe. `Overflow` hodnoty `hidden` způsobí to, že pokud text přesáhne šířku svého rodičovského elementu, v daném místě se usekne. `Text-overflow` hodnoty `ellipsis` nastaví u useknutého textu tři tečky.

```
@media only screen and (min-width: 768px) {
```

Obrázek 11 Mediální dotaz pro tablety a větší displeje (autor)

Pro aplikaci jsou aplikovány odlišné styly pro zařízení větší než 768 pixelů, což je stanovená minimální šířka tabletů podle (Bootstrap, 2019). Obrázek 11 obsahuje mediální dotaz, jehož styly jsou definované pouze pro obrazovky s minimální šířkou 768 pixelů. V aplikaci je použit pouze tento mediální dotaz z důvodu menšího množství elementů, takže je aplikovaný jak pro tablety, tak i pro počítače a jiné zařízení s obrazovkou minimální šířky 768 pixelů.

8.1.5 Volání API

Hlavní funkcionality aplikace jsou zobrazení a přehrání hudby z daných hudebních platforem. Toho se docílilo voláním API z obou hudebních platforem, který je obsažen v hlavním javaskriptovém souboru *main.js* ve složce *js*.

8.1.5.1 YouTube Data API

YouTube Data API umožňuje implementovat množství funkcionality z YouTube do aplikace, jako třeba přehrávání videí a hledání obsahu, které byly pro aplikaci využity. Nevýhoda tohoto API je omezená kvóta pro připojení k API při základním nastavení pro vývojáře.

Prvně byl přidán skript tag Google API klientské knihovny pro JavaScript na konci značky `<body>` HTML souboru *index.html*, který umožňuje volání jeho metod. Zobrazení použitelných metod lze vypsat do konzole prohlížeče pomocí kódu `console.log(window);` v javaskriptovém souboru pod parametrem `gapi`.

```
                                                                                                                    main.js:79
▼{load: ƒ, loaded_0: ƒ}
  ▶auth: {authorize: ƒ, checkSessionState: ƒ, getAuthHeaderValueForFirstParty: ƒ, getToken: ƒ, getVersionInf…
  ▶auth2: {init: ƒ, authorize: ƒ, _gt: ƒ, enableDebugLogs: ƒ, getAuthInstance: ƒ, …}
  ▼client:
    ▶AuthType: {LH: "auto", NONE: "none", CP: "oauth2", NH: "1p", AUTO: "auto", …}
    ▶getToken: ƒ (a)
    ▶init: ƒ (a)
    ▶load: ƒ (a,c,f,g)
    ▶newBatch: ƒ (a)
    ▶newHttpBatch: ƒ (a)
    ▶newRpcBatch: ƒ ()
    ▶register: ƒ (a,c)
    ▶request: ƒ (a,c)
    ▶rpcRequest: ƒ (a,c,f)
    ▶setApiKey: ƒ (a)
    ▶setApiVersions: ƒ (a)
    ▶setToken: ƒ (a,c)
    ▶youtube: {activities: {…}, kB: {…}, captions: {…}, channelBanners: {…}, channelSections: {…}, …}
    ▶__proto__: Object
  ▶config: {get: ƒ, update: ƒ}
  ▶load: ƒ (a,b)
   loaded_0: null
  ▶platform: {go: ƒ}
  ▶widget: {make: ƒ}
  ▶_: {b: ƒ, _DumpException: ƒ, aa: Array(34), PK: ƒ, jb: ƒ, …}
  ▶__proto__: Object
```

Obrázek 12 Výpis parametrů Google API (autor)

Obrázek 12 zobrazuje různé metody Google API, které lze nastavit, jako je přihlášení na Google účet pomocí parametru `auth`. Pro aplikaci je nejdůležitější nastavení klientské metody `load`

a `setApiKey`, ve kterém se definuje unikátní klíč a je důležitý ke klientském přístupu k YouTube API a `youtube`, který obsahuje metodu `search` pro hledání videí.

```
window.gapi_onload = function() {
    gapi.client.setApiKey("A[redacted]w");
    gapi.client.load("youtube", "v3", function() {
        console.log("YouTube Data API v3 je připraveno");
    });

}
```

Obrázek 13 Nastavení klíče a načtení Youtube API v3 (autor)

Pro přístup byl vytvořen kód, který iniciuje registraci k Youtube API, obsahuje unikátní API klíč, který lze získat z oficiálních vývojářských stránek Google. Dále stanovujeme, co se má načíst, v tomto případě YouTube Data API s verzí tři a pro potvrzení připojení vypíšeme do konzole prohlížeče, že je API připraveno k použití.

```
var request = gapi.client.youtube.search.list({
    part: "snippet",
    type: "video",
    q: encodeURIComponent($("#search").val()).replace(/%20|%2C/g, "+"),
    maxResults: 5,
    videoCategoryId: 10,
    order: "viewCount",
});
```

Obrázek 14 Definování parametrů omezení pro hledání YouTube videí (autor)

Obrázek 14 obsahuje kód, který stanoví parametry omezení pro hledaní videí, který se definuje v proměnné `request`. Parametr `q` obsahuje textový řetězec, který se má najít, `maxResults` nastavuje počet videí k zobrazení, `videoCategoryId` nastavuje, který typ videí chceme zobrazit, v tomto případě číslo deset stanovuje typ hudbu. Další parametry, které lze nastavit, jsou na vývojářských stránkách Google (Google, 2019).

8.1.5.2 SoundCloud API

SoundCloud API je API hudební platformy, ze kterého lze získat informace o hudbě. Výhodou tohoto API je neomezená kvóta pro připojení k API už při základním nastavení pro vývojáře.

```
fetch("https://api.soundcloud.com/tracks?client_id=e[redacted]f&q="+ $("#search").val())
.then(function(response2) {
```

Obrázek 15 Metoda fetch pro přístup k SoundCloud API (autor)

Pro přístup k API SoundCloudu je využit metoda `fetch`, který obsahuje URL API s pararametry `client_id` obsahující unikátní API klíč vygenerovaný ze stránek SoundCloud pro každého registrovaného vývojáře a `q` obsahující textový řetězec, v případě této aplikace řetězec, který zadáme do hledacího políčka, tzn. stanovuje, co se má najít. Pokud je vše v pořádku, pokračuje to funkcí s parametrem `response2`, který představuje pole s objekty. Jednotlivé objekty, které reprezentují jednotlivé nalezené hudby, obsahují jejich údaje ve formě klíč – hodnota.

8.2 Přidání komponent PWA

Jelikož vyžaduje přístup PWA zabezpečené připojení, byla aplikace vyvíjena pomocí lokálního serveru vytvořeného pomocí nástroje Web Server for Chrome, kdy přístup PWA nevyžaduje zabezpečené připojení pro lokální server. Aplikace je pak nasazena na webový hosting a z něho je finálně testována.

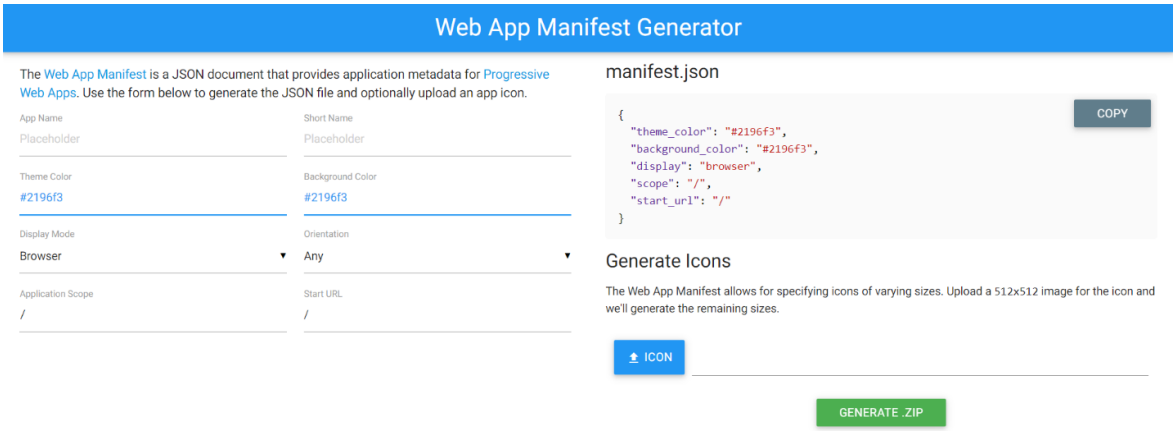
Pro implementaci přístupu PWA byly definovány a vytvořeny jeho základní komponenty, aplikační *shell*, manifestační soubor a *service worker*. Pro aplikaci byla také vytvořena funkce pro zobrazení notifikace.

8.2.1 Aplikační shell

Aplikačním *shellem* se zde rozumí soubory aplikace, které jsou nutné pro chod aplikace bez připojení k internetu. Mezi definovanými soubory patří *index.html*, *menu-content.html* a *style.css* jako základní stavební kostra aplikace, *menu-content.js* pro zobrazení obsahu menu a *push-notification.js*, který obsahuje možnost *push* notifikace bez připojení k internetu a různé obrázky patřící k základní kostře aplikace.

8.2.2 Manifest

Manifestační soubor lze vytvořit ručně nebo pomocí různých nástrojů. Tento soubor byl vytvořen pomocí generátoru Web App Manifest společnosti Firebase, který je dostupný na adrese https://app-manifest.firebaseapp.com/.



Obrázek 16 Generátor souboru manifest společnosti Firebase (Firebase, 2019)

Výhodou této aplikace je, že po nahrání ikonky se vygenerují ikonky ve zbylých rozlišeních. Po zadání zbylých potřebných informací se vygeneruje JSON soubor, který se následně připojí k aplikaci pomocí značky `<link>`. Manifest je většinou soubor s příponou *.json*, v poslední době se začíná využívat i přípona *.webmanifest*, které doporučuje konsorcium W3C, ale nadále spadá do sekce návrhu (*Working Draft*) konsorcia W3C (W3C, 2019). Využití obou přípon je možné.

```
    >
    <link rel="manifest" href="manifest.json">
</head>
```

Obrázek 17 Připojení manifestu k HTML souboru (autor)

O potvrzení přidání manifestu se lze přesvědčit ve vývojářském nástroji prohlížeče Chrome nebo otevřením aplikace v prohlížeči na mobilním zařízení podporující manifest, kde by se mělo objevit vyskakovací oznámení o možnosti přidání aplikace na plochu.

8.2.3 Service Worker

Pro uložení aplikace do mezipaměti je vytvořen soubor *sw.js*, který říká prohlížeči, jaké soubory se mají uložit do mezipaměti a lze použít v případě bez připojení k internetu. Jako strategii ukládání do mezipaměti byla vybrána strategie *cache first* (první z mezipaměti), kdy prohlížeč první načte soubory z mezipaměti a až poté jsou soubory aktualizovány, takže po příští návštěvě aplikace bude mít uživatel aktuální verzi (Novicki, 2019).

```
if ('serviceWorker' in navigator) {
  window.addEventListener('load', function() {
    navigator.serviceWorker.register('/sw.js').then(function(registration) {
      console.log('ServiceWorker registrován: ', registration);
    }, function(err) {
      console.log('ServiceWorker registrace selhala: ', err);
    });
  });
}
```

Obrázek 18 Registrace service worker (autor)

Obrázek 18 obsahuje kód, který je přidán v javaskriptovém souboru *main.js* a jeho funkcí je registrace *service worker*. Soubor *sw.js* je pak registrován jako *service worker*. Důležité je umístit soubor *sw.js* do *rootu* aplikace k zaregistrování všech jeho souborů.

```
var cacheName = "soundforyou";
var staticAssets = [
  '/',
  '/index.html',
  '/menu-content.html',
  '/css/style.css',
  '/js/menu-content.js',
  '/js/push-notification.js',
  '/assets/search-icon.png',
  '/assets/soundforyou-icon.png',
  '/assets/icons/icon-192x192.png',
  '/manifest.json',
]
```

Obrázek 19 Název a definování souborů pro mezipaměť (autor)

V souboru *sw.js* byly prvně vytvořeny proměnné `cacheName` jako název mezipaměti, pod kterým se soubory ukládají a `staticAssets` obsahující pole, které definuje soubory k uložení do mezipaměti. Obvykle to bývají statické neměnné soubory, nejčastěji kostra úvodní

stránky. V poli se definuje i javaskriptové soubory pro zobrazení obsahu v menu a možnost *push* notifikace v offline režimu.

```
self.addEventListener('install', function(event) {
  event.waitUntil(
    caches.open(cacheName).then(function(cache) {
      return cache.addAll(staticAssets);
    })
  );
});
```

Obrázek 20 Událost install pro service worker (autor)

Proměnné `self`, která zde zastupuje *service worker*, se definuje událost `install`. Dále se ve funkci definuje nová mezipaměť s názvem v proměnné `cacheName` a do něho se uloží všechny nadefinované soubory v proměnné `staticAssets`. Metodou `waitUntil()` *service worker* nepokračuje v kódu, v tomto případě, dokud se neuloží všechny nadefinované soubory do mezipaměti. Pokud dojde k chybě uložení nějakého souboru ze `staticAssets`, např. špatně definovaným názvem souboru, pak se *service worker* zastaví v provádění dalších událostí a nedojde k aktualizování ani přidání mezipaměti. Díky této metodě se zamezí chybám při manipulaci se *service worker* (Bushyn, 2019).

```
self.addEventListener('activate', function(event) {
  event.waitUntil(
    caches.keys().then(function(cacheNames) {
      return Promise.all(
        cacheNames.filter(function(cacheName) {
        }).map(function(cacheName) {
          return caches.delete(cacheName);
        })
      );
    })
  );
});
```

Obrázek 21 Událost activate pro service worker (autor)

Další události `activate` se spustí, pokud není přechozí verze *service worker* využívána (např. je pořád otevřená webová stránka, která stále využívá starý service worker, protože nedošlo k její aktualizaci) a došlo k úspěšné instalaci nového *service worker*. Tato událost je důležitá k vyřazení staré mezipaměti.

```
self.addEventListener('fetch', function(event) {
  event.respondWith(
    caches.match(event.request).then(function(response) {
      return response || fetch(event.request);
    })
  );
});
```

Obrázek 22 Událost fetch pro service worker (autor)

Událost `fetch` se spustí kdykoliv prohlížeč požaduje zdroje. V události fetch se definuje, že se má prohlížeč pokusit načíst soubory z mezipaměti, pokud mezipaměť nemá žádné zdroje nebo neexistuje, tak se prohlížeč pokusí načíst aplikaci ze sítě.

8.2.4 Push notifikace

V aplikaci byla využita technologie *push* notifikace, kdy aplikace zobrazí *push* notifikaci o oznámení výpadku připojení k internetu při kliknutí na tlačítko pro vyhledávání nebo spuštění hudby v aplikaci.

```
Notification.requestPermission();
```

Obrázek 23 Požadavek o zobrazení notifikace (autor)

V souboru *push-notification.js* se prvně definuje kód (Obrázek 23), pomocí kterého se aplikace zeptá uživatele, jestli si přeje zobrazovat notifikace.

```
$(document).on("click", ".btn-search, .btn-play, .btn-play2", function() {
  if(!navigator.onLine){
    return displayNotification();
  }
});
```

Obrázek 24 Podmínka pro zobrazení notifikace v události click (autor)

Tato podmínka vrací funkci `displayNotification()`, pokud se zařízení nachází v offline režimu. Podmínka se spustí, pokud uživatel klikne na tlačítko pro vyhledávání nebo spuštění hudby v aplikaci. Důležité je zahrnout soubor, který tuto událost obsahuje, do mezipaměti.

```
function displayNotification() {
  if (Notification.permission == 'granted') {
    navigator.serviceWorker.getRegistration().then(function(reg) {
      var options = {
        body: 'Zkontrolujte připojení a obnovte aplikaci.',
        icon: 'assets/icons/icon-192x192.png',
        vibrate: [200, 100, 200],
      };
      reg.showNotification('Stav připojení: Offline', options);
    });
  }
}
```

Obrázek 25 Funkce pro zobrazení notifikace (autor)

Funkce `displayNotification()` obsahuje podmínku, která definuje, co se má stát, pokud uživatel povolí notifikace. Proměnná `options` definuje obsah notifikace a také interval vibrace v milisekundách, v tomto případě 200ms vibruje, 100ms přestane vibrovat a 200ms znovu vibruje. Metodou `showNotification` se zobrazí notifikace, pokud se zavolá funkce `displayNotification()` a obsahuje dva parametry, první obsahuje název notifikace a druhý definuje nastavení notifikace, které je zde definováno v proměnné `options`.

8.3 Testování vytvořené mobilní aplikace

V této podkapitole jsou testovány různé aspekty aplikace definované autorem. V aplikaci došlo především k testování přístupu PWA.

8.3.1 Testování aspektů PWA aplikace na vybraných webových prohlížečích

Webová aplikace byla testována autorem práce na různých prohlížečích mobilních zařízení, jejichž výběr odpovídá jejich celosvětové oblíbenosti. Pro testování na prohlížeči Safari byl vybrán iPhone 7 s platformou iOS 12.2. Testování na ostatních prohlížečích bylo vybráno zařízení s platformou Android 7.0.

Tabulka 19 Testování aplikace na mobilních prohlížečích (autor)

Prohlížeč	**Zobrazení vyskakovacího okna pro přidání na domovskou obrazovku**	**Přidání na domovskou obrazovku**	**Zobrazení aplikace bez horního panelu**	**Push notifikace**	**Zobrazení uvítací obrazovky**
Google Chrome (73.0.3683.90)	Ano	Ano	Ano	Ano	Ano
Safari (12.2)	Ne	Ano	Ano	Ne	Ne
Mozilla Firefox (66.0.2)	Ne	Ano	Ano	Ano	Ano
Microsoft Edge (42.0.2.3367)	Ano	Ano	Ano	Ano	Ano
Opera (51.3.2461.138727)	Ne	Ano	Ano	Ano	Ano

Tabulka 19 ukazuje, že prohlížeče Google Chrome a Microsoft Edge umožňují všechny definované funkce. Prohlížeče Mozilla Firefox a Opera nezobrazují vyskakovací okno pro přidání na domovskou obrazovku, místo toho prohlížeč Mozilla Firefox zobrazí ikonku vedle vyhledávacího pole k jeho přidání. Všechny definované prohlížeče umožňují přidání jakékoliv webové stránky nebo aplikace alespoň jako záložku na domovskou obrazovku. S nejhorší podporou PWA skončil prohlížeč Safari od společnosti Apple, která do dnešní doby[1] nepodporuje *push* notifikace pro webové aplikace a také zobrazení uvítací obrazovky.

8.3.2 Testování dalších aspektů aplikace

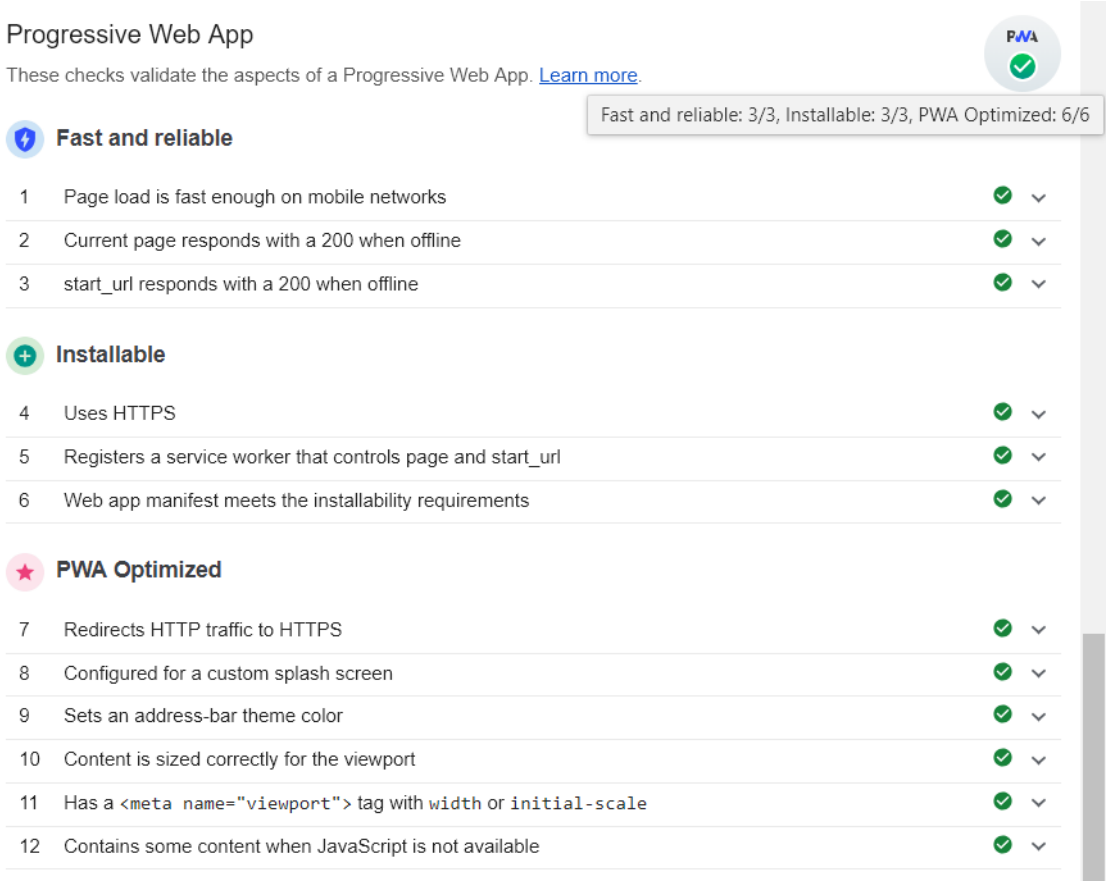
Aplikace byla ještě testována v dalších aspektech jako je validita HTML kódu nebo responzivita aplikace na mobilních zařízeních. Aplikace načte správně elementy pro zařízení o velikostí šířky alespoň 320px. Pro zařízení s obrazovkou minimální šířky 768 pixelů začne aplikace měnit rozložení jejích elementů. Rozložení elementů aplikace na počítači je obsažen v příloze C. Validace HTML souboru proběhla pomocí validátoru konsorcia W3C, který nahlásil pouze

[1] k datu vytvoření bakalářské práce – květen 2019

upozornění pro použití parametru `user-scalable=no` v meta tagu oblasti rozvržení zabraňující škálování aplikace, který je u aplikace nativního charakteru typický.

8.3.3 Výsledek testování pomocí nástroje Lighthouse

Testování aspektů PWA proběhlo také pomocí nástroje Lighthouse, který je součástí vývojářského nástroje v prohlížeči Chrome.



Obrázek 26 Testování přístupu PWA aplikace pomocí nástroje Lighthouse (autor)

Obrázek 26 zobrazuje hodnocení aspektů PWA pomocí nástroje Lighthouse. Aplikace splňuje všechny aspekty PWA, je rychlá a spolehlivá, instalovatelná a optimalizovaná pro přístup PWA.

8.4 Shrnutí

V této části práce došlo k návrhu a tvorbě webové aplikace přizpůsobené především pro mobilní zařízení. Bylo vytvořeno logo, ikonka a drátové modely aplikace, podle kterých byla vytvořena základní kostra aplikace pro mobilní a desktopové zařízení. Byla přidána hlavní funkce aplikace, a to zobrazení a přehrání hudby obou hudebních platforem pomocí javaskriptového volání API. Dále byl implementován přístup PWA a došlo k testování finální aplikace. Výstupem této kapitoly je webová aplikace s přístupem PWA, která je zároveň hlavním cílem této práce. V příloze C je zobrazena některá funkcionalita aplikace na prohlížeči Chrome (verze 73.0.3683.90) v mobilním zařízení LG G6 s platformou Android 7.0.

Závěr

Hlavním cílem této bakalářské práce bylo navrhnout a vytvořit webovou aplikaci SoundForYou pomocí metodiky MMSP a přístupů *mobile first* a Progressive Web Apps. Vedlejšími požadavky úlohy bylo vytvoření loga a ikonky aplikace.

V teoretické části byly stručně popsány typy mobilních aplikací a přístup Progressive Web Apps. Praktická část práce byla rozdělena na dvě části. V první části byly definovány požadavky na funkcionalitu aplikace podle metodiky MMSP a návrh aplikace. V druhé části došlo k implementaci daného návrhu aplikace pomocí webových technologií a její testování. V implementaci byly představeny důležité a zajímavé části kódů aplikace.

Aplikace umožňuje přidání ikonky na domovskou obrazovku. Možnost uložení nadefinovaných souborů zajistí rychlé načtení aplikace a možnost offline načtení kostry aplikace např. namísto klasické „downasaurské“ hry na prohlížeči Google Chrome. Při testování aplikace prokázala kvality Progressive Web Apps. Aplikace je přístupna na webové adrese https://soundforyou.000webhostapp.com/.

Přínosem této bakalářské práce je pohled na proces tvorby webové aplikace zaměřené na mobilní zařízení, od jejího definování návrhu a požadavků přes implementaci a testování. Dalším přínosem je rozšiřování poznatků o přístupu Progressive Web Apps, jelikož v České republice a nejenom zde není tento přístup příliš známý. Další aplikace, které jsou založené na tomto přístupu, jsou k vyzkoušení např. na stránkách https://pwa.rocks/.

Vytvořená webová aplikace je zaměřena pouze na uživatele platforem YouTube a SoundCloud, což je omezení, které velmi ohraničuje množství potenciálních zákazníků. V budoucnu by se měla aplikace zaměřit i na uživatele jiných platforem. Další nevýhodou je, že aplikace zobrazuje videa pro obě platformy zároveň, kde se stává, že uživatelům odcházejí data, i když pro konkrétní platformu nechtějí vyhledat hudbu. Tato skutečnost se dá vylepšit tím, že by se přidala funkcionalita, která by umožnila uživatelům si vybrat, ze které platformy by chtěli načítat videa. Další vylepšením se plánuje možnost vybrání jazyka aplikace, přidání vlastního seznamu skladeb a textu hudby.

Použitá literatura

Apple. 2018. APNs Overview. *Apple.* [Online] 4. Červen 2018. [Citace: 1. Květen 2019.] https://developer.apple.com/library/archive/documentation/NetworkingInternet/Conceptual/RemoteNotificationsPG/APNSOverview.html#//apple_ref/doc/uid/TP40008194-CH8-SW1.

archer-soft. 2017. Native vs Hybrid vs Mobile Web Apps. *archer-soft.* [Online] 23. Leden 2017. [Citace: 16. Březen 2019.] https://www.archer-soft.com/en/blog/native-vs-hybrid-vs-mobile-web-apps.

Bartík, Marek. 2017. Proč a jak psát progresivní webové aplikace. *Ackee.* [Online] 11. Leden 2017. [Citace: 11. Březen 2019.] https://www.ackee.cz/blog/proc-a-jak-psat-progresivni-webove-aplikace/.

Bevacqua, Nicolas. 2017. Making a Simple Site Work Offline with ServiceWorker. *CSS-Tricks.* [Online] 8. Květen 2017. [Citace: 25. Duben 2019.] https://css-tricks.com/serviceworker-for-offline/.

Bootstrap. 2019. Layout - Overview. *Bootstrap.* [Online] 2019. [Citace: 27. duben 2019.] https://getbootstrap.com/docs/4.3/layout/overview/.

Břoušek, Pavel. 2017. *Evaluation and usage of Google Progressive Web Apps technology.* Brno : Masarykova Univerzita, 2017. Bakalářská práce.

Bushyn, Arseniy. 2019. ExtendableEvent.waitUntil(). *MDN.* [Online] 17. Duben 2019. [Citace: 22. Duben 2019.] https://developer.mozilla.org/en-US/docs/Web/API/ExtendableEvent/waitUntil.

Castledine, Earle, Eftos, Myles a Wheeler, Max. 2011. *Vytváříme mobilní web a aplikace.* místo neznámé : SitePoint Pty. Ltd, 2011. ISBN 978-80-251-3763-5.

Deveria, Alexis. 2019. Push API. *Can I use.* [Online] Březen 2019. [Citace: 25. Duben 2019.] https://caniuse.com/#search=push%20notification.

—. 2019. Service Workers. *Can I use.* [Online] Březen 2019. [Citace: 25. Duben 2019.] https://caniuse.com/#feat=serviceworkers.

—. 2019. Web App Manifest. *Can I use.* [Online] Březen 2019. [Citace: 25. Duben 2019.] https://caniuse.com/#feat=web-app-manifest.

Firebase. 2019. Web App Manifest. *Web App Manifest.* [Online] 2019. [Citace: 13. Duben 2019.] https://app-manifest.firebaseapp.com/.

Gasston, Peter. 2013. *Moderní web - Perfektní stránky v každém prohlížeči a na každém zařízení.* místo neznámé : No Starch Press, 2013. ISBN 978-80-7413-280-3.

Gaunt, Matt a Kinlan, Paul. 2019. The Web App Manifest. *Google Developers.* [Online] 12. Březen 2019. [Citace: 17. Březen 2019.] https://developers.google.com/web/fundamentals/web-app-manifest/.

Gaunt, Matt. 2019. Service Workers: an Introduction. *Google Developers.* [Online] 12. Únor 2019. [Citace: 17. Březen 2019.] https://developers.google.com/web/fundamentals/primers/service-workers/.

Google Developers. 2018. Lighthouse. *Google Developers.* [Online] 21. Srpen 2018. [Citace: 19. Březen 2019.] https://developers.google.com/web/tools/lighthouse/.

—. 2018. Your First Progressive Web App. *Google Developers.* [Online] 29. Říjen 2018. [Citace: 14. Březen 2019.] https://codelabs.developers.google.com/codelabs/your-first-pwapp/#0.

Google. 2017. Web Server for Chrome. *chrome web store.* [Online] 22. Září 2017. [Citace: 23. Duben 2019.] https://chrome.google.com/webstore/detail/web-server-for-chrome/ofhbbkphhbklhfoeikjpcbhemlocgigb.

—. 2019. YouTube Data API - Search: list. *YouTube Data API.* [Online] 4. Duben 2019. [Citace: 13. Duben 2019.] https://developers.google.com/youtube/v3/docs/search/list.

iQuest. 2018. Hybridní, nativní nebo webové aplikace? *iQuest.* [Online] 2018. [Citace: 13. Březen 2019.] http://www.hybridniaplikace.cz/srovnani.html.

Janovský, Dušan. 2017. HTML příručka. *Jak psát web.* [Online] 2017. [Citace: 19. Březen 2019.] https://www.jakpsatweb.cz/html/.

Kadlec, Tim. 2013. *Responzivní design - profesionálně.* místo neznámé : Pearson Education, Inc, 2013. ISBN 978-80-251-4345-2.

LePage, Pete. 2019. Add to Home Screen. *Google Developers.* [Online] 12. Únor 2019. [Citace: 16. Březen 2019.] https://developers.google.com/web/fundamentals/app-install-banners/.

Novicki, David. 2019. PWA Asset Caching Strategies. *codeburst.io.* [Online] 29. Leden 2019. [Citace: 22. Duben 2019.] https://codeburst.io/pwa-asset-caching-strategies-8a20c31b2181.

Oragui, David. 2018. 11 Examples of Progressive Web App. *The Manifest.* [Online] 10. Duben 2018. [Citace: 18. Březen 2019.] https://medium.com/@the_manifest/11-examples-of-progressive-web-apps-944f6db25a5a.

Osmani, Addy. 2019. The App Shell Model. *Google Developers.* [Online] 12. Únor 2019. [Citace: 17. Březen 2019.] https://developers.google.com/web/fundamentals/architecture/app-shell.

Prasad, Perera. 2018. A Comparison: Native Apps vs Web Apps. *Technocrat.* [Online] 2018. [Citace: 16. Březen 2019.] https://www.technocrat.com.au/blog/comparison-native-apps-vs-web-apps.

Rejnková, Petra. 2011. *Lokalizace a přizpůsobení metodiky OpenUP.* Praha : Vysoká škola ekonomická, 2011. Diplomová práce.

StatCounter. 2019. Mobile, Tablet & Console Operating System Market Share Worldwide. *StatCounter.* [Online] 2019. [Citace: 11. Březen 2019.] http://gs.statcounter.com/os-market-share/mobile-tablet-console/worldwide/#monthly-201212-201901.

Statista. 2019. Number of smartphone users worldwide from 2014 to 2020 (in billions). *Statista.* [Online] 2019. [Citace: 11. Březen 2019.] https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/.

Sublime Text. 2019. A sophisticated text editor for code, markup and prose. *Sublime Text.* [Online] 6. Duben 2019. [Citace: 10. Duben 2019.] https://www.sublimetext.com/.

Thakur, Parbat. 2018. *Evaluation and Implementation of Progressive Web Application.* Helsinki : Helsinki Metropolia University of Applied Sciences, 2018. Bakalářská práce.

W3C. 2019. Web App Manifest - Living Document. *W3C Working Draft.* [Online] 18. Duben 2019. [Citace: 24. Duben 2019.] https://www.w3.org/TR/appmanifest/.

Wikipedie. 2019. JavaScript. *Wikipedie.* [Online] 12. Leden 2019. [Citace: 14. Březen 2019.] https://cs.wikipedia.org/wiki/JavaScript.

—. 2019. jQuery. *Wikipedie.* [Online] 22. Duben 2019. [Citace: 23. Duben 2019.] https://en.wikipedia.org/wiki/JQuery.

—. 2019. Kaskádové styly. *Wikipedie.* [Online] 1. Leden 2019. [Citace: 19. Březen 2019.] https://cs.wikipedia.org/wiki/Kask%C3%A1dov%C3%A9_styly.

—. 2018. Mobilní aplikace. *Wikipedie.* [Online] 27. Únor 2018. [Citace: 15. Březen 2019.] https://cs.wikipedia.org/wiki/Mobiln%C3%AD_aplikace.

—. 2017. Webová aplikace. *Wikipedie.* [Online] 4. Říjen 2017. [Citace: 15. Březen 2019.] https://cs.wikipedia.org/wiki/Webov%C3%A1_aplikace.

Xia, Vincent. 2017. What is Mobile First Design? Why It's Important & How To Make It? *Medium.* [Online] 21. Prosinec 2017. [Citace: 15. Březen 2019.] https://medium.com/@Vincentxia77/what-is-mobile-first-design-why-its-important-how-to-make-it-7d3cf2e29d00.

Přílohy

Příloha A: Logo a ikonka aplikace

SFY

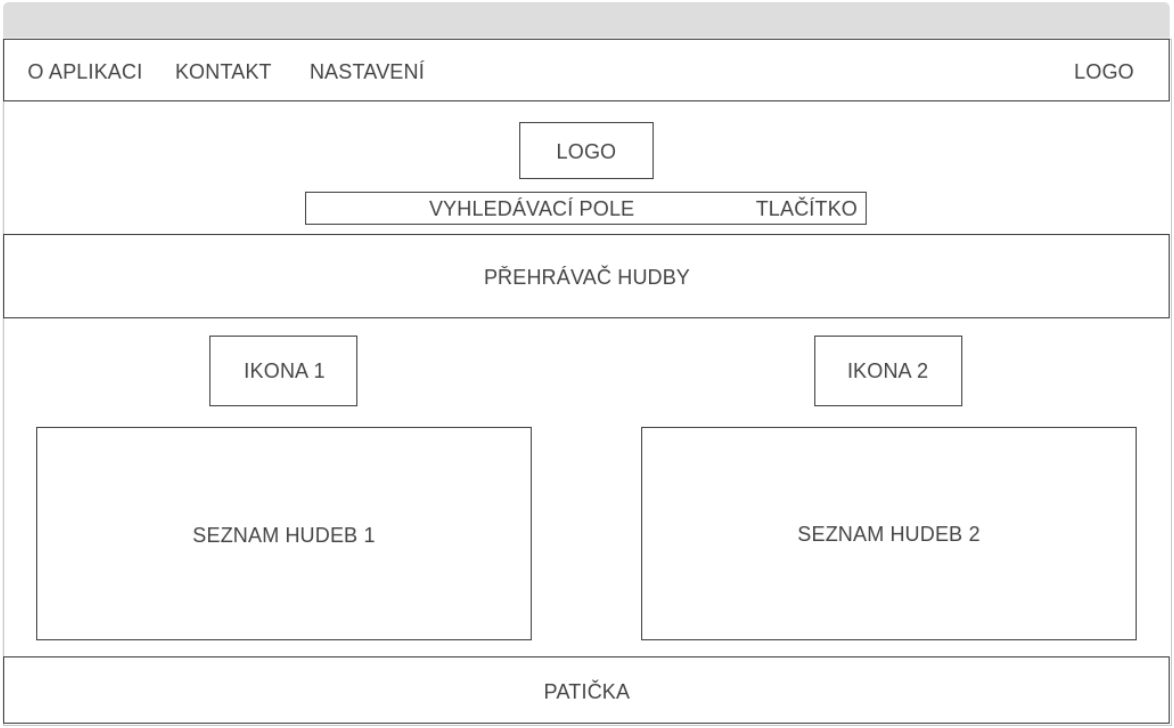
Logo aplikace.

SFY

Ikonka aplikace.

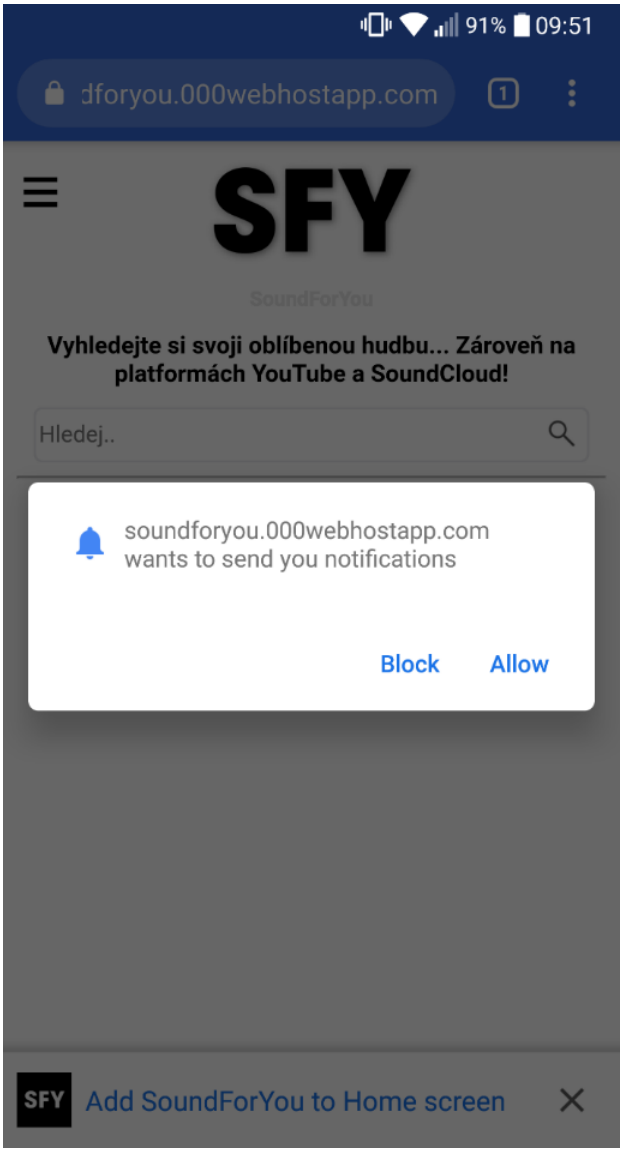
Příloha B: Drátové modely aplikace

Drátový model aplikace pro mobily.

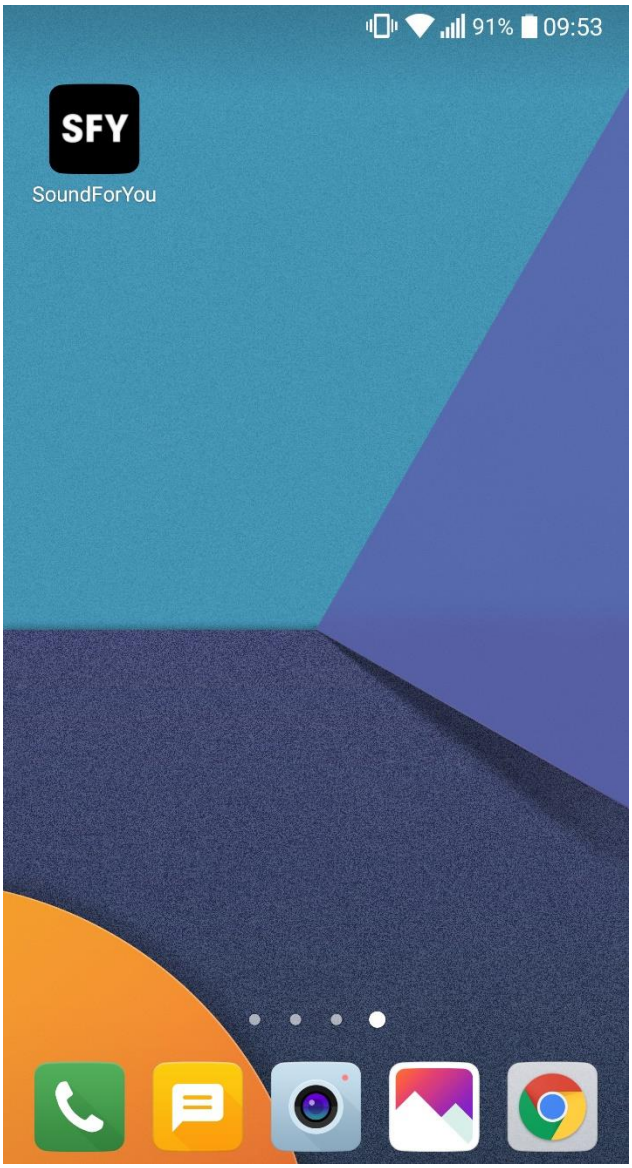


Drátový model aplikace pro tablety a počítače.

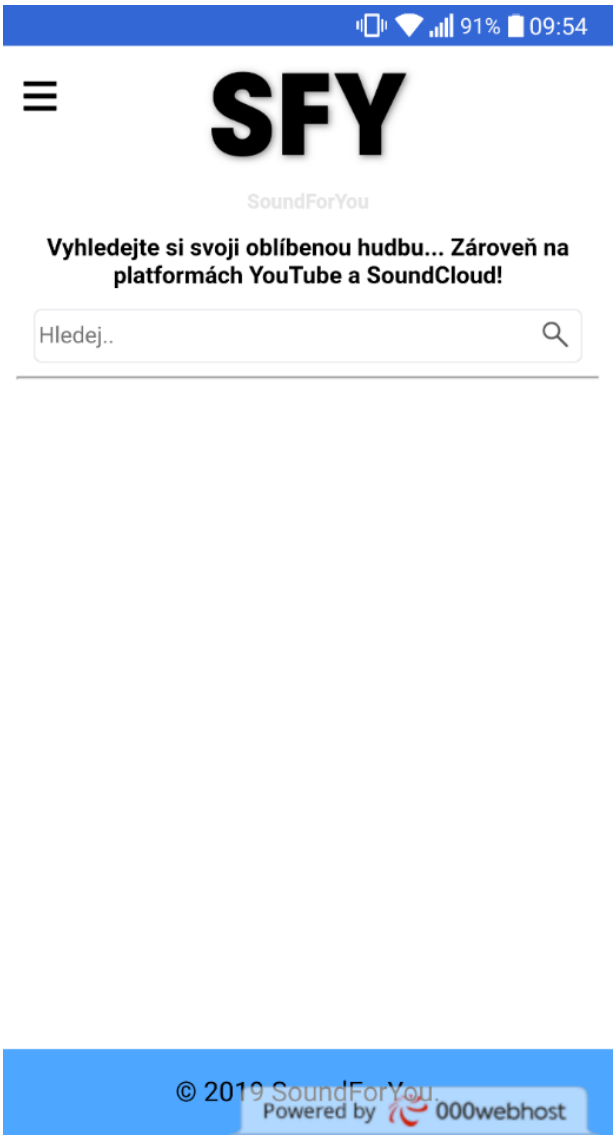
Příloha C: Snímky aplikace



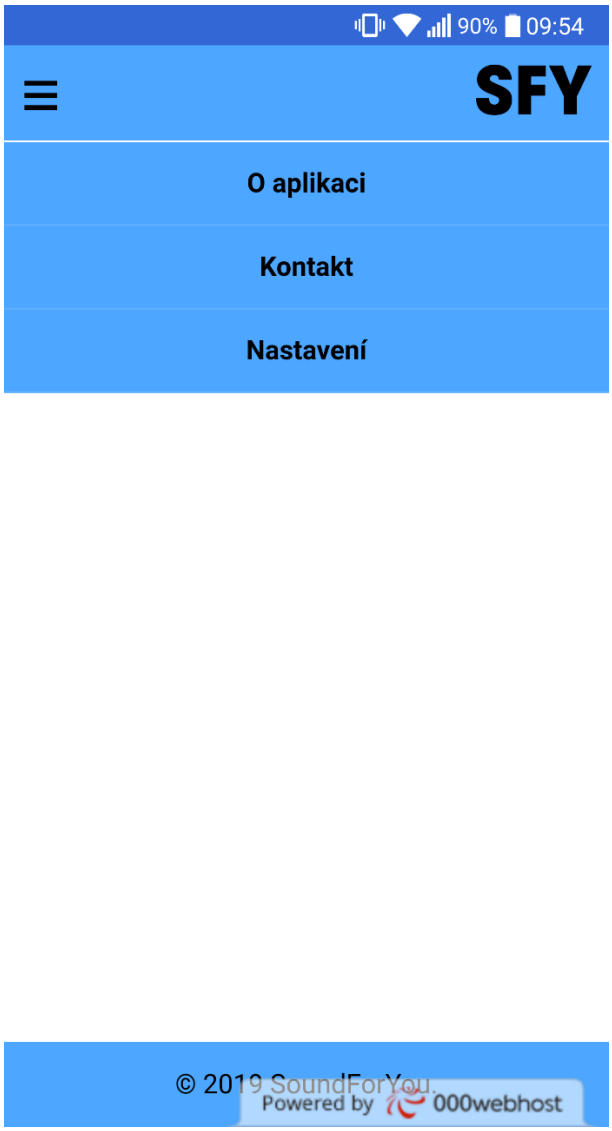
Snímek žádosti o povolení notifikací a dialogové okno přidání aplikace na plochu.



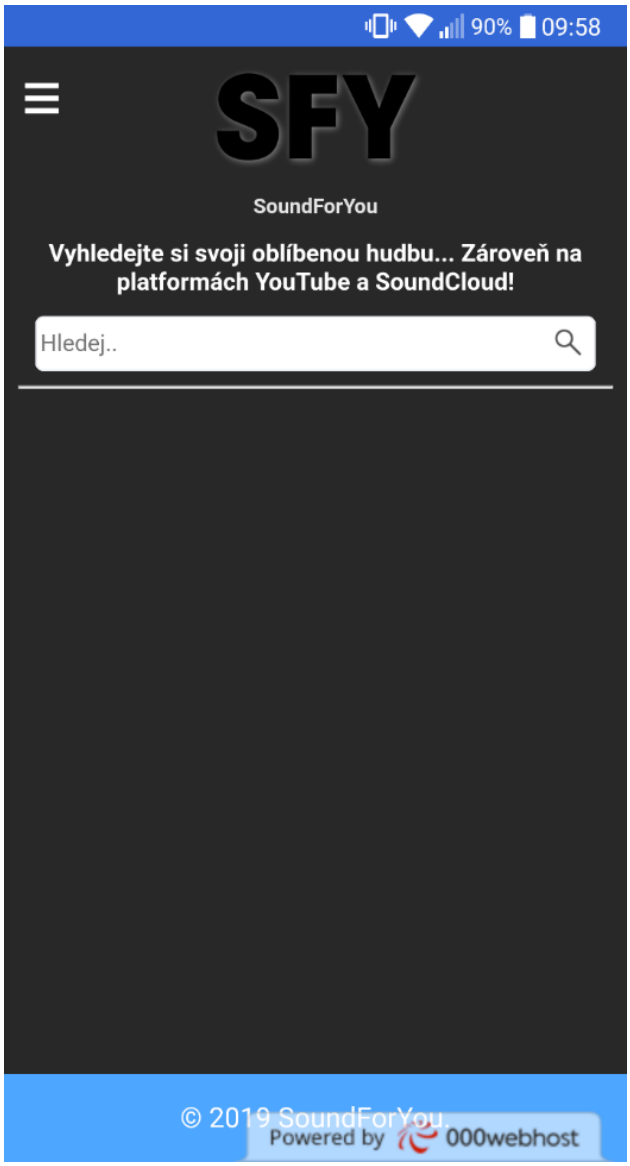
Snímek ikonky aplikace na ploše.



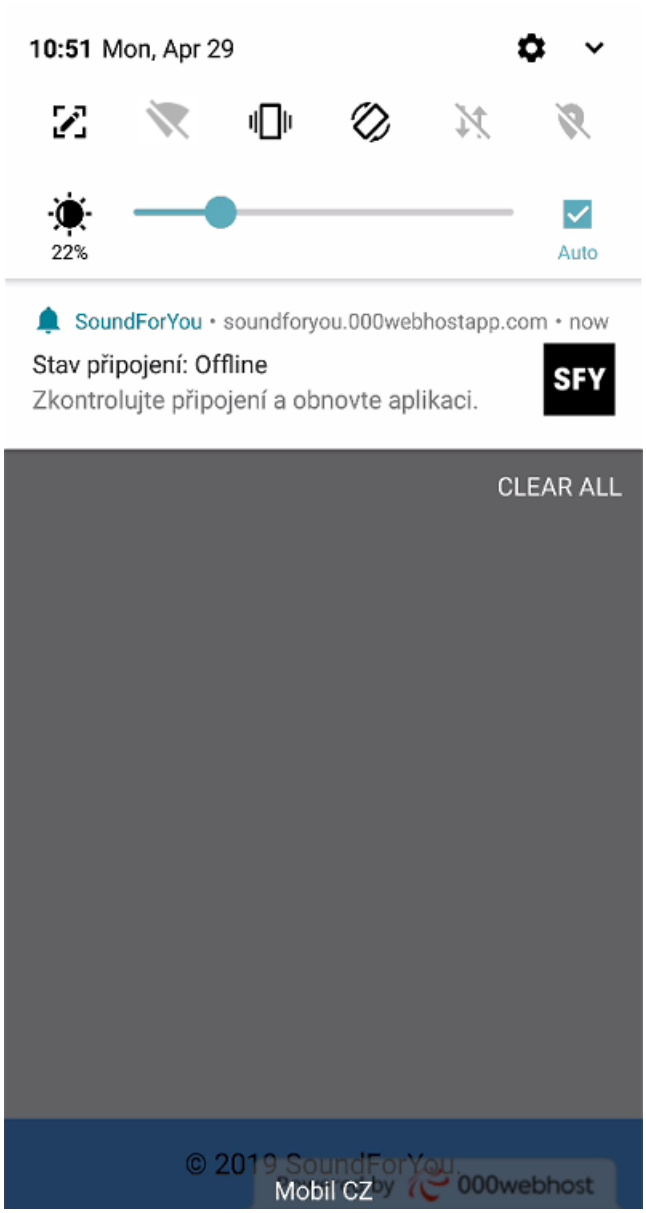
Snímek aplikace bez horního panelu prohlížeče po zapnutí přes ikonku.



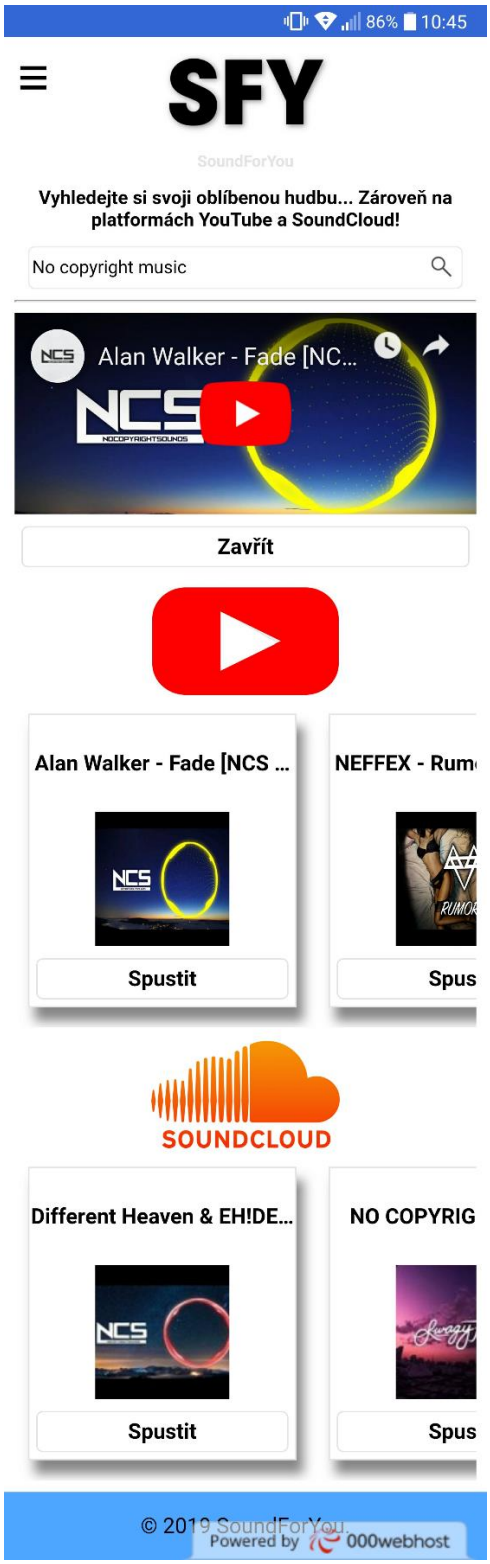
Snímek menu aplikace.



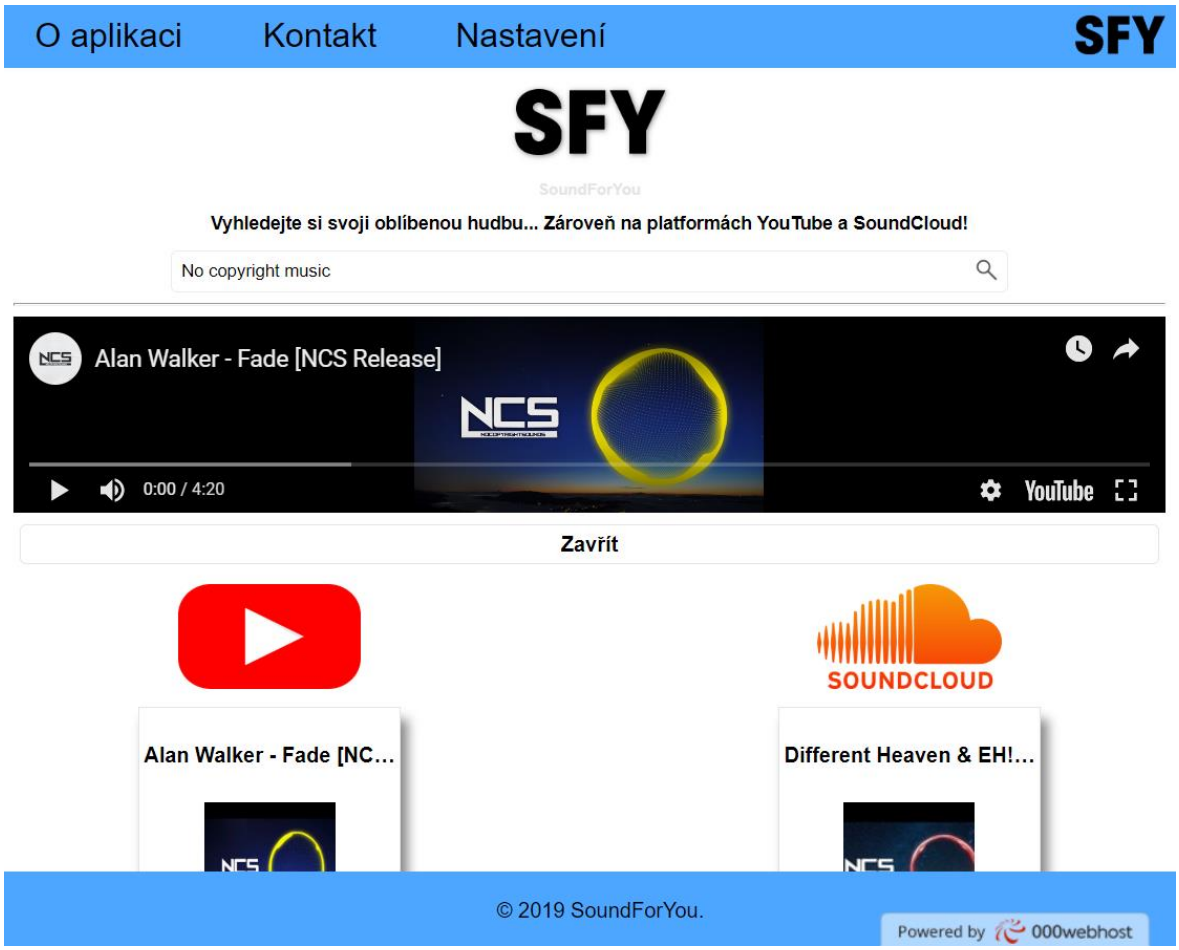
Snímek aplikace s tmavým tématem.



Snímek notifikace upozornění o ztrátě připojení.



Prodloužený snímek aplikace s příkladem vyhledání a přehrávačem.



Snímek aplikace na počítači s příkladem vyhledání a přehrávačem.

Příloha D: Zdrojový kód aplikace

hoan01_sfy.zip